

TraDemGen

1.00.2

Generated by Doxygen 1.6.1

Sat Jun 27 21:24:53 2015

Contents

1	TraDemGen Documentation	1
1.1	Getting Started	1
1.2	TraDemGen at SourceForge	1
1.3	TraDemGen Development	1
1.4	External Libraries	2
1.5	Support TraDemGen	2
1.6	About TraDemGen	2
2	People	2
2.1	Project Admins (and Developers)	2
2.2	Retired Developers	2
2.3	Contributors	2
2.4	Distribution Maintainers	3
3	Coding Rules	3
3.1	Default Naming Rules for Variables	3
3.2	Default Naming Rules for Functions	3
3.3	Default Naming Rules for Classes and Structures	3
3.4	Default Naming Rules for Files	3
3.5	Default Functionality of Classes	4
4	Copyright and License	4
4.1	GNU LESSER GENERAL PUBLIC LICENSE	4
4.1.1	Version 2.1, February 1999	4
4.2	Preamble	4
4.3	TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION	5
4.3.1	NO WARRANTY	9
4.3.2	END OF TERMS AND CONDITIONS	10
4.4	How to Apply These Terms to Your New Programs	10
5	Documentation Rules	10
5.1	General Rules	10
5.2	File Header	11
5.3	Grouping Various Parts	12
6	Main features	12
6.1	Demand generation	12

6.2	Other features	13
7	Make a Difference	13
8	Make a new release	13
8.1	Introduction	13
8.2	Initialisation	13
8.3	Release branch maintenance	14
8.4	Commit and publish the release branch	14
8.5	Create distribution packages	14
8.6	Upload the HTML documentation to SourceForge	15
8.7	Generate the RPM packages	15
8.8	Update distributed change log	15
8.9	Create the binary package, including the documentation	15
8.10	Upload the files to SourceForge	16
8.11	Make a new post	16
8.12	Send an email on the announcement mailing-list	16
9	Installation	16
9.1	Table of Contents	16
9.2	Fedora/RedHat Linux distributions	16
9.3	TraDemGen Requirements	17
9.4	Basic Installation	17
9.5	Compilers and Options	18
9.6	Compiling For Multiple Architectures	18
9.7	Installation Names	19
9.8	Optional Features	20
9.9	Particular systems	20
9.10	Specifying the System Type	21
9.11	Sharing Defaults	21
9.12	Defining Variables	22
9.13	'cmake' Invocation	22
10	Linking with TraDemGen	26
10.1	Table of Contents	26
10.2	Introduction	26
10.3	Using the pkg-config command	26
10.4	Using the trademgen-config script	27

10.5 M4 macro for the GNU Autotools	27
10.6 Using TraDemGen with dynamic linking	27
11 Test Rules	27
11.1 The Test Source Files	28
11.2 The Reference File	28
11.3 Testing TraDemGen Library	28
12 Users Guide	28
12.1 Table of Contents	28
12.2 Introduction	29
12.3 Get Started	29
12.3.1 Get the TraDemGen library	29
12.3.2 Build the TraDemGen project	29
12.3.3 Build and Run the Tests	29
12.3.4 Install the TraDemGen Project (Binaries, Documentation)	29
12.4 Exploring the Predefined BOM Tree	29
12.4.1 Demand Stream Engine BOM Tree	29
12.5 Extending the BOM Tree	29
13 Supported Systems	29
13.1 Table of Contents	29
13.2 Introduction	30
13.3 TraDemGen 3.10.x	30
13.3.1 Linux Systems	30
13.3.2 Windows Systems	34
13.3.3 Unix Systems	37
14 TraDemGen Supported Systems (Previous Releases)	38
14.1 TraDemGen 3.9.1	38
14.2 TraDemGen 3.9.0	38
14.3 TraDemGen 3.8.1	38
15 Tutorials	38
15.1 Table of Contents	38
15.2 Introduction	38
15.2.1 Preparing the StdAir Project for Development	38
15.3 Build a Predefined BOM Tree	38
15.3.1 Instantiate the BOM Root Object	39

15.3.2	Instantiate the (Airline) Inventory Object	39
15.3.3	Link the Inventory Object with the BOM Root	39
15.3.4	Build Another Airline Inventory	39
15.3.5	Dump The BOM Tree Content	39
15.3.6	Result of the Tutorial Program	40
15.4	Extend the Pre-Defined BOM Tree	40
15.4.1	Extend an Airline Inventory Object	40
15.4.2	Build the Specific BOM Objects	40
15.4.3	Result of the Tutorial Program	41
16	Command-Line Test to Demonstrate How To Use TraDemGen elements	41
17	Directory Hierarchy	46
17.1	Directories	46
18	Namespace Index	47
18.1	Namespace List	47
19	Class Index	47
19.1	Class Hierarchy	47
20	Class Index	50
20.1	Class List	50
21	File Index	53
21.1	File List	53
22	Directory Documentation	55
22.1	trademgen/basic/ Directory Reference	55
22.2	trademgen/batches/ Directory Reference	55
22.3	trademgen/bom/ Directory Reference	56
22.4	trademgen/ui/cmdline/ Directory Reference	56
22.5	trademgen/command/ Directory Reference	56
22.6	trademgen/factory/ Directory Reference	56
22.7	trademgen/python/ Directory Reference	56
22.8	trademgen/service/ Directory Reference	57
22.9	test/ Directory Reference	57
22.10	trademgen/ Directory Reference	57
22.11	test/trademgen/ Directory Reference	57

22.12trademgen/ui/ Directory Reference	57
23 Namespace Documentation	58
23.1 SEVMGR Namespace Reference	58
23.1.1 Function Documentation	58
23.2 stdair Namespace Reference	59
23.2.1 Detailed Description	59
23.3 TRADEMGEN Namespace Reference	59
23.3.1 Typedef Documentation	62
23.3.2 Function Documentation	67
23.3.3 Variable Documentation	67
23.4 TRADEMGEN::DemandParserHelper Namespace Reference	68
23.4.1 Function Documentation	69
23.4.2 Variable Documentation	72
24 Class Documentation	72
24.1 BomAbstract Class Reference	72
24.2 TRADEMGEN::BomDisplay Class Reference	73
24.2.1 Detailed Description	73
24.2.2 Member Function Documentation	73
24.3 stdair::CategoricalAttribute< T > Struct Template Reference	74
24.3.1 Detailed Description	74
24.3.2 Member Typedef Documentation	74
24.3.3 Constructor & Destructor Documentation	75
24.3.4 Member Function Documentation	75
24.4 TRADEMGEN::CategoricalAttributeLite< T > Struct Template Reference	76
24.4.1 Detailed Description	77
24.4.2 Member Typedef Documentation	77
24.4.3 Constructor & Destructor Documentation	77
24.4.4 Member Function Documentation	78
24.5 CmdAbstract Class Reference	78
24.6 TRADEMGEN::ContinuousAttribute< T > Struct Template Reference	79
24.6.1 Detailed Description	79
24.6.2 Member Typedef Documentation	79
24.6.3 Constructor & Destructor Documentation	80
24.6.4 Member Function Documentation	80
24.7 TRADEMGEN::ContinuousAttributeLite< T > Struct Template Reference	81

24.7.1 Detailed Description	81
24.7.2 Member Typedef Documentation	82
24.7.3 Constructor & Destructor Documentation	82
24.7.4 Member Function Documentation	82
24.8 TRADEMGEN::DBManager Class Reference	83
24.8.1 Detailed Description	83
24.8.2 Member Function Documentation	84
24.9 TRADEMGEN::DBParams Struct Reference	84
24.9.1 Detailed Description	85
24.9.2 Constructor & Destructor Documentation	85
24.9.3 Member Function Documentation	85
24.10 TRADEMGEN::DefaultMap Struct Reference	88
24.10.1 Detailed Description	88
24.10.2 Member Function Documentation	88
24.11 TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT > Struct Template Reference	88
24.11.1 Detailed Description	89
24.11.2 Constructor & Destructor Documentation	90
24.11.3 Member Function Documentation	91
24.11.4 Member Data Documentation	91
24.12 TRADEMGEN::DemandCharacteristics Struct Reference	100
24.12.1 Detailed Description	101
24.12.2 Constructor & Destructor Documentation	101
24.12.3 Member Function Documentation	102
24.12.4 Member Data Documentation	102
24.13 TRADEMGEN::DemandDistribution Struct Reference	105
24.13.1 Detailed Description	105
24.13.2 Constructor & Destructor Documentation	105
24.13.3 Member Function Documentation	106
24.13.4 Member Data Documentation	106
24.14 TRADEMGEN::DemandFileParser Class Reference	107
24.14.1 Detailed Description	107
24.14.2 Constructor & Destructor Documentation	107
24.14.3 Member Function Documentation	108
24.15 TRADEMGEN::DemandFilePath Class Reference	108
24.15.1 Detailed Description	108
24.15.2 Constructor & Destructor Documentation	108

24.16DemandGenerationTestSuite Class Reference	108
24.16.1 Detailed Description	109
24.16.2 Constructor & Destructor Documentation	109
24.16.3 Member Function Documentation	109
24.16.4 Member Data Documentation	109
24.17TRADEMGEN::DemandInputFileNotFoundException Class Reference	109
24.17.1 Detailed Description	110
24.17.2 Constructor & Destructor Documentation	110
24.18TRADEMGEN::DemandManager Class Reference	110
24.18.1 Detailed Description	110
24.18.2 Friends And Related Function Documentation	111
24.19TRADEMGEN::DemandParser Class Reference	111
24.19.1 Detailed Description	111
24.19.2 Member Function Documentation	112
24.20TRADEMGEN::DemandParserHelper::DemandParser Struct Reference	112
24.20.1 Detailed Description	113
24.20.2 Constructor & Destructor Documentation	113
24.20.3 Member Data Documentation	114
24.21TRADEMGEN::DemandStream Class Reference	114
24.21.1 Detailed Description	117
24.21.2 Member Typedef Documentation	117
24.21.3 Constructor & Destructor Documentation	117
24.21.4 Member Function Documentation	117
24.21.5 Friends And Related Function Documentation	127
24.21.6 Member Data Documentation	128
24.22TRADEMGEN::DemandStreamKey Struct Reference	130
24.22.1 Detailed Description	130
24.22.2 Constructor & Destructor Documentation	130
24.22.3 Member Function Documentation	131
24.23TRADEMGEN::DemandStruct Struct Reference	132
24.23.1 Detailed Description	133
24.23.2 Constructor & Destructor Documentation	133
24.23.3 Member Function Documentation	134
24.23.4 Member Data Documentation	134
24.24TRADEMGEN::DictionaryManager Class Reference	140
24.24.1 Detailed Description	140

24.24.2 Member Function Documentation	140
24.25TRADEMGEN::DemandParserHelper::doEndDemand Struct Reference	141
24.25.1 Detailed Description	142
24.25.2 Constructor & Destructor Documentation	142
24.25.3 Member Function Documentation	142
24.25.4 Member Data Documentation	142
24.26FacServiceAbstract Class Reference	143
24.27TRADEMGEN::FacTRADEMGENServiceContext Class Reference	144
24.27.1 Detailed Description	144
24.27.2 Constructor & Destructor Documentation	144
24.27.3 Member Function Documentation	145
24.28FileNotFoundException Class Reference	145
24.29TRADEMGEN::FlagSaver Struct Reference	146
24.29.1 Detailed Description	146
24.29.2 Constructor & Destructor Documentation	146
24.30grammar Class Reference	146
24.31TRADEMGEN::IndexOutOfRangeException Class Reference	147
24.31.1 Detailed Description	147
24.31.2 Constructor & Destructor Documentation	147
24.32InputFilePath Class Reference	147
24.33KeyAbstract Class Reference	148
24.34TRADEMGEN::DemandParserHelper::ParserSemanticAction Struct Reference	148
24.34.1 Detailed Description	149
24.34.2 Constructor & Destructor Documentation	149
24.34.3 Member Data Documentation	150
24.35TRADEMGEN::RandomGenerationContext Struct Reference	150
24.35.1 Detailed Description	151
24.35.2 Constructor & Destructor Documentation	151
24.35.3 Member Function Documentation	151
24.36RootException Class Reference	153
24.37ServiceAbstract Class Reference	153
24.38TRADEMGEN::DemandParserHelper::storeChannelCode Struct Reference	153
24.38.1 Detailed Description	154
24.38.2 Constructor & Destructor Documentation	154
24.38.3 Member Function Documentation	154
24.38.4 Member Data Documentation	154

24.39TRADEMGEN::DemandParserHelper::storeChannelProbMass Struct Reference	155
24.39.1 Detailed Description	155
24.39.2 Constructor & Destructor Documentation	156
24.39.3 Member Function Documentation	156
24.39.4 Member Data Documentation	156
24.40TRADEMGEN::DemandParserHelper::storeDemandChangeFeeDisutility Struct Reference	157
24.40.1 Detailed Description	157
24.40.2 Constructor & Destructor Documentation	157
24.40.3 Member Function Documentation	157
24.40.4 Member Data Documentation	158
24.41TRADEMGEN::DemandParserHelper::storeDemandChangeFeeProb Struct Reference . .	158
24.41.1 Detailed Description	159
24.41.2 Constructor & Destructor Documentation	159
24.41.3 Member Function Documentation	159
24.41.4 Member Data Documentation	159
24.42TRADEMGEN::DemandParserHelper::storeDemandMean Struct Reference	160
24.42.1 Detailed Description	160
24.42.2 Constructor & Destructor Documentation	161
24.42.3 Member Function Documentation	161
24.42.4 Member Data Documentation	161
24.43TRADEMGEN::DemandParserHelper::storeDemandNonRefundableDisutility Struct Ref- erence	162
24.43.1 Detailed Description	162
24.43.2 Constructor & Destructor Documentation	162
24.43.3 Member Function Documentation	162
24.43.4 Member Data Documentation	163
24.44TRADEMGEN::DemandParserHelper::storeDemandNonRefundableProb Struct Reference	163
24.44.1 Detailed Description	164
24.44.2 Constructor & Destructor Documentation	164
24.44.3 Member Function Documentation	164
24.44.4 Member Data Documentation	164
24.45TRADEMGEN::DemandParserHelper::storeDemandStdDev Struct Reference	165
24.45.1 Detailed Description	165
24.45.2 Constructor & Destructor Documentation	166
24.45.3 Member Function Documentation	166
24.45.4 Member Data Documentation	166
24.46TRADEMGEN::DemandParserHelper::storeDestination Struct Reference	167

24.46.1 Detailed Description	167
24.46.2 Constructor & Destructor Documentation	167
24.46.3 Member Function Documentation	167
24.46.4 Member Data Documentation	168
24.47TRADEMGEN::DemandParserHelper::storeDow Struct Reference	168
24.47.1 Detailed Description	169
24.47.2 Constructor & Destructor Documentation	169
24.47.3 Member Function Documentation	169
24.47.4 Member Data Documentation	169
24.48TRADEMGEN::DemandParserHelper::storeDTD Struct Reference	170
24.48.1 Detailed Description	170
24.48.2 Constructor & Destructor Documentation	171
24.48.3 Member Function Documentation	171
24.48.4 Member Data Documentation	171
24.49TRADEMGEN::DemandParserHelper::storeDTDProbMass Struct Reference	172
24.49.1 Detailed Description	172
24.49.2 Constructor & Destructor Documentation	172
24.49.3 Member Function Documentation	172
24.49.4 Member Data Documentation	173
24.50TRADEMGEN::DemandParserHelper::storeFFCode Struct Reference	173
24.50.1 Detailed Description	174
24.50.2 Constructor & Destructor Documentation	174
24.50.3 Member Function Documentation	174
24.50.4 Member Data Documentation	174
24.51TRADEMGEN::DemandParserHelper::storeFFProbMass Struct Reference	175
24.51.1 Detailed Description	175
24.51.2 Constructor & Destructor Documentation	175
24.51.3 Member Function Documentation	176
24.51.4 Member Data Documentation	176
24.52TRADEMGEN::DemandParserHelper::storeOrigin Struct Reference	176
24.52.1 Detailed Description	177
24.52.2 Constructor & Destructor Documentation	177
24.52.3 Member Function Documentation	177
24.52.4 Member Data Documentation	177
24.53TRADEMGEN::DemandParserHelper::storePosCode Struct Reference	178
24.53.1 Detailed Description	179

24.53.2 Constructor & Destructor Documentation	179
24.53.3 Member Function Documentation	179
24.53.4 Member Data Documentation	179
24.54TRADEMGEN::DemandParserHelper::storePosProbMass Struct Reference	180
24.54.1 Detailed Description	180
24.54.2 Constructor & Destructor Documentation	180
24.54.3 Member Function Documentation	181
24.54.4 Member Data Documentation	181
24.55TRADEMGEN::DemandParserHelper::storePrefCabin Struct Reference	181
24.55.1 Detailed Description	182
24.55.2 Constructor & Destructor Documentation	182
24.55.3 Member Function Documentation	182
24.55.4 Member Data Documentation	182
24.56TRADEMGEN::DemandParserHelper::storePrefDepDateRangeEnd Struct Reference	183
24.56.1 Detailed Description	184
24.56.2 Constructor & Destructor Documentation	184
24.56.3 Member Function Documentation	184
24.56.4 Member Data Documentation	184
24.57TRADEMGEN::DemandParserHelper::storePrefDepDateRangeStart Struct Reference	185
24.57.1 Detailed Description	185
24.57.2 Constructor & Destructor Documentation	185
24.57.3 Member Function Documentation	186
24.57.4 Member Data Documentation	186
24.58TRADEMGEN::DemandParserHelper::storePrefDepTime Struct Reference	187
24.58.1 Detailed Description	187
24.58.2 Constructor & Destructor Documentation	187
24.58.3 Member Function Documentation	187
24.58.4 Member Data Documentation	188
24.59TRADEMGEN::DemandParserHelper::storePrefDepTimeProbMass Struct Reference	188
24.59.1 Detailed Description	189
24.59.2 Constructor & Destructor Documentation	189
24.59.3 Member Function Documentation	189
24.59.4 Member Data Documentation	189
24.60TRADEMGEN::DemandParserHelper::storeStayCode Struct Reference	190
24.60.1 Detailed Description	190
24.60.2 Constructor & Destructor Documentation	191

24.60.3 Member Function Documentation	191
24.60.4 Member Data Documentation	191
24.61 TRADEMGEN::DemandParserHelper::storeStayProbMass Struct Reference	192
24.61.1 Detailed Description	192
24.61.2 Constructor & Destructor Documentation	192
24.61.3 Member Function Documentation	192
24.61.4 Member Data Documentation	193
24.62 TRADEMGEN::DemandParserHelper::storeTimeValue Struct Reference	193
24.62.1 Detailed Description	194
24.62.2 Constructor & Destructor Documentation	194
24.62.3 Member Function Documentation	194
24.62.4 Member Data Documentation	194
24.63 TRADEMGEN::DemandParserHelper::storeTimeValueProbMass Struct Reference	195
24.63.1 Detailed Description	195
24.63.2 Constructor & Destructor Documentation	196
24.63.3 Member Function Documentation	196
24.63.4 Member Data Documentation	196
24.64 TRADEMGEN::DemandParserHelper::storeTripCode Struct Reference	197
24.64.1 Detailed Description	197
24.64.2 Constructor & Destructor Documentation	197
24.64.3 Member Function Documentation	197
24.64.4 Member Data Documentation	198
24.65 TRADEMGEN::DemandParserHelper::storeTripProbMass Struct Reference	198
24.65.1 Detailed Description	199
24.65.2 Constructor & Destructor Documentation	199
24.65.3 Member Function Documentation	199
24.65.4 Member Data Documentation	199
24.66 TRADEMGEN::DemandParserHelper::storeWTP Struct Reference	200
24.66.1 Detailed Description	200
24.66.2 Constructor & Destructor Documentation	201
24.66.3 Member Function Documentation	201
24.66.4 Member Data Documentation	201
24.67 StructAbstract Class Reference	202
24.68 TestFixture Class Reference	202
24.69 TRADEMGEN::TRADEMGEN_Abstract Struct Reference	202
24.69.1 Detailed Description	203

24.69.2 Constructor & Destructor Documentation	203
24.69.3 Member Function Documentation	203
24.70TRADEMGEN::TRADEMGEN_Service Class Reference	204
24.70.1 Detailed Description	205
24.70.2 Constructor & Destructor Documentation	205
24.70.3 Member Function Documentation	206
24.71TRADEMGEN::TRADEMGEN_ServiceContext Class Reference	214
24.71.1 Detailed Description	214
24.71.2 Friends And Related Function Documentation	214
24.72TRADEMGEN::Trademgener Struct Reference	215
24.72.1 Detailed Description	215
24.72.2 Constructor & Destructor Documentation	215
24.72.3 Member Function Documentation	216
24.73TRADEMGEN::TrademgenGenerationException Class Reference	216
24.73.1 Detailed Description	217
24.73.2 Constructor & Destructor Documentation	217
25 File Documentation	218
25.1 doc/local/authors.doc File Reference	218
25.2 doc/local/codingrules.doc File Reference	218
25.3 doc/local/copyright.doc File Reference	218
25.4 doc/local/documentation.doc File Reference	218
25.5 doc/local/features.doc File Reference	218
25.6 doc/local/help_wanted.doc File Reference	218
25.7 doc/local/howto_release.doc File Reference	218
25.8 doc/local/index.doc File Reference	218
25.9 doc/local/installation.doc File Reference	218
25.10doc/local/linking.doc File Reference	218
25.11doc/local/test.doc File Reference	218
25.12doc/local/users_guide.doc File Reference	218
25.13doc/local/verification.doc File Reference	218
25.14doc/tutorial/tutorial.doc File Reference	218
25.15test/trademgen/DemandGenerationTestSuite.cpp File Reference	218
25.16DemandGenerationTestSuite.cpp	219
25.17test/trademgen/DemandGenerationTestSuite.hpp File Reference	225
25.17.1 Function Documentation	225
25.18DemandGenerationTestSuite.hpp	226

25.19test/trademgen/generateEvents.cpp File Reference	227
25.19.1 Function Documentation	227
25.20generateEvents.cpp	228
25.21trademgen/basic/BasConst.cpp File Reference	229
25.22BasConst.cpp	230
25.23trademgen/basic/BasConst_DemandGeneration.hpp File Reference	232
25.24BasConst_DemandGeneration.hpp	233
25.25trademgen/basic/BasConst_TRADEMGEN_Service.hpp File Reference	234
25.26BasConst_TRADEMGEN_Service.hpp	235
25.27trademgen/basic/BasParserTypes.hpp File Reference	236
25.28BasParserTypes.hpp	237
25.29trademgen/basic/CategoricalAttribute.hpp File Reference	239
25.30CategoricalAttribute.hpp	240
25.31trademgen/basic/CategoricalAttributeLite.hpp File Reference	243
25.32CategoricalAttributeLite.hpp	244
25.33trademgen/basic/ContinuousAttribute.hpp File Reference	247
25.34ContinuousAttribute.hpp	248
25.35trademgen/basic/ContinuousAttributeLite.hpp File Reference	251
25.36ContinuousAttributeLite.hpp	252
25.37trademgen/basic/DemandCharacteristics.cpp File Reference	255
25.38DemandCharacteristics.cpp	256
25.39trademgen/basic/DemandCharacteristics.hpp File Reference	259
25.40DemandCharacteristics.hpp	260
25.41trademgen/basic/DemandCharacteristicsTypes.hpp File Reference	262
25.42DemandCharacteristicsTypes.hpp	263
25.43trademgen/basic/DemandDistribution.cpp File Reference	265
25.44DemandDistribution.cpp	266
25.45trademgen/basic/DemandDistribution.hpp File Reference	267
25.46DemandDistribution.hpp	268
25.47trademgen/basic/DictionaryManager.cpp File Reference	269
25.48DictionaryManager.cpp	270
25.49trademgen/basic/DictionaryManager.hpp File Reference	271
25.50DictionaryManager.hpp	272
25.51trademgen/basic/RandomGenerationContext.cpp File Reference	273
25.52RandomGenerationContext.cpp	274
25.53trademgen/basic/RandomGenerationContext.hpp File Reference	275

25.54RandomGenerationContext.hpp	276
25.55trademgen/batches/trademgen_generateDemand.cpp File Reference	278
25.55.1 Typedef Documentation	279
25.55.2 Function Documentation	279
25.55.3 Variable Documentation	281
25.56trademgen_generateDemand.cpp	283
25.57trademgen/batches/trademgen_with_db.cpp File Reference	290
25.57.1 Typedef Documentation	291
25.57.2 Function Documentation	291
25.57.3 Variable Documentation	293
25.58trademgen_with_db.cpp	294
25.59trademgen/bom/BomDisplay.cpp File Reference	300
25.60BomDisplay.cpp	301
25.61 trademgen/bom/BomDisplay.hpp File Reference	303
25.62BomDisplay.hpp	304
25.63trademgen/bom/DemandStream.cpp File Reference	305
25.64DemandStream.cpp	306
25.65trademgen/bom/DemandStream.hpp File Reference	316
25.66DemandStream.hpp	317
25.67trademgen/bom/DemandStreamKey.cpp File Reference	322
25.68DemandStreamKey.cpp	323
25.69trademgen/bom/DemandStreamKey.hpp File Reference	324
25.70DemandStreamKey.hpp	325
25.71 trademgen/bom/DemandStreamTypes.hpp File Reference	327
25.72DemandStreamTypes.hpp	328
25.73trademgen/bom/DemandStruct.cpp File Reference	329
25.74DemandStruct.cpp	330
25.75trademgen/bom/DemandStruct.hpp File Reference	333
25.76DemandStruct.hpp	334
25.77trademgen/command/DBManager.cpp File Reference	336
25.78DBManager.cpp	337
25.79trademgen/command/DBManager.hpp File Reference	340
25.80DBManager.hpp	341
25.81 trademgen/command/DemandManager.cpp File Reference	342
25.82DemandManager.cpp	343
25.83trademgen/command/DemandManager.hpp File Reference	358

25.84DemandManager.hpp	359
25.85trademgen/command/DemandParser.cpp File Reference	361
25.86DemandParser.cpp	362
25.87trademgen/command/DemandParser.hpp File Reference	363
25.88DemandParser.hpp	364
25.89trademgen/command/DemandParserHelper.cpp File Reference	365
25.90DemandParserHelper.cpp	366
25.91trademgen/command/DemandParserHelper.hpp File Reference	380
25.92DemandParserHelper.hpp	382
25.93trademgen/DBParams.hpp File Reference	386
25.94DBParams.hpp	387
25.95trademgen/factory/FacTRADEMGENSEerviceContext.cpp File Reference	389
25.96FacTRADEMGENSEerviceContext.cpp	390
25.97trademgen/factory/FacTRADEMGENSEerviceContext.hpp File Reference	391
25.98FacTRADEMGENSEerviceContext.hpp	392
25.99trademgen/python/pytrademgen.cpp File Reference	393
25.99.1 Typedef Documentation	394
25.99.2 Function Documentation	394
25.100pytrademgen.cpp	395
25.101trademgen/service/TRADEMGEN_Service.cpp File Reference	400
25.102TRADEMGEN_Service.cpp	402
25.103trademgen/service/TRADEMGEN_ServiceContext.cpp File Reference	417
25.104TRADEMGEN_ServiceContext.cpp	418
25.105trademgen/service/TRADEMGEN_ServiceContext.hpp File Reference	420
25.106TRADEMGEN_ServiceContext.hpp	421
25.107trademgen/TRADEMGEN_Abstract.hpp File Reference	423
25.107. Function Documentation	423
25.108TRADEMGEN_Abstract.hpp	424
25.109trademgen/TRADEMGEN_Exceptions.hpp File Reference	425
25.110TRADEMGEN_Exceptions.hpp	426
25.111trademgen/TRADEMGEN_Service.hpp File Reference	427
25.112TRADEMGEN_Service.hpp	428
25.113trademgen/TRADEMGEN_Types.hpp File Reference	431
25.114TRADEMGEN_Types.hpp	432
25.115trademgen/ui/cmdline/trademgen.cpp File Reference	433
25.116trademgen.cpp	434

1 TraDemGen Documentation

1.1 Getting Started

- [Main features](#)
- [Installation](#)
- [Linking with TraDemGen](#)
- [Users Guide](#)
- [Tutorials](#)
- [Copyright and License](#)
- [Make a Difference](#)
- [Make a new release](#)
- [People](#)

1.2 TraDemGen at SourceForge

- [Project page](#)
- [Download TraDemGen](#)
- [Open a ticket for a bug or feature](#)
- [Mailing lists](#)
- [Forums](#)
 - [Discuss about Development issues](#)
 - [Ask for Help](#)
 - [Discuss TraDemGen](#)

1.3 TraDemGen Development

- [Git Repository](#) (Subversion is deprecated)
- [Coding Rules](#)
- [Documentation Rules](#)
- [Test Rules](#)

1.4 External Libraries

- [Boost](#) (C++ STL extensions)
- [Python](#)
- [MySQL client](#)
- [SOI](#) (C++ DB API)

1.5 Support TraDemGen

1.6 About TraDemGen

TraDemGen aims at providing a clean API, and the corresponding C++ implementation, able to generate demand for travel solutions (e.g., from JFK to PEK on 25-05-2009) according to characteristics (e.g., Willingness-To-Pay, preferred airline, etc). TraDemGen mainly targets simulation purposes. [N](#)

TraDemGen makes an extensive use of existing open-source libraries for increased functionality, speed and accuracy. In particular the [Boost](#) (C++ *Standard Extensions*) library is used.

The TraDemGen library originates from the department of Operational Research and Innovation at [Amadeus](#), Sophia Antipolis, France. TraDemGen is released under the terms of the [GNU Lesser General Public License](#) (LGPLv2.1) for you to enjoy.

TraDemGen should work on [GNU/Linux](#), [Sun Solaris](#), Microsoft Windows (with [Cygwin](#), [MinGW/MSYS](#), or [Microsoft Visual C++ .NET](#)) and [Mac OS X](#) operating systems.

Note:

(N) - The TraDemGen library is **NOT** intended, in any way, to be used by airlines for production systems. If you want to report issue, bug or feature request, or if you just want to give feedback, have a look on the right-hand side of this page for the preferred reporting methods. In any case, please do not contact Amadeus directly for any matter related to TraDemGen.

2 People

2.1 Project Admins (and Developers)

- Anh Quan Nguyen <quannaus@users.sourceforge.net> ([N](#))
- Denis Arnaud <denis_arnaud@users.sourceforge.net> ([N](#))
- Gabrielle Sabatier <gsabatier@users.sourceforge.net> ([N](#))

2.2 Retired Developers

- Mehdi Ayouni <mehdi.ayouni@gmail.com>
- Son Nguyen Kim <snguyenkim@users.sourceforge.net> ([N](#))

2.3 Contributors

- Emmanuel Bastien <ebastien@users.sourceforge.net> ([N](#))

2.4 Distribution Maintainers

- [Fedora/RedHat](#): Denis Arnaud <denis_arnaud@users.sourceforge.net> ([N](#))
- [Debian](#): Emmanuel Bastien <ebastien@users.sourceforge.net> ([N](#))

Note:

(N) - [Amadeus](#) employees.

3 Coding Rules

In the following sections we describe the naming conventions which are used for files, classes, structures, local variables, and global variables.

3.1 Default Naming Rules for Variables

Variables names follow Java naming conventions. Examples:

- `lNumberOfPassengers`
- `lSeatAvailability`

3.2 Default Naming Rules for Functions

Function names follow Java naming conventions. Example:

- `int myFunctionName (const int& a, int b)`

3.3 Default Naming Rules for Classes and Structures

Each new word in a class or structure name should always start with a capital letter and the words should be separated with an under-score. Abbreviations are written with capital letters. Examples:

- `MyClassName`
- `MyStructName`

3.4 Default Naming Rules for Files

Files are named after the C++ class names.

Source files are named using `.cpp` suffix, whereas header files end with `.hpp` extension. Examples:

- `FlightDate.hpp`
- `SegmentDate.cpp`

3.5 Default Functionality of Classes

All classes that are configured by input parameters should include:

- default empty constructor
- one or more additional constructor(s) that takes input parameters and initializes the class instance
- setup function, preferably named `'setup'` or `'set_parameters'`

Explicit destructor functions are not required, unless they are needed. It shall not be possible to use any of the other member functions unless the class has been properly initiated with the input parameters.

4 Copyright and License

4.1 GNU LESSER GENERAL PUBLIC LICENSE

4.1.1 Version 2.1, February 1999

Copyright (C) 1991, 1999 Free Software Foundation, Inc.
51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA

Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.

[This is the first released version of the Lesser GPL. It also counts
as the successor of the GNU Library Public License, version 2, hence
the version number 2.1.]

4.2 Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public Licenses are intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users.

This license, the Lesser General Public License, applies to some specially designated software packages--typically libraries--of the Free Software Foundation and other authors who decide to use it. You can use it too, but we suggest you first think carefully about whether this license or the ordinary General Public License is the better strategy to use in any particular case, based on the explanations below.

When we speak of free software, we are referring to freedom of use, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish); that you receive source code or can get it if you want it; that you can change the software and use pieces of it in new free programs; and that you are informed that you can do these things.

To protect your rights, we need to make restrictions that forbid distributors to deny you these rights or to ask you to surrender these rights. These restrictions translate to certain responsibilities for you if you distribute copies of the library or if you modify it.

For example, if you distribute copies of the library, whether gratis or for a fee, you must give the recipients all the rights that we gave you. You must make sure that they, too, receive or can get the source code. If you link other code with the library, you must provide complete object files to the recipients, so that they can relink them with the library after making changes to the library and recompiling it. And you must show them these terms so they know their rights.

We protect your rights with a two-step method: (1) we copyright the library, and (2) we offer you this license, which gives you legal permission to copy, distribute and/or modify the library.

To protect each distributor, we want to make it very clear that there is no warranty for the free library. Also, if the library is modified by someone else and passed on, the recipients should know that what they have is not the original version, so that the original author's reputation will not be affected by problems that might be introduced by others.

Finally, software patents pose a constant threat to the existence of any free program. We wish to make sure that a company cannot effectively restrict the users of a free program by obtaining a restrictive license from a patent holder. Therefore, we insist that any patent license obtained for a version of the library must be consistent with the full freedom of use specified in this license.

Most GNU software, including some libraries, is covered by the ordinary GNU General Public License. This license, the GNU Lesser General Public License, applies to certain designated libraries, and is quite different from the ordinary General Public License. We use this license for certain libraries in order to

permit linking those libraries into non-free programs.

When a program is linked with a library, whether statically or using a shared library, the combination of the two is legally speaking a combined work, a derivative of the original library. The ordinary General Public License therefore permits such linking only if the entire combination fits its criteria of freedom. The Lesser General Public License permits more lax criteria for linking other code with the library.

We call this license the "Lesser" General Public License because it does Less to protect the user's freedom than the ordinary General Public License. It also provides other free software developers Less of an advantage over competing non-free programs. These disadvantages are the reason we use the ordinary General Public License for many libraries. However, the Lesser license provides advantages in certain special circumstances.

For example, on rare occasions, there may be a special need to encourage the widest possible use of a certain library, so that it becomes a de-facto standard. To achieve this, non-free programs must be allowed to use the library. A more frequent case is that a free library does the same job as widely used non-free libraries. In this case, there is little to gain by limiting the free library to free software only, so we use the Lesser General Public License.

In other cases, permission to use a particular library in non-free programs enables a greater number of people to use a large body of free software. For example, permission to use the GNU C Library in non-free programs enables many more people to use the whole GNU operating system, as well as its variant, the GNU/Linux operating system.

Although the Lesser General Public License is Less protective of the users' freedom, it does ensure that the user of a program that is linked with the Library has the freedom and the wherewithal to run that program using a modified version of the Library.

The precise terms and conditions for copying, distribution and modification follow. Pay close attention to the difference between a "work based on the library" and a "work that uses the library". The former contains code derived from the library, whereas the latter must be combined with the library in order to run.

4.3 TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License Agreement applies to any software library or other program which contains a notice placed by the copyright holder or other authorized party saying it may be distributed under the terms of this Lesser General Public License (also called "this License"). Each licensee is addressed as "you".

A "library" means a collection of software functions and/or data prepared so as to be conveniently linked with application programs (which use some of those functions and data) to form executables.

The "Library", below, refers to any such software library or work which has been distributed under these terms. A "work based on the Library" means either the Library or any derivative work under copyright law: that is to say, a work containing the Library or a portion of it, either verbatim or with modifications and/or translated straightforwardly into another language. (Hereinafter, translation is included without limitation in the term "modification".)

"Source code" for a work means the preferred form of the work for making modifications to it. For a library, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the library.

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running a program using the Library is not restricted, and output from such a program is covered only if its contents constitute a work based on the Library (independent of the use of the Library in a tool for writing it). Whether that is true depends on what the Library does and what the program that uses the Library does.

1. You may copy and distribute verbatim copies of the Library's complete source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and distribute a copy of this License along with the Library.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Library or any portion of it, thus forming a work based on the Library, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

a) The modified work must itself be a software library.

b) You must cause the files modified to carry prominent notices stating that you changed the files and the date of any change.

c) You must cause the whole of the work to be licensed at no charge to all third parties under the terms of this License.

d) If a facility in the modified Library refers to a function or a table of data to be supplied by an application program that uses the facility, other than as an argument passed when the facility is invoked, then you must make a good faith effort to ensure that, in the event an application does not supply such function or table, the facility still operates, and performs whatever part of its purpose remains meaningful.

(For example, a function in a library to compute square roots has a purpose that is entirely well-defined independent of the application. Therefore, Subsection 2d requires that any application-supplied function or table used by this function must be optional: if the application does not supply it, the square root function must still compute square roots.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Library, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Library, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Library.

In addition, mere aggregation of another work not based on the Library with the Library (or with a work based on the Library) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may opt to apply the terms of the ordinary GNU General Public License instead of this License to a given copy of the Library. To do this, you must alter all the notices that refer to this License, so that they refer to the ordinary GNU General Public License, version 2, instead of to this License. (If a newer version than version 2 of the ordinary GNU General Public License has appeared, then you can specify that version instead if you wish.) Do not make any other change in these notices.

Once this change is made in a given copy, it is irreversible for that copy, so the ordinary GNU General Public License applies to all subsequent copies and derivative works made from that copy.

This option is useful when you wish to copy part of the code of the Library into a program that is not a library.

4. You may copy and distribute the Library (or a portion or derivative of it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange.

If distribution of object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place satisfies the requirement to distribute the source code, even though third parties are not compelled to copy the source along with the object code.

5. A program that contains no derivative of any portion of the Library, but is designed to work with the Library by being compiled or linked with it, is called a "work that uses the Library". Such a work, in isolation, is not a derivative work of the Library, and therefore falls outside the scope of this License.

However, linking a "work that uses the Library" with the Library creates an executable that is a derivative of the Library (because it contains portions of the Library), rather than a "work that uses the library". The executable is therefore covered by this License. Section 6 states terms for distribution of such executables.

When a "work that uses the Library" uses material from a header file that is part of the Library, the object code for the work may be a derivative work of the Library even though the source code is not. Whether this is true is especially significant if the work can be linked without the Library, or if the work is itself a library. The threshold for this to be true is not precisely defined by law.

If such an object file uses only numerical parameters, data structure layouts and accessors, and small macros and small inline functions (ten lines or less in length), then the use of the object file is unrestricted, regardless of whether it is legally a derivative work. (Executables containing this object code plus portions of the Library will still fall under Section 6.)

Otherwise, if the work is a derivative of the Library, you may distribute the object code for the work under the terms of Section 6. Any executables containing that work also fall under Section 6, whether or not they are linked directly with the Library itself.

6. As an exception to the Sections above, you may also combine or link a "work that uses the Library" with the Library to produce a work containing portions of the Library, and distribute that work under terms of your choice, provided that the terms permit modification of the work for the customer's own use and reverse engineering for debugging such modifications.

You must give prominent notice with each copy of the work that the Library is used in it and that the Library and its use are covered by this License. You must supply a copy of this License. If the work during execution displays copyright notices, you must include the copyright notice for the Library among them, as well as a reference directing the user to the copy of this License. Also, you must do one of these things:

- a) Accompany the work with the complete corresponding machine-readable source code for the Library including whatever changes were used in the work (which must be distributed under Sections 1 and 2 above); and, if the work is an executable linked with the Library, with the complete machine-readable "work that uses the Library", as object code and/or source code, so that the user can modify the Library and then relink to produce a modified executable containing the modified Library. (It is understood that the user who changes the contents of definitions files in the Library will not necessarily be able to recompile the application to use the modified definitions.)
- b) Use a suitable shared library mechanism for linking with the Library. A suitable mechanism is one that (1) uses at run time a copy of the library already present on the user's computer system, rather than copying library functions into the executable, and (2) will operate properly with a modified version of the library, if the user installs one, as long as the modified version is interface-compatible with the version that the work was made with.
- c) Accompany the work with a written offer, valid for at least three years, to give the same user the materials specified in Subsection 6a, above, for a charge no more than the cost of performing this distribution.
- d) If distribution of the work is made by offering access to copy from a designated place, offer equivalent access to copy the above specified materials from the same place.
- e) Verify that the user has already received a copy of these materials or that you have already sent this user a copy.

For an executable, the required form of the "work that uses the Library" must include any data and utility

programs needed for reproducing the executable from it. However, as a special exception, the materials to be distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

It may happen that this requirement contradicts the license restrictions of other proprietary libraries that do not normally accompany the operating system. Such a contradiction means you cannot use both them and the Library together in an executable that you distribute.

7. You may place library facilities that are a work based on the Library side-by-side in a single library together with other library facilities not covered by this License, and distribute such a combined library, provided that the separate distribution of the work based on the Library and of the other library facilities is otherwise permitted, and provided that you do these two things:

a) Accompany the combined library with a copy of the same work based on the Library, uncombined with any other library facilities. This must be distributed under the terms of the Sections above.

b) Give prominent notice with the combined library of the fact that part of it is a work based on the Library, and explaining where to find the accompanying uncombined form of the same work.

8. You may not copy, modify, sublicense, link with, or distribute the Library except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense, link with, or distribute the Library is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

9. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Library or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Library (or any work based on the Library), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Library or works based on it.

10. Each time you redistribute the Library (or any work based on the Library), the recipient automatically receives a license from the original licensor to copy, distribute, link with or modify the Library subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties with this License.

11. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Library at all. For example, if a patent license would not permit royalty-free redistribution of the Library by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Library.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply, and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

12. If the distribution and/or use of the Library is restricted in certain countries either by patents or by

copyrighted interfaces, the original copyright holder who places the Library under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

13. The Free Software Foundation may publish revised and/or new versions of the Lesser General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Library specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Library does not specify a license version number, you may choose any version ever published by the Free Software Foundation.

14. If you wish to incorporate parts of the Library into other free programs whose distribution conditions are incompatible with these, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

4.3.1 NO WARRANTY

15. BECAUSE THE LIBRARY IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE LIBRARY, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE LIBRARY "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE LIBRARY IS WITH YOU. SHOULD THE LIBRARY PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

16. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE LIBRARY AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE LIBRARY (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE LIBRARY TO OPERATE WITH ANY OTHER SOFTWARE), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

4.3.2 END OF TERMS AND CONDITIONS

4.4 How to Apply These Terms to Your New Programs

If you develop a new library, and you want it to be of the greatest possible use to the public, we recommend making it free software that everyone can redistribute and change. You can do so by permitting redistribution under these terms (or, alternatively, under the terms of the ordinary General Public License).

To apply these terms, attach the following notices to the library. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

<one line to give the library's name and a brief idea of what it does.>

```
Copyright (C) <year> <name of author>
```

```
This library is free software; you can redistribute it and/or
modify it under the terms of the GNU Lesser General Public
License as published by the Free Software Foundation; either
version 2.1 of the License, or (at your option) any later version.
```

```
This library is distributed in the hope that it will be useful,
but WITHOUT ANY WARRANTY; without even the implied warranty of
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU
Lesser General Public License for more details.
```

```
You should have received a copy of the GNU Lesser General Public
License along with this library; if not, write to the Free Software
Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA
```

Also add information on how to contact you by electronic and paper mail.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the library, if necessary. Here is a sample; alter the names:

```
Yoyodyne, Inc., hereby disclaims all copyright interest in the
library 'Frob' (a library for tweaking knobs) written by James Random Hacker.
```

```
<signature of Ty Coon>, 1 April 1990
Ty Coon, President of Vice
```

That's all there is to it!

[Source](#)

5 Documentation Rules

5.1 General Rules

All classes in TraDemGen should be properly documented with Doxygen comments in include (.hpp) files. Source (.cpp) files should be documented according to a normal standard for well documented C++ code.

An example of how the interface of a class shall be documented in TraDemGen is shown here:

```
/*!
 * \brief Brief description of MyClass here
 *
 * Detailed description of MyClass here. With example code if needed.
 */
class MyClass {
public:
    //! Default constructor
    MyClass(void) { setup_done = false; }

    /*!
     * \brief Constructor that initializes the class with parameters
     *
     * Detailed description of the constructor here if needed
     *
     * \param[in] param1 Description of \a param1 here
     * \param[in] param2 Description of \a param2 here
     */
    MyClass(TYPE1 param1, TYPE2 param2) { setup(param1, param2); }
```

```

/*!
 * \brief Setup function for MyClass
 *
 * Detailed description of the setup function here if needed
 *
 * \param[in] param1 Description of \a param1 here
 * \param[in] param2 Description of \a param2 here
 */
void setup(TYPE1 param1, TYPE2 param2);

/*!
 * \brief Brief description of memberFunction1
 *
 * Detailed description of memberFunction1 here if needed
 *
 * \param[in] param1 Description of \a param1 here
 * \param[in] param2 Description of \a param2 here
 * \param[in,out] param3 Description of \a param3 here
 * \return Description of the return value here
 */
TYPE4 memberFunction1(TYPE1 param1, TYPE2 param2, TYPE3 &param3);

private:

    bool _setupDone;          /*!< Variable that checks if the class is properly
                               initialized with parameters */
    TYPE1 _privateVariable1; /*!< Short description of _privateVariable1 here
    TYPE2 _privateVariable2; /*!< Short description of _privateVariable2 here
};

```

5.2 File Header

All files should start with the following header, which include Doxygen's \file, \brief and \author tags, \$Date\$ and \$Revisions\$ CVS tags, and a common copyright note:

```

/*!
 * \file
 * \brief Brief description of the file here
 * \author Names of the authors who contributed to this code
 * \date Date
 *
 * Detailed description of the file here if needed.
 *
 * -----
 *
 * TraDemGen - C++ Simulated Revenue Accounting (RAC) System Library
 *
 * Copyright (C) 2009-2011 (\see authors file for a list of contributors)
 *
 * \see copyright file for license information
 *
 * -----
 */

```

5.3 Grouping Various Parts

All functions must be added to a Doxygen group in order to appear in the documentation. The following code example defines the group 'my_group':

```

/*!
 * \defgroup my_group Brief description of the group here
 *

```

```
* Detailed description of the group here
*/
```

The following example shows how to document the function `myFunction` and how to add it to the group `my_group`:

```
/*!
 * \brief Brief description of myFunction here
 * \ingroup my_group
 *
 * Detailed description of myFunction here
 *
 * \param[in] param1 Description of \a param1 here
 * \param[in] param2 Description of \a param2 here
 * \return Description of the return value here
 */
TYPE3 myFunction(TYPE1 param1, TYPE2 &param2);
```

6 Main features

A short list of the main features of TraDemEn is given below sorted in different categories. Many more features and functions exist and for these we refer to the reference documentation.

6.1 Demand generation

The demand can be generated thanks to two relatively advanced pieces of algorithm, both following a sequential principle. That is, the **events** (booking requests) are generated one after the other, sequentially, rather than being generated all at once at the beginning of the process (e.g., a simulation).

The two sequential methods are:

- 'Intuitive' method. The booking period is sliced in **intervals**, where the arrival rate of events (booking requests) is known for each of those intervals, say λ_i . The inter-arrival process then follows an **exponential law**. That is, the final number of booking requests follows a **Non homogeneous Poisson distribution**. With that method, the **variance** of that **distribution** is therefore equal to the **mean**.
- 'Advanced' method. The process uses **order statistics** in order to mimic the behaviour of **uniform distributions** projected onto the known arrival pattern of events. With that method, the final number of booking requests is first drawn, following any probability distribution (e.g., **normal**, **Gamma**, **Beta** or even Weibull law) with any required standard deviation. Then, each booking request is drawn in sequence:
 - according to a mere **uniform distribution**,
 - and projected onto the known booking arrival pattern.

6.2 Other features

- CSV input file parsing
- Memory handling

7 Make a Difference

Do not ask what TraDemGen can do for you. Ask what you can do for TraDemGen.

You can help us to develop the TraDemGen library. There are always a lot of things you can do:

- Start using TraDemGen
- Tell your friends about TraDemGen and help them to get started using it
- If you find a bug, report it to us. Without your help we can never hope to produce a bug free code.
- Help us to improve the documentation by providing information about documentation bugs
- Answer support requests in the TraDemGen discussion forums on SourceForge. If you know the answer to a question, help others to overcome their TraDemGen problems.
- Help us to improve our algorithms. If you know of a better way (e.g. that is faster or requires less memory) to implement some of our algorithms, then let us know.
- Help us to port TraDemGen to new platforms. If you manage to compile TraDemGen on a new platform, then tell us how you did it.
- Send us your code. If you have a good TraDemGen compatible code, which you can release under the LGPL, and you think it should be included in TraDemGen, then send it to us.
- Become an TraDemGen developer. Send us an e-mail and tell what you can do for TraDemGen.

8 Make a new release

8.1 Introduction

This document describes briefly the recommended procedure of releasing a new version of TraDemGen using a Linux development machine and the SourceForge project site.

The following steps are required to make a release of the distribution package.

8.2 Initialisation

Clone locally the full [Git project](#):

```
cd ~
mkdir -p dev/sim
cd ~/dev/sim
git clone git://trademgen.git.sourceforge.net/gitroot/trademgen/trademgen trademgengit
cd trademgengit
git checkout trunk
```

8.3 Release branch maintenance

Switch to the release branch, on your local clone, and merge the latest updates from the trunk. Decide about the new version to be released.

```
cd ~/dev/sim/trademgengit
git checkout releases
git merge trunk
```

Update the version in the various build system files, replacing the old version numbers by the correct ones:

```
vi CMakeLists.txt
vi autogen.sh
vi README
```

Update the version, add some news in the NEWS file, add a change-log in the ChangeLog file and in the RPM specification files:

```
vi NEWS
vi ChangeLog
vi trademgen.spec
```

8.4 Commit and publish the release branch

Commit the new release:

```
cd ~/dev/sim/trademgengit
git add -A
git commit -m "[Release 0.5.0] Release of the 0.5.0 version of TraDemGen."
git push
```

8.5 Create distribution packages

Create the distribution packages using the following command:

```
cd ~/dev/sim/trademgengit
git checkout releases
rm -rf build && mkdir -p build
cd build
export INSTALL_BASEDIR=/home/user/dev/deliveries
export LIBSUFFIX_4_CMAKE="-DLIB_SUFFIX=64"
cmake -DCMAKE_INSTALL_PREFIX=${INSTALL_BASEDIR}/trademgen-0.5.0 \
  -DWITH_STDAIR_PREFIX=${INSTALL_BASEDIR}/stdair-stable \
  -DWITH_AIRAC_PREFIX=${INSTALL_BASEDIR}/airsched-stable \
  -DWITH_AIRAC_PREFIX=${INSTALL_BASEDIR}/airrac-stable \
  -DWITH_RMOL_PREFIX=${INSTALL_BASEDIR}/rmol-stable \
  -DWITH_RMOL_PREFIX=${INSTALL_BASEDIR}/airinv-stable \
  -DWITH_RMOL_PREFIX=${INSTALL_BASEDIR}/simfqt-stable \
  -DCMAKE_BUILD_TYPE:STRING=Debug -DINSTALL_DOC:BOOL=ON \
  ${LIBSUFFIX_4_CMAKE} ..
make check && make dist
make install
```

This will configure, compile and check the package. The output packages will be named, for instance, trademgen-0.5.0.tar.gz and trademgen-0.5.0.tar.bz2.

8.6 Upload the HTML documentation to SourceForge

In order to update the Web site files, either:

- **synchronise them with rsync and SSH:** Upload the just generated HTML (and PDF) documentation onto the **SourceForge Web site**.

```
cd ~/dev/sim/trademgengit/build
git checkout releases
rsync -aiv ${INSTALL_BASEDIR}/trademgen-0.5.0/share/doc/trademgen-0.5.0/html/ \
  your_sf_user,trademgen@web.sourceforge.net:htdocs/
```

where `-aiv` options mean:

- `-a`: archive/mirror mode; equals `-rlptgoD` (no `-H`, `-A`, `-X`)
- `-v`: increase verbosity
- `-i`: output a change-summary for all updates
- Note the trailing slashes (/) at the end of both the source and target directories. It means that the content of the source directory (`doc/html`), rather than the directory itself, has to be copied into the content of the target directory.

- or use the [SourceForge Shell service](#).

8.7 Generate the RPM packages

Optionally, generate the RPM package (for instance, for [Fedora/RedHat](#)):

```
cd ~/dev/sim/trademgengit/build
git checkout releases
make dist
```

To perform this step, `rpm-build`, `rpmlint` and `rpmdevtools` have to be available on the system.

```
cp ../trademgen.spec ~/dev/packages/SPECS \
  && cp trademgen-0.5.0.tar.bz2 ~/dev/packages/SOURCES
cd ~/dev/packages/SPECS
rpmbuild -ba trademgen.spec
cd ~/dev/packages
rpmlint -i SPECS/trademgen.spec SRPMS/trademgen-0.5.0-1.fc16.src.rpm \
  RPMS/noarch/trademgen-* RPMS/i686/trademgen-*
```

8.8 Update distributed change log

Update the `NEWS` and `ChangeLog` files with appropriate information, including what has changed since the previous release. Then commit and push the changes into the [TraDemGen's Git repository](#).

8.9 Create the binary package, including the documentation

Create the binary package, which includes HTML and PDF documentation, using the following command:

```
cd ~/dev/sim/trademgengit/build
git checkout releases
make package
```

The output binary package will be named, for instance, `trademgen-0.5.0-Linux.tar.bz2`. That package contains both the HTML and PDF documentation. The binary package contains also the executables and shared libraries, as well as C++ header files, but all of those do not interest us for now.

8.10 Upload the files to SourceForge

Upload the distribution and documentation packages to the SourceForge server. Check [SourceForge help page on uploading software](#).

8.11 Make a new post

- submit a new entry in the [SourceForge project-related news feed](#)
- make a new post on the [SourceForge hosted WordPress blog](#)
- and update, if necessary, [Trac tickets](#).

8.12 Send an email on the announcement mailing-list

Finally, you should send an announcement to trademgen-announce@lists.sourceforge.net (see <https://lists.sourceforge.net/lists/listinfo/trademgen-announce> for the archives)

9 Installation

9.1 Table of Contents

- [Fedora/RedHat Linux distributions](#)
- [TraDemGen Requirements](#)
- [Basic Installation](#)
- [Compilers and Options](#)
- [Compiling For Multiple Architectures](#)
- [Installation Names](#)
- [Optional Features](#)
- [Particular systems](#)
- [Specifying the System Type](#)
- [Sharing Defaults](#)
- [Defining Variables](#)
- [‘cmake’ Invocation](#)

9.2 Fedora/RedHat Linux distributions

Note that on [Fedora/RedHat](#) Linux distributions, RPM packages are available and can be installed with your usual package manager. For instance:

```
yum -y install trademgen-devel trademgen-doc
```

RPM packages can also be available on the [SourceForge download site](#).

9.3 TraDemGen Requirements

TraDemGen should compile without errors or warnings on most GNU/Linux systems, on UNIX systems like Solaris SunOS, and on POSIX based environments for Microsoft Windows like Cygwin or MinGW with MSYS. It can be also built on Microsoft Windows NT/2000/XP/Vista/7 using Microsoft's Visual C++ .NET, but our support for this compiler is limited. For GNU/Linux, SunOS, Cygwin and MinGW we assume that you have at least the following GNU software installed on your computer:

- GNU Autotools:
 - `autoconf`,
 - `automake`,
 - `libtool`,
 - `make`, version 3.72.1 or later (check version with `'make --version'`)
- `GCC` - GNU C++ Compiler (g++), version 4.3.x or later (check version with `'gcc --version'`)
- `Boost` - C++ STL extensions, version 1.35 or later (check version with `'grep "define BOOST_LIB_VERSION" /usr/include/boost/version.hpp'`)
- `MySQL` - Database client libraries, version 5.0 or later (check version with `'mysql --version'`)
- `SOCHI` - C++ database client library wrapper, version 3.0.0 or later (check version with `'soci-config --version'`)

Optionally, you might need a few additional programs: `Doxygen`, `LaTeX`, `Dvips` and `Ghostscript`, to generate the HTML and PDF documentation.

We strongly recommend that you use recent stable releases of the GCC, if possible. We do not actively work on supporting older versions of the GCC, and they may therefore (without prior notice) become unsupported in future releases of TraDemGen.

9.4 Basic Installation

Briefly, the shell commands `./cmake .. && make install` should configure, build and install this package. The following more-detailed instructions are generic; see the `'README'` file for instructions specific to this package. Some packages provide this `'INSTALL'` file but do not implement all of the features documented below. The lack of an optional feature in a given package is not necessarily a bug. More recommendations for GNU packages can be found in the info page corresponding to "Makefile Conventions: (standards)Makefile Conventions".

The `'cmake'` shell script attempts to guess correct values for various system-dependent variables used during compilation. It uses those values to create a `'Makefile'` in each directory of the package. It may also create one or more `'h'` files containing system-dependent definitions. Finally, it creates a `'CMakeCache.txt'` cache file that you can refer to in the future to recreate the current configuration, and files `'CMakeFiles'` containing compiler output (useful mainly for debugging `'cmake'`).

It can also use an optional file (typically called `'config.cache'` and enabled with `'--cache-file=config.cache'` or simply `'-C'`) that saves the results of its tests to speed up reconfiguring. Caching is disabled by default to prevent problems with accidental use of stale cache files.

If you need to do unusual things to compile the package, please try to figure out how `'configure'` could check whether to do them, and mail diffs or instructions to the address given in the `'README'` so they can be considered for the next release. If you are using the cache, and at some point `'config.cache'` contains results you don't want to keep, you may remove or edit it.

The file `CMakeLists.txt` is used to create the `Makefile` files.

The simplest way to compile this package is:

1. `cd` to the directory containing the package's source code and type `./cmake .` to configure the package for your system. Running `cmake` is generally fast. While running, it prints some messages telling which features it is checking for.
2. Type `make` to compile the package.
3. Optionally, type `make check` to run any self-tests that come with the package, generally using the just-built uninstalled binaries.
4. Type `make install` to install the programs and any data files and documentation. When installing into a prefix owned by root, it is recommended that the package be configured and built as a regular user, and only the `make install` phase executed with root privileges.
5. You can remove the program binaries and object files from the source code directory by typing `make clean`. To also remove the files that `configure` created (so you can compile the package for a different kind of computer), type `make distclean`. There is also a `make maintainer-clean` target, but that is intended mainly for the package's developers. If you use it, you may have to get all sorts of other programs in order to regenerate files that came with the distribution.
6. Often, you can also type `make uninstall` to remove the installed files again. In practice, not all packages have tested that uninstallation works correctly, even though it is required by the GNU Coding Standards.

9.5 Compilers and Options

Some systems require unusual options for compilation or linking that the `cmake` script does not know about. Run `./cmake --help` for details on some of the pertinent environment variables.

You can give `cmake` initial values for configuration parameters by setting variables in the command line or in the environment. Here is an example:

```
./cmake CC=c99 CFLAGS=-g LIBS=-lposix
```

See also:

[Defining Variables](#) for more details.

9.6 Compiling For Multiple Architectures

You can compile the package for more than one kind of computer at the same time, by placing the object files for each architecture in their own directory. To do this, you can use GNU `make`. `cd` to the directory where you want the object files and executables to go and

run the 'configure' script. 'configure' automatically checks for the source code in the directory that 'configure' is in and in '..'. This is known as a "VPATH" build.

With a non-GNU 'make', it is safer to compile the package for one architecture at a time in the source code directory. After you have installed the package for one architecture, use 'make distclean' before reconfiguring for another architecture.

On MacOS X 10.5 and later systems, you can create libraries and executables that work on multiple system types--known as "fat" or "universal" binaries--by specifying multiple '-arch' options to the compiler but only a single '-arch' option to the preprocessor. Like this:

```
./configure CC="gcc -arch i386 -arch x86_64 -arch ppc -arch ppc64" \  
           CXX="g++ -arch i386 -arch x86_64 -arch ppc -arch ppc64" \  
           CPP="gcc -E" CXXCPP="g++ -E"
```

This is not guaranteed to produce working output in all cases, you may have to build one architecture at a time and combine the results using the 'lipo' tool if you have problems.

9.7 Installation Names

By default, 'make install' installs the package's commands under '/usr/local/bin', include files under '/usr/local/include', etc. You can specify an installation prefix other than '/usr/local' by giving 'configure' the option '--prefix=PREFIX', where PREFIX must be an absolute file name.

You can specify separate installation prefixes for architecture-specific files and architecture-independent files. If you pass the option '--exec-prefix=PREFIX' to 'configure', the package uses PREFIX as the prefix for installing programs and libraries. Documentation and other data files still use the regular prefix.

In addition, if you use an unusual directory layout you can give options like '--bindir=DIR' to specify different values for particular kinds of files. Run 'configure --help' for a list of the directories you can set and what kinds of files go in them. In general, the default for these options is expressed in terms of '\${prefix}', so that specifying just '--prefix' will affect all of the other directory specifications that were not explicitly provided.

The most portable way to affect installation locations is to pass the correct locations to 'configure'; however, many packages provide one or both of the following shortcuts of passing variable assignments to the 'make install' command line to change installation locations without having to reconfigure or recompile.

The first method involves providing an override variable for each affected directory. For example, 'make install prefix=/alternate/directory' will choose an alternate location for all directory configuration variables that were expressed in terms of '\${prefix}'. Any directories that were specified during 'configure',

but not in terms of `'${prefix}'`, must each be overridden at install time for the entire installation to be relocated. The approach of makefile variable overrides for each directory variable is required by the GNU Coding Standards, and ideally causes no recompilation. However, some platforms have known limitations with the semantics of shared libraries that end up requiring recompilation when using this method, particularly noticeable in packages that use GNU Libtool.

The second method involves providing the `'DESTDIR'` variable. For example, `'make install DESTDIR=/alternate/directory'` will prepend `'/alternate/directory'` before all installation names. The approach of `'DESTDIR'` overrides is not required by the GNU Coding Standards, and does not work on platforms that have drive letters. On the other hand, it does better at avoiding recompilation issues, and works well even when some directory options were not specified in terms of `'${prefix}'` at `'configure'` time.

9.8 Optional Features

If the package supports it, you can cause programs to be installed with an extra prefix or suffix on their names by giving `'cmake'` the option `'--program-prefix=PREFIX'` or `'--program-suffix=SUFFIX'`.

Some packages pay attention to `'--enable-FEATURE'` options to `'configure'`, where `FEATURE` indicates an optional part of the package. They may also pay attention to `'--with-PACKAGE'` options, where `PACKAGE` is something like `'gnu-as'` or `'x'` (for the X Window System). The `'README'` should mention any `'--enable-'` and `'--with-'` options that the package recognizes.

For packages that use the X Window System, `'configure'` can usually find the X include and library files automatically, but if it doesn't, you can use the `'configure'` options `'--x-includes=DIR'` and `'--x-libraries=DIR'` to specify their locations.

Some packages offer the ability to configure how verbose the execution of `'make'` will be. For these packages, running `'./configure --enable-silent-rules'` sets the default to minimal output, which can be overridden with `'make V=1'`; while running `'./configure --disable-silent-rules'` sets the default to verbose, which can be overridden with `'make V=0'`.

9.9 Particular systems

On HP-UX, the default C compiler is not ANSI C compatible. If GNU CC is not installed, it is recommended to use the following options in order to use an ANSI C compiler:

```
./configure CC="cc -Ae -D_XOPEN_SOURCE=500"
```

and if that doesn't work, install pre-built binaries of GCC for HP-UX.

On OSF/1 a.k.a. Tru64, some versions of the default C compiler cannot parse its `'<wchar.h>'` header file. The option `'-nodtk'` can be used as

a workaround. If GNU CC is not installed, it is therefore recommended to try

```
./configure CC="cc"
```

and if that doesn't work, try

```
./configure CC="cc -nodtk"
```

On Solaris, don't put `/usr/ucb` early in your `PATH`. This directory contains several dysfunctional programs; working variants of these programs are available in `/usr/bin`. So, if you need `/usr/ucb` in your `PATH`, put it `_after_` `/usr/bin`.

On Haiku, software installed for all users goes in `/boot/common`, not `/usr/local`. It is recommended to use the following options:

```
./cmake -DCMAKE_INSTALL_PREFIX=/boot/common
```

9.10 Specifying the System Type

There may be some features `'configure'` cannot figure out automatically, but needs to determine by the type of machine the package will run on. Usually, assuming the package is built to be run on the `_same_` architectures, `'configure'` can figure that out, but if it prints a message saying it cannot guess the machine type, give it the `'--build=TYPE'` option. TYPE can either be a short name for the system type, such as `'sun4'`, or a canonical name which has the form CPU-COMPANY-SYSTEM

where SYSTEM can have one of these forms:

- OS
- KERNEL-OS

See the file `'config.sub'` for the possible values of each field. If `'config.sub'` isn't included in this package, then this package doesn't need to know the machine type.

If you are `_building_` compiler tools for cross-compiling, you should use the option `'--target=TYPE'` to select the type of system they will produce code for.

If you want to `_use_` a cross compiler, that generates code for a platform different from the build platform, you should specify the `"host"` platform (i.e., that on which the generated programs will eventually be run) with `'--host=TYPE'`.

9.11 Sharing Defaults

If you want to set default values for `'configure'` scripts to share, you can create a site shell script called `'config.site'` that gives

default values for variables like `'CC'`, `'cache_file'`, and `'prefix'`. `'configure'` looks for `'PREFIX/share/config.site'` if it exists, then `'PREFIX/etc/config.site'` if it exists. Or, you can set the `'CONFIG_SITE'` environment variable to the location of the site script. A warning: not all `'configure'` scripts look for a site script.

9.12 Defining Variables

Variables not defined in a site shell script can be set in the environment passed to `'configure'`. However, some packages may run `'configure'` again during the build, and the customized values of these variables may be lost. In order to avoid this problem, you should set them in the `'configure'` command line, using `'VAR=value'`. For example:

```
./configure CC=/usr/local2/bin/gcc
```

causes the specified `'gcc'` to be used as the C compiler (unless it is overridden in the site shell script).

Unfortunately, this technique does not work for `'CONFIG_SHELL'` due to an Autoconf bug. Until the bug is fixed you can use this workaround:

```
CONFIG_SHELL=/bin/bash /bin/bash ./configure CONFIG_SHELL=/bin/bash
```

9.13 'cmake' Invocation

`'cmake'` recognizes the following options to control how it operates.

- `'--help'`, `'-h'` print a summary of all of the options to `'configure'`, and exit.
- `'--help=short'`, `'--help=recursive'` print a summary of the options unique to this package's `'configure'`, and exit. The `'short'` variant lists options used only in the top level, while the `'recursive'` variant lists options also present in any nested packages.
- `'--version'`, `'-V'` print the version of Autoconf used to generate the `'configure'` script, and exit.
- `'--cache-file=FILE'` enable the cache: use and save the results of the tests in FILE, traditionally `'config.cache'`. FILE defaults to `'/dev/null'` to disable caching.
- `'--config-cache'`, `'-C'` alias for `'--cache-file=config.cache'`.
- `'--quiet'`, `'--silent'`, `'-q'` do not print messages saying which checks are being made. To suppress all normal output, redirect it to `'/dev/null'` (any error messages will still be shown).
- `'--srcdir=DIR'` look for the package's source code in directory DIR. Usually `'configure'` can determine that directory automatically.
- `'--prefix=DIR'` use DIR as the installation prefix.

See also:

[Installation Names](#) for more details, including other options available for fine-tuning the installation locations.

- '--no-create', '-n' run the configure checks, but stop before creating any output files.

'cmake' also accepts some other, not widely useful, options. Run 'cmake --help' for more details.

The 'cmake' script produces an output like this:

```
cmake -DCMAKE_INSTALL_PREFIX=/home/user/dev/deliveries/trademgen-99.99.99 -DLIB_SUFFIX=64 -DCMAKE_BUILD_TYPE=Debug
-- The C compiler identification is GNU
-- The CXX compiler identification is GNU
-- Check for working C compiler: /usr/lib64/ccache/gcc
-- Check for working C compiler: /usr/lib64/ccache/gcc -- works
-- Detecting C compiler ABI info
-- Detecting C compiler ABI info - done
-- Check for working CXX compiler: /usr/lib64/ccache/c++
-- Check for working CXX compiler: /usr/lib64/ccache/c++ -- works
-- Detecting CXX compiler ABI info
-- Detecting CXX compiler ABI info - done
-- Requires Git without specifying any version
-- Current Git revision name: 4856624ea4978b3b2bfe26cb6702cce0be531084 trunk
-- Requires PythonLibs-2.6
-- Found PythonLibs: /usr/lib64/libpython2.7.so (Required is at least version "2.6")
-- Found PythonLibs 2.7
-- Requires Boost-1.41
-- Boost version: 1.46.0
-- Found the following Boost libraries:
--   program_options
--   date_time
--   iostreams
--   serialization
--   filesystem
--   unit_test_framework
--   python
-- Found Boost version: 1.46.0
-- Found BoostWrapper: /usr/include (Required is at least version "1.41")
-- Requires MySQL without specifying any version
-- Using mysql-config: /usr/bin/mysql_config
-- Found MySQL: /usr/lib64/mysql/libmysqlclient.so
-- Found MySQL version: 5.5.14
-- Requires SOCI-3.0
-- Using soci-config: /usr/bin/soci-config
-- SOCI headers are buried
-- Found SOCI: /usr/lib64/libsoci_core.so (Required is at least version "3.0")
-- Found SOCIMySQL: /usr/lib64/libsoci_mysql.so (Required is at least version "3.0")
-- Found SOCI with MySQL back-end support version: 3.0.0
-- Requires StdAir-0.35
-- Found StdAir version: 0.36.2
-- Requires Doxygen without specifying any version
-- Found Doxygen: /usr/bin/doxygen
-- Found DoxygenWrapper: /usr/bin/doxygen
-- Found Doxygen version: 1.7.4
-- Had to set the linker language for 'trademgenlib' to CXX
-- Had to set the linker language for 'pytrademgenlib' to CXX
-- Test 'TrademgenTest' to be built with 'DemandGenerationTestSuite.cpp'
--
-- =====
-- -----
-- ---      Project Information      ---
-- -----
-- PROJECT_NAME ..... : trademgen
```



```

-- PACKAGE_PRETTY_NAME ..... : TraDemGen
-- PACKAGE ..... : trademgen
-- PACKAGE_NAME ..... : TRADEMGEN
-- PACKAGE_BRIEF ..... : C++ Simulated Travel Demand Generation Library
-- PACKAGE_VERSION ..... : 99.99.99
-- GENERIC_LIB_VERSION ..... : 99.99.99
-- GENERIC_LIB_SOVERSION ..... : 99.99
--
-- -----
-- ---      Build Configuration      ---
-- -----
-- Modules to build ..... : trademgen
-- Libraries to build/install ..... : trademgenlib;pytrademgenlib
-- Binaries to build/install ..... : trademgen;trademgen_with_db;pytrademgen.py
-- Modules to test ..... : trademgen
-- Binaries to test ..... : TrademgenTesttst
--
-- * Module ..... : trademgen
--   + Layers to build ..... : .;basic;bom;factory;command;service
--   + Dependencies on other layers :
--   + Libraries to build/install . : trademgenlib;pytrademgenlib
--   + Executables to build/install : trademgen;trademgen_with_db;pytrademgen.py
--   + Tests to perform ..... : TrademgenTesttst
--
-- BUILD_SHARED_LIBS ..... : ON
-- CMAKE_BUILD_TYPE ..... : Debug
-- * CMAKE_C_FLAGS ..... :
-- * CMAKE_CXX_FLAGS ..... : -Wall -Werror
-- * BUILD_FLAGS ..... :
-- * COMPILE_FLAGS ..... :
-- CMAKE_MODULE_PATH ..... : /home/user/dev/sim/trademgen/trademgengithub/config/
-- CMAKE_INSTALL_PREFIX ..... : /home/user/dev/deliveries/trademgen-99.99.99
--
-- * Doxygen:
--   - DOXYGEN_VERSION ..... : 1.7.4
--   - DOXYGEN_EXECUTABLE ..... : /usr/bin/doxygen
--   - DOXYGEN_DOT_EXECUTABLE ..... : /usr/bin/dot
--   - DOXYGEN_DOT_PATH ..... : /usr/bin
--
-- -----
-- ---      Installation Configuration      ---
-- -----
-- INSTALL_LIB_DIR ..... : /home/user/dev/deliveries/trademgen-99.99.99/lib64
-- INSTALL_BIN_DIR ..... : /home/user/dev/deliveries/trademgen-99.99.99/bin
-- INSTALL_INCLUDE_DIR ..... : /home/user/dev/deliveries/trademgen-99.99.99/include
-- INSTALL_DATA_DIR ..... : /home/user/dev/deliveries/trademgen-99.99.99/share
-- INSTALL_SAMPLE_DIR ..... : /home/user/dev/deliveries/trademgen-99.99.99/share/trademgen/samples
-- INSTALL_DOC ..... : ON
--
-- -----
-- ---      Packaging Configuration      ---
-- -----
-- CPACK_PACKAGE_CONTACT ..... : Denis Arnaud <denis_arnaud - at - users dot sourceforge dot net>
-- CPACK_PACKAGE_VENDOR ..... : Denis Arnaud
-- CPACK_PACKAGE_VERSION ..... : 99.99.99
-- CPACK_PACKAGE_DESCRIPTION_FILE . : /home/user/dev/sim/trademgen/trademgengithub/README
-- CPACK_RESOURCE_FILE_LICENSE .... : /home/user/dev/sim/trademgen/trademgengithub/COPYING
-- CPACK_GENERATOR ..... : TBZ2
-- CPACK_DEBIAN_PACKAGE_DEPENDS ... :
-- CPACK_SOURCE_GENERATOR ..... : TBZ2;TGZ
-- CPACK_SOURCE_PACKAGE_FILE_NAME . : trademgen-99.99.99
--
-- -----
-- ---      External libraries      ---
-- -----
--
-- * Python:

```

```

-- - PYTHONLIBS_VERSION ..... : 2.7
-- - PYTHON_LIBRARIES ..... : /usr/lib64/libpython2.7.so
-- - PYTHON_INCLUDE_PATH ..... : /usr/include/python2.7
-- - PYTHON_INCLUDE_DIRS ..... : /usr/include/python2.7
-- - PYTHON_DEBUG_LIBRARIES ..... :
-- - Python_ADDITIONAL_VERSIONS . :
--
-- * Boost:
-- - Boost_VERSION ..... : 104600
-- - Boost_LIB_VERSION ..... : 1_46
-- - Boost_HUMAN_VERSION ..... : 1.46.0
-- - Boost_INCLUDE_DIRS ..... : /usr/include
-- - Boost required components .. : program_options;date_time;iostreams;serialization;filesystem;unit_te
-- - Boost required libraries ... : optimized;/usr/lib64/libboost_iostreams-mt.so;debug;/usr/lib64/libbo
--
-- * MySQL:
-- - MYSQL_VERSION ..... : 5.5.14
-- - MYSQL_INCLUDE_DIR ..... : /usr/include/mysql
-- - MYSQL_LIBRARIES ..... : /usr/lib64/mysql/libmysqlclient.so
--
-- * SOCI:
-- - SOCI_VERSION ..... : 3.0.0
-- - SOCI_INCLUDE_DIR ..... : /usr/include/soci
-- - SOCI_MYSQL_INCLUDE_DIR ..... : /usr/include/soci
-- - SOCI_LIBRARIES ..... : /usr/lib64/libsoci_core.so
-- - SOCI_MYSQL_LIBRARIES ..... : /usr/lib64/libsoci_mysql.so
--
-- * StdAir:
-- - STDAIR_VERSION ..... : 0.36.2
-- - STDAIR_BINARY_DIRS ..... : /home/user/dev/deliveries/stdair-0.36.2/bin
-- - STDAIR_EXECUTABLES ..... : stdair
-- - STDAIR_LIBRARY_DIRS ..... : /home/user/dev/deliveries/stdair-0.36.2/lib64
-- - STDAIR_LIBRARIES ..... : stdairlib;stdairuiclib
-- - STDAIR_INCLUDE_DIRS ..... : /home/user/dev/deliveries/stdair-0.36.2/include
-- - STDAIR_SAMPLE_DIR ..... : /home/user/dev/deliveries/stdair-0.36.2/share/stdair/samples
--
-- Change a value with: cmake -D<Variable>=<Value>
-- =====
--
-- Configuring done
-- Generating done
-- Build files have been written to: /home/user/dev/sim/trademgen/trademgengithub/build

```

It is recommended that you check if your library has been compiled and linked properly and works as expected. To do so, you should execute the testing process 'make check'. As a result, you should obtain a similar report:

```

[ 0%] Built target hdr_cfg_trademgen
[ 94%] Built target trademgenlib
[100%] Built target TrademgenTesttst
Scanning dependencies of target check_trademgentst
Test project /home/user/dev/sim/trademgen/trademgengithub/build/test/trademgen
  Start 1: TrademgenTesttst
1/1 Test #1: TrademgenTesttst ..... Passed    0.37 sec

100% tests passed, 0 tests failed out of 1

Total Test time (real) = 10.82 sec
[100%] Built target check_trademgentst
Scanning dependencies of target check
[100%] Built target check

```

Check if all the executed tests PASSED. If not, please contact us by

filling a `bug-report`.

Finally, you should install the compiled and linked library, include files and (optionally) HTML and PDF documentation by typing:

```
make install
```

Depending on the `PREFIX` settings during configuration, you might need the root (administrator) access to perform this step.

Eventually, you might invoke the following command

```
make clean
```

to remove all files created during compilation process, or even

```
cd ~/dev/sim/trademgengit
rm -rf build && mkdir build
cd build
```

to remove everything.

10 Linking with TraDemGen

10.1 Table of Contents

- [Introduction](#)
- [Using the `pkg-config` command](#)
- [Using the `trademgen-config` script](#)
- [M4 macro for the GNU Autotools](#)
- [Using TraDemGen with dynamic linking](#)

10.2 Introduction

There are two convenient methods of linking your programs with the TraDemGen library. The first one employs the `'pkg-config'` command (see <http://pkgconfig.freedesktop.org/>), whereas the second one uses `'trademgen-config'` script. These methods are shortly described below.

10.3 Using the `pkg-config` command

`'pkg-config'` is a helper tool used when compiling applications and libraries. It helps you insert the correct compiler and linker options. The syntax of the `'pkg-config'` is as follows:

```
pkg-config <options> <library_name>
```

For instance, assuming that you need to compile an TraDemGen based program `'my_prog.cpp'`, you should use the following command:

```
g++ `pkg-config --cflags trademgen` -o my_prog my_prog.cpp `pkg-config --libs tra
demgen`
```

For more information see the `'pkg-config'` man pages.

10.4 Using the trademgen-config script

TraDemGen provides a shell script called `'trademgen-config'`, which is installed by default in `'$prefix/bin'` (`'/usr/local/bin'`) directory. It can be used to simplify compilation and linking of TraDemGen based programs. The usage of this script is quite similar to the usage of the `'pkg-config'` command.

Assuming that you need to compile the program `'my_prog.cpp'` you can now do that with the following command:

```
g++ 'trademgen-config --cflags' -o my_prog_opt my_prog.cpp 'trademgen-config --li
bs'
```

A list of `'trademgen-config'` options can be obtained by typing:

```
trademgen-config --help
```

If the `'trademgen-config'` command is not found by your shell, you should add its location `'$prefix/bin'` to the `PATH` environment variable, e.g.:

```
export PATH=/usr/local/bin:$PATH
```

10.5 M4 macro for the GNU Autotools

A M4 macro file is delivered with TraDemGen, namely `'trademgen.m4'`, which can be found in, e.g., `'/usr/share/aclocal'`. When used by a `'configure'` script, thanks to the `'AM_PATH_TraDemGen'` macro (specified in the M4 macro file), the following Makefile variables are then defined:

- `'TraDemGen_VERSION'` (e.g., defined to 0.23.0)
- `'TraDemGen_CFLAGS'` (e.g., defined to `'-I${prefix}/include'`)
- `'TraDemGen_LIBS'` (e.g., defined to `'-L${prefix}/lib -ltrademgen'`)

10.6 Using TraDemGen with dynamic linking

When using static linking some of the library routines in TraDemGen are copied into your executable program. This can lead to unnecessary large executables. To avoid having too large executable files you may use dynamic linking instead. Dynamic linking means that the actual linking is performed when the program is executed. This requires that the system is able to locate the shared TraDemGen library file during your program execution. If you install the TraDemGen library using a non-standard prefix, the `'LD_LIBRARY_PATH'` environment variable might be used to inform the linker of the dynamic library location, e.g.:

```
export LD_LIBRARY_PATH=<TraDemGen installation prefix>/lib:$LD_LIBRARY_PATH
```

11 Test Rules

This section describes how the functionality of the TraDemGen library should be verified. In the `'test/trademgen'` subdirectory, test source files are provided. All functionality should be tested using these test source files.

11.1 The Test Source Files

Each new TraDemGen module/class should be accompanied with a test source file. The test source file is an implementation in C++ that tests the functionality of a function/class or a group of functions/classes called test suites. The test source file should test relevant parameter settings and input/output relations to guarantee correct functionality of the corresponding classes/functions. The test source files should be maintained using version control and updated whenever new functionality is added to the TraDemGen library.

The test source file should print relevant data to a standard output that can be used to verify the functionality. All relevant parameter settings should be tested.

The test source file should be placed in the `'test/trademgen'` subdirectory and should have a name ending with `'TestSuite.cpp'`.

11.2 The Reference File

Consider a test source file named `'YieldTestSuite.cpp'`. A reference file named `'YieldTestSuite.ref'` should accompany the test source file. The reference file contains a reference printout of the standard output generated when running the test program. The reference file should be maintained using version control and updated according to the test source file.

11.3 Testing TraDemGen Library

One can compile and execute all test programs from the `'test/trademgen'` sub-directory by typing:

```
% make check
```

after successful compilation of the TraDemGen library.

12 Users Guide

12.1 Table of Contents

- [Introduction](#)
- [Get Started](#)
 - [Get the TraDemGen library](#)
 - [Build the TraDemGen project](#)
 - [Build and Run the Tests](#)
 - [Install the TraDemGen Project \(Binaries, Documentation\)](#)
- [Exploring the Predefined BOM Tree](#)
 - [Demand Stream Engine BOM Tree](#)
- [Extending the BOM Tree](#)

12.2 Introduction

The `TraDemGen` library contains classes for yield rule management. This document does not cover all the aspects of the `TraDemGen` library. It does however explain the most important things you need to know in order to start using `TraDemGen`.

12.3 Get Started

12.3.1 Get the `TraDemGen` library

12.3.2 Build the `TraDemGen` project

To run the configuration script the first time, go to the top directory (where the `TraDemGen` package has been un-packed), and issue either of the following two commands, depending on whether the `TraDemGen` project has been checked out from the Subversion repository or downloaded as a tar-ball package from the Sourceforge Web site:

- `./autogen.sh`
- `./configure`

12.3.3 Build and Run the Tests

12.3.4 Install the `TraDemGen` Project (Binaries, Documentation)

12.4 Exploring the Predefined BOM Tree

`TraDemGen` predefines a BOM (Business Object Model) tree specific to the airline IT arena.

12.4.1 Demand Stream Engine BOM Tree

- `TRADEMGEN::DemandStream`

12.5 Extending the BOM Tree

13 Supported Systems

13.1 Table of Contents

- [Introduction](#)
- [TraDemGen 3.10.x](#)
 - [Linux Systems](#)
 - * [Fedora Core 4 with ATLAS](#)
 - * [Gentoo Linux with ACML](#)
 - * [Gentoo Linux with ATLAS](#)
 - * [Gentoo Linux with MKL](#)
 - * [Gentoo Linux with NetLib's BLAS and LAPACK](#)

- * Red Hat Enterprise Linux with TraDemGen External
- * SUSE Linux 10.0 with NetLib's BLAS and LAPACK
- * SUSE Linux 10.0 with MKL
- Windows Systems
 - * Microsoft Windows XP with Cygwin
 - * Microsoft Windows XP with Cygwin and ATLAS
 - * Microsoft Windows XP with Cygwin and ACML
 - * Microsoft Windows XP with MinGW, MSYS and ACML
 - * Microsoft Windows XP with MinGW, MSYS and TraDemGen External
 - * Microsoft Windows XP with MS Visual C++ and Intel MKL
- Unix Systems
 - * SunOS 5.9 with TraDemGen External
- TraDemGen 3.9.1
- TraDemGen 3.9.0
- TraDemGen 3.8.1

13.2 Introduction

This page is intended to provide a list of TraDemGen supported systems, i.e. the systems on which configuration, installation and testing process of the TraDemGen library has been successful. Results are grouped based on minor release number. Therefore, only the latest tests for bug-fix releases are included. Besides, the information on this page is divided into sections dependent on the operating system.

Where necessary, some extra information is given for each tested configuration, e.g. external libraries installed, configuration commands used, etc.

If you manage to compile, install and test the TraDemGen library on a system not mentioned below, please let us know, so we could update this database.

13.3 TraDemGen 3.10.x

13.3.1 Linux Systems

13.3.1.1 Fedora Core 4 with ATLAS

- **Platform:** Intel Pentium 4
- **Operating System:** Fedora Core 4 (x86)
- **Compiler:** g++ (GCC) 4.0.2 20051125
- **TraDemGen release:** 3.10.0
- **External Libraries:** From FC4 distribution:
 - `fftw3.i386-3.0.1-3`
 - `fftw3-devel.i386-3.0.1-3`
 - `atlas-sse2.i386-3.6.0-8.fc4`
 - `atlas-sse2-devel.i386-3.6.0-8.fc4`

```
- blas.i386-3.0-35.fc4  
- lapack.i386-3.0-35.fc4
```

- **Tests Status:** All tests PASSED
- **Comments:** TraDemGen configured with:

```
% CXXFLAGS="-O3 -pipe -march=pentium4" ./configure
```

- **Date:** March 7, 2006
- **Tester:** Tony Ottosson

13.3.1.2 Gentoo Linux with ACML

- **Platform:** AMD Sempron 3000+
- **Operating System:** Gentoo Linux 2006.0 (x86 arch)
- **Compiler(s):** g++ (GCC) 3.4.5
- **TraDemGen release:** 3.10.1
- **External Libraries:** Compiled and installed from portage tree:

```
- sci-libs/acml-3.0.0
```

- **Tests Status:** All tests PASSED
- **Comments:** BLAS and LAPACK libs set by using the following system commands:

```
% eselect blas set ACML  
% eselect lapack set ACML
```

TraDemGen configured with:

```
% export CPPFLAGS="-I/usr/include/acml"  
% ./configure --with-blas="-lblas"
```

- **Date:** March 31, 2006
- **Tester:** Adam Piatyszek (ediap)

13.3.1.3 Gentoo Linux with ATLAS

- **Platform:** Intel Pentium M Centrino
- **Operating System:** Gentoo Linux 2006.0 (x86)
- **Compiler:** g++ (GCC) 3.4.5
- **TraDemGen release:** 3.10.1
- **External Libraries:** Compiled and installed from portage tree:

```
- sci-libs/fftw-3.1  
- sci-libs/blas-atlas-3.6.0-r1  
- sci-libs/lapack-atlas-3.6.0
```


- **Tests Status:** All tests PASSED
- **Comments:** BLAS and LAPACK libs set by using the following system commands:

```
% eselect blas set ATLAS
% eselect lapack set ATLAS
```

TraDemGen configured with:

```
% ./configure --with-blas="-lblas"
```

- **Date:** March 31, 2006
- **Tester:** Adam Piatyszek (ediap)

13.3.1.4 Gentoo Linux with MKL

- **Platform:** Intel Pentium M Centrino
- **Operating System:** Gentoo Linux 2006.0 (x86 arch)
- **Compiler:** g++ (GCC) 3.4.5
- **TraDemGen release:** 3.10.0
- **External Libraries:** Intel Math Kernel Library (MKL) 8.0.1 installed manually in the following directory: /opt/intel/mkl/8.0.1
- **Tests Status:** All tests PASSED
- **Comments:** TraDemGen configured using the following commands:

```
% export LDFLAGS="-L/opt/intel/mkl/8.0.1/lib/32"
% export CPPFLAGS="-I/opt/intel/mkl/8.0.1/include"
% ./configure
```

- **Date:** February 28, 2006
- **Tester:** Adam Piatyszek (ediap)

13.3.1.5 Gentoo Linux with NetLib's BLAS and LAPACK

- **Platform:** Intel Pentium M Centrino
- **Operating System:** Gentoo Linux 2006.0 (x86)
- **Compiler:** g++ (GCC) 3.4.5
- **TraDemGen release:** 3.10.1
- **External Libraries:** Compiled and installed from portage tree:

```
- sci-libs/fftw-3.1
- sci-libs/blas-reference-19940131-r2
- sci-libs/cblas-reference-20030223
- sci-libs/lapack-reference-3.0-r2
```

- **Tests Status:** All tests PASSED
- **Comments:** BLAS and LAPACK libs set by using the following system commands:

```
% blas-config reference
% lapack-config reference
```

TraDemGen configured with:

```
% ./configure --with-blas="-lblas"
```

- **Date:** March 31, 2006
- **Tester:** Adam Piatyszek (ediap)

13.3.1.6 Red Hat Enterprise Linux with TraDemGen External

- **Platform:** Intel Pentium 4
- **Operating System:** Red Hat Enterprise Linux AS release 4 (Nahant Update 2)
- **Compiler:** g++ (GCC) 3.4.4 20050721 (Red Hat 3.4.4-2)
- **TraDemGen release:** 3.10.0
- **External Libraries:** BLAS, CBLAS, LAPACK and FFTW libraries from TraDemGen External 2.1.1 package
- **Tests Status:** All tests PASSED
- **Date:** March 7, 2006
- **Tester:** Erik G. Larsson

13.3.1.7 SUSE Linux 10.0 with NetLib's BLAS and LAPACK

- **Platform:** Intel Pentium 4 CPU 3.20GHz (64-bit)
- **Operating System:** SUSE Linux 10.0 (x86_64)
- **Compiler(s):** g++ (GCC) 4.0.2
- **TraDemGen release:** 3.10.0
- **External Libraries:** BLAS, LAPACK and FFTW libraries installed from OpenSuse 10.0 RPM repository:

```
- blas-3.0-926
- lapack-3.0-926
- fftw3-3.0.1-114
- fftw3-threads-3.0.1-114
- fftw3-devel-3.0.1-114
```

- **Tests Status:** All tests PASSED
- **Comments:** TraDemGen configured with:


```
% export CXXFLAGS="-m64 -march=nocona -O3 -pipe"
% ./configure --with-lapack="/usr/lib64/liblapack.so.3"
```
- **Date:** March 1, 2006
- **Tester:** Adam Piatyszek (ediap)

13.3.1.8 SUSE Linux 10.0 with MKL

- **Platform:** Intel Pentium 4 CPU 3.20GHz (64-bit)
- **Operating System:** SUSE Linux 10.0 (x86_64)
- **Compiler(s):** g++ (GCC) 4.0.2
- **TraDemGen release:** 3.10.0
- **External Libraries:** Intel Math Kernel Library (MKL) 8.0.1 installed manually in the following directory: /opt/intel/mkl/8.0.1
- **Tests Status:** All tests PASSED
- **Comments:** TraDemGen configured with:

```
% export CXXFLAGS="-m64 -march=nocona -O3 -pipe"  
% export LDFLAGS="-L/opt/intel/mkl/8.0.1/lib/em64t"  
% export CPPFLAGS="-I/opt/intel/mkl/8.0.1/include"  
% ./configure
```

- **Date:** March 1, 2006
- **Tester:** Adam Piatyszek (ediap)

13.3.2 Windows Systems

13.3.2.1 Microsoft Windows XP with Cygwin

- **Platform:** AMD Sempron 3000+
- **Operating System:** Microsoft Windows XP SP2, Cygwin 1.5.19-4
- **Compiler(s):** g++ (GCC) 3.4.4 (cygming special)
- **TraDemGen release:** 3.10.1
- **External Libraries:** Installed from Cygwin's repository:
 - fftw-3.0.1-2
 - fftw-dev-3.0.1-1
 - lapack-3.0-4
- **Tests Status:** All tests PASSED
- **Comments:** Only static library can be built. TraDemGen configured with:

```
% ./configure
```

- **Date:** March 31, 2006
- **Tester:** Adam Piatyszek (ediap)

13.3.2.2 Microsoft Windows XP with Cygwin and ATLAS

- **Platform:** AMD Sempron 3000+
- **Operating System:** Microsoft Windows XP SP2, Cygwin 1.5.19-4
- **Compiler(s):** g++ (GCC) 3.4.4 (cygming special)
- **TraDemGen release:** 3.10.1
- **External Libraries:** Installed from Cygwin's repository:

```
- fftw-3.0.1-2
- fftw-dev-3.0.1-1
```

ATLAS BLAS and LAPACK libraries from TraDemGen External 2.1.1 package configured using:

```
% ./configure --enable-atlas --disable-fftw
```

- **Tests Status:** All tests PASSED
- **Comments:** Only static library can be built. TraDemGen configured with:

```
% export LDFLAGS="-L/usr/local/lib"
% ./configure
```

- **Date:** March 31, 2006
- **Tester:** Adam Piatyszek (ediap)

13.3.2.3 Microsoft Windows XP with Cygwin and ACML

- **Platform:** AMD Sempron 3000+
- **Operating System:** Microsoft Windows XP SP2, Cygwin 1.5.19-4
- **Compiler(s):** g++ (GCC) 3.4.4 (cygming special)
- **TraDemGen release:** 3.10.2
- **External Libraries:** ACML version 3.1.0 (acml3.1.0-32-win32-g77.exe) installed into a default directory, i.e. "c:\Program Files\AMD\acml3.1.0"
- **Tests Status:** All tests PASSED
- **Comments:** Only static library can be built. TraDemGen configured with:

```
% export LDFLAGS="-L/cygdrive/c/Progra~1/AMD/acml3.1.0/gnu32/lib"
% export CPPFLAGS="-I/cygdrive/c/Progra~1/AMD/acml3.1.0/gnu32/include"
% ./configure --enable-debug
```

- **Date:** May 15, 2006
- **Tester:** Adam Piatyszek (ediap)

13.3.2.4 Microsoft Windows XP with MinGW, MSYS and ACML

- **Platform:** AMD Sempron 3000+
- **Operating System:** Microsoft Windows XP SP2, MinGW 5.0.2, MSYS 1.0.10
- **Compiler(s):** g++ (GCC) 3.4.4 (mingw special)
- **TraDemGen release:** 3.10.2
- **External Libraries:** ACML version 3.1.0 (acml3.1.0-32-win32-g77.exe) installed into a default directory, i.e. "c:\Program Files\AMD\acml3.1.0"
- **Tests Status:** All tests PASSED
- **Comments:** Only static library can be built. TraDemGen configured with:

```
% export LDFLAGS="-L/c/Progra~1/AMD/acml3.1.0/gnu32/lib"
% export CPPFLAGS="-I/c/Progra~1/AMD/acml3.1.0/gnu32/include"
% ./configure --enable-debug
```

- **Date:** May 15, 2006
- **Tester:** Adam Piatyszek (ediap)

13.3.2.5 Microsoft Windows XP with MinGW, MSYS and TraDemGen External

- **Platform:** AMD Sempron 3000+
- **Operating System:** Microsoft Windows XP SP2, MinGW 5.0.2, MSYS 1.0.10
- **Compiler(s):** g++ (GCC) 3.4.4 (mingw special)
- **TraDemGen release:** 3.10.5
- **External Libraries:** BLAS, CBLAS, LAPACK and FFTW libraries from TraDemGen External 2.2.0 package
- **Tests Status:** All tests PASSED
- **Comments:** Only static library can be built. TraDemGen configured with:

```
% export LDFLAGS="-L/usr/local/lib"
% export CPPFLAGS="-I/usr/local/include"
% export CXXFLAGS="-Wall -O3 -march=athlon-tbird -pipe"
% ./configure --disable-html-doc
```

- **Date:** August 11, 2006
- **Tester:** Adam Piatyszek (ediap)

13.3.2.6 Microsoft Windows XP with MS Visual C++ and Intel MKL

- **Platform:** AMD Sempron 3000+
- **Operating System:** Microsoft Windows XP SP2
- **Compiler(s):** Microsoft Visual C++ 2005 .NET
- **TraDemGen release:** 3.10.5
- **External Libraries:** Intel Math Kernel Library (MKL) 8.1 installed manually in the following directory: "C:\Program Files\Intel\MKL\8.1"
- **Tests Status:** Not fully tested. Some TraDemGen based programs compiled and run with success.
- **Comments:** Only static library can be built. TraDemGen built by opening the "win32\trademgen.vcproj" project file in MSVC++ and executing "Build -> Build Solution" command from menu.
- **Date:** August 11, 2006
- **Tester:** Adam Piatyszek (ediap)

13.3.3 Unix Systems

13.3.3.1 SunOS 5.9 with TraDemGen External

- **Platform:** SUNW, Sun-Blade-100 (SPARC)
- **Operating System:** SunOS 5.9 Generic_112233-10
- **Compiler(s):** g++ (GCC) 3.4.5
- **TraDemGen release:** 3.10.2
- **External Libraries:** BLAS, CBLAS, LAPACK and FFTW libraries from TraDemGen External 2.1.1 package. The following configuration command has been used:

```
% export CFLAGS="-mcpu=ultrasparc -O2 -pipe -funroll-all-loops"  
% ./configure
```

- **Tests Status:** All tests PASSED
- **Comments:** TraDemGen configured with:

```
% export LDFLAGS="-L/usr/local/lib"  
% export CPPFLAGS="-I/usr/local/include"  
% export CXXFLAGS="-mcpu=ultrasparc -O2 -pipe"  
% ./configure --enable-debug
```

- **Date:** May 15, 2006
- **Tester:** Adam Piatyszek (ediap)

14 TraDemGen Supported Systems (Previous Releases)

14.1 TraDemGen 3.9.1

14.2 TraDemGen 3.9.0

14.3 TraDemGen 3.8.1

15 Tutorials

15.1 Table of Contents

- [Introduction](#)
 - [Preparing the StdAir Project for Development](#)
- [Build a Predefined BOM Tree](#)
 - [Instantiate the BOM Root Object](#)
 - [Instantiate the \(Airline\) Inventory Object](#)
 - [Link the Inventory Object with the BOM Root](#)
 - [Build Another Airline Inventory](#)
 - [Dump The BOM Tree Content](#)
 - [Result of the Tutorial Program](#)
- [Extend the Pre-Defined BOM Tree](#)
 - [Extend an Airline Inventory Object](#)
 - [Build the Specific BOM Objects](#)
 - [Result of the Tutorial Program](#)

15.2 Introduction

This page contains some tutorial examples that will help you getting started using StdAir. Most examples show how to construct some simple business objects, i.e., instances of the so-named Business Object Model (BOM).

15.2.1 Preparing the StdAir Project for Development

The source code for these examples can be found in the `batches` and `test/stdair` directories. They are compiled along with the rest of the StdAir project. See the User Guide ([Users Guide](#)) for more details on how to build the StdAir project.

15.3 Build a Predefined BOM Tree

A few steps:

- [Instantiate the BOM Root Object](#)
- [Instantiate the \(Airline\) Inventory Object](#)
- [Link the Inventory Object with the BOM Root](#)

15.3.1 Instantiate the BOM Root Object

First, a BOM root object (i.e., a root for all the classes in the project) is instantiated by the `stdair::STDAIR_ServiceContext` context object, when the `stdair::STDAIR_Service` is itself instantiated. The corresponding `StdAir` type (class) is `stdair::BomRoot`.

In the following sample, that object is named `ioBomRoot`, and is given as input/output parameter of the `stdair::CmdBomManager::buildSampleBom()` method:

15.3.2 Instantiate the (Airline) Inventory Object

An airline inventory object can then be instantiated. Let us give it the "BA" airline code (corresponding to `British Airways`) as the object key. That is, an object (let us name it `lBAKey`) of type (class) `stdair::InventoryKey` has first to be instantiated.

Thanks to that key, an airline inventory object, i.e. of type (class) `stdair::Inventory`, can be instantiated. Let us name that airline inventory object `lBAInv`.

15.3.3 Link the Inventory Object with the BOM Root

Then, both objects have to be linked: the airline inventory object (`stdair::Inventory`) has to be linked with the root of the BOM tree (`stdair::BomRoot`). That operation is as simple as using the `stdair::FacBomManager::addToListAndMap()` method:

15.3.4 Build Another Airline Inventory

Another airline inventory object, corresponding to the Air France (`Air France`) company, is instantiated the same way:

See the corresponding full program (`cmd_bom_manager_cpp`) for more details.

15.3.5 Dump The BOM Tree Content

From the `BomRoot` (of type `stdair::BomRoot`) object instance, the list of airline inventories (of type `stdair::Inventory`) can then be retrieved...

... and browsed:

See the corresponding full program (`bom_display_cpp`) for more details.

15.3.6 Result of the Tutorial Program

When the `stdair.cpp` program is run (with the `-b` option), the output should look like:

See the corresponding full program (`batch_stdair.cpp`) for more details.

15.4 Extend the Pre-Defined BOM Tree

Now that we master how to instantiate the pre-defined `StdAir` classes, let us see how to extend that BOM.

15.4.1 Extend an Airline Inventory Object

For instance, let us assume that some (IT) provider (e.g., you) would like to have a specific implementation of the `Inventory` object. The corresponding class has just to extend the `stdair::Inventory` class:

The STL containers have to be defined accordingly too:

See the full class definition (`test_archi_inv.hpp`) and implementation (`test_archi_inv.cpp`) for more details.

15.4.2 Build the Specific BOM Objects

The BOM root object (`stdair::BomRoot`) is instantiated the classical way:

Then, the specific implementation of the airline inventory object (`myprovider::Inventory`) can be instantiated the same way as a standard `Inventory` (`stdair::Inventory`) would be:

Then, the specific implementation of the airline inventory object (`myprovider::Inventory`) is linked to the root of the BOM tree (`stdair::BomRoot`) the same way as the standard `Inventory` (`stdair::Inventory`) would be:

Another specific airline inventory object is instantiated the same way:

From the `BomRoot` (of type `stdair::BomRoot`) object instance, the list of specific airline inventories (of type `stdair::Inventory`) can then be retrieved...

... and browsed:

15.4.3 Result of the Tutorial Program

When this program is run, the output should look like:

See the corresponding full program (StandardAirlineITTestSuite_cpp) for more details.

16 Command-Line Test to Demonstrate How To Use TraDemGen elements

```

*/
// //////////////////////////////////////
// Import section
// //////////////////////////////////////
// STL
#include <sstream>
#include <fstream>
#include <map>
#include <cmath>
// Boost Unit Test Framework (UTF)
#define BOOST_TEST_DYN_LINK
#define BOOST_TEST_MAIN
#define BOOST_TEST_MODULE DemandGenerationTest
#include <boost/test/unit_test.hpp>
// StdAir
#include <stdair/stdair_basic_types.hpp>
#include <stdair/basic/BasConst_General.hpp>
#include <stdair/basic/BasLogParams.hpp>
#include <stdair/basic/BasDBParams.hpp>
#include <stdair/basic/BasFileMgr.hpp>
#include <stdair/basic/ProgressStatusSet.hpp>
#include <stdair/bom/EventStruct.hpp>
#include <stdair/bom/BookingRequestStruct.hpp>
#include <stdair/service/Logger.hpp>
// TraDemGen
#include <trademgen/TRADEMGEN_Exceptions.hpp>
#include <trademgen/TRADEMGEN_Service.hpp>
#include <trademgen/bom/DemandStreamKey.hpp>
#include <trademgen/config/trademgen-paths.hpp>

namespace boost_utf = boost::unit_test;

// (Boost) Unit Test XML Report
std::ofstream utfReportStream ("DemandGenerationTestSuite_utfresults.xml");

struct UnitTestConfig {
    UnitTestConfig() {
        boost_utf::unit_test_log.set_stream (utfReportStream);
        boost_utf::unit_test_log.set_format (boost_utf::XML);
        boost_utf::unit_test_log.set_threshold_level (boost_utf::log_test_units);
        //boost_utf::unit_test_log.set_threshold_level (boost_utf::log_successful_tests);
    }

    ~UnitTestConfig() {
    }
};

// Specific type definitions
typedef std::pair<stdair::Count_T, stdair::Count_T> NbOfEventsPair_T;
typedef std::map<const stdair::DemandStreamKeyStr_T,

```

```

        NbOfEventsPair_T> NbOfEventsByDemandStreamMap_T;

// ////////////////////////////////////////
void testDemandGenerationHelper (const unsigned short iTestFlag,
                                const stdair::Filename_T& iDemandInputFilename,
                                const stdair::DemandGenerationMethod& iDemandGen
                                erationMethod,
                                const bool isBuiltin) {

    // Seed for the random generation
    const stdair::RandomSeed_T lRandomSeed = stdair::DEFAULT_RANDOM_SEED;

    // Output log File
    std::ostream oStr;
    oStr << "DemandGenerationTestSuite_" << iTestFlag << ".log";
    const stdair::Filename_T lLogFilename (oStr.str());

    // Set the log parameters
    std::ofstream logOutputFile;
    // Open and clean the log outputfile
    logOutputFile.open (lLogFilename.c_str());
    logOutputFile.clear();

    // Initialise the TraDemGen service object
    const stdair::BasLogParams lLogParams (stdair::LOG::DEBUG, logOutputFile);
    TRADEMGEN::TRADEMGEN_Service trademgenService (lLogParams, lRandomSeed);

    NbOfEventsByDemandStreamMap_T lNbOfEventsMap;

    // Total number of events
    stdair::Count_T lRefExpectedNbOfEvents (0);
    stdair::Count_T lRefActualNbOfEvents (0);

    // Check whether or not a (CSV) input file should be read
    if (isBuiltin == true) {

        // Build the default sample BOM tree (filled with demand streams) for TraDemG
        en
        trademgenService.buildSampleBom();

        lNbOfEventsMap.insert (NbOfEventsByDemandStreamMap_T::
                                value_type ("SIN-BKK 2010-Feb-08 Y",
                                                NbOfEventsPair_T (4, 60)));
        lNbOfEventsMap.insert (NbOfEventsByDemandStreamMap_T::
                                value_type ("BKK-HKG 2010-Feb-08 Y",
                                                NbOfEventsPair_T (4, 60)));
        lNbOfEventsMap.insert (NbOfEventsByDemandStreamMap_T::
                                value_type ("SIN-HKG 2010-Feb-08 Y",
                                                NbOfEventsPair_T (4, 60)));

        // Total number of events, for the 3 demand streams: 180
        lRefExpectedNbOfEvents = 180;
        lRefActualNbOfEvents = 186;

    } else {

        // Create the DemandStream objects, and insert them within the BOM tree
        const TRADEMGEN::DemandFilePath lDemandFilePath (iDemandInputFilename);
        trademgenService.parseAndLoad (lDemandFilePath);

        lNbOfEventsMap.insert (NbOfEventsByDemandStreamMap_T::
                                value_type ("SIN-HND 2010-Feb-08 Y",
                                                NbOfEventsPair_T (1, 10)));
        lNbOfEventsMap.insert (NbOfEventsByDemandStreamMap_T::
                                value_type ("SIN-HND 2010-Feb-09 Y",
                                                NbOfEventsPair_T (1, 10)));
        lNbOfEventsMap.insert (NbOfEventsByDemandStreamMap_T::

```

```

        value_type ("SIN-BKK 2010-Feb-08 Y",
                    NbOfEventsPair_T (1, 10));
lNbOfEventsMap.insert (NbOfEventsByDemandStreamMap_T::
    value_type ("SIN-BKK 2010-Feb-09 Y",
                NbOfEventsPair_T (1, 10)));

// Total number of events, for the 4 demand streams: 40
lRefExpectedNbOfEvents = 40;
lRefActualNbOfEvents = 40;
}

// Retrieve the expected (mean value of the) number of events to be
// generated
const stdair::Count_T& lExpectedNbOfEventsToBeGenerated =
    trademngenService.getExpectedTotalNumberOfRequestsToBeGenerated();

BOOST_CHECK_EQUAL (lRefExpectedNbOfEvents,
                    std::floor (lExpectedNbOfEventsToBeGenerated));

BOOST_CHECK_MESSAGE (lRefExpectedNbOfEvents ==
    std::floor (lExpectedNbOfEventsToBeGenerated),
    "Expected total number of requests to be generated: "
    << lExpectedNbOfEventsToBeGenerated
    << " (=) "
    << std::floor (lExpectedNbOfEventsToBeGenerated)
    << "). Reference value: " << lRefExpectedNbOfEvents);

const stdair::Count_T& lActualNbOfEventsToBeGenerated =
    trademngenService.generateFirstRequests(iDemandGenerationMethod);

// DEBUG
STDAIR_LOG_DEBUG ("Expected number of events: "
    << lExpectedNbOfEventsToBeGenerated << ", actual: "
    << lActualNbOfEventsToBeGenerated);

// Total number of events, for all the demand streams:
BOOST_CHECK_EQUAL (lRefActualNbOfEvents, lActualNbOfEventsToBeGenerated);

BOOST_CHECK_MESSAGE (lRefActualNbOfEvents == lActualNbOfEventsToBeGenerated,
    "Actual total number of requests to be generated: "
    << lExpectedNbOfEventsToBeGenerated
    << " (=) "
    << std::floor (lExpectedNbOfEventsToBeGenerated)
    << "). Reference value: " << lRefActualNbOfEvents);

const bool isQueueDone = trademngenService.isQueueDone();
BOOST_REQUIRE_MESSAGE (isQueueDone == false,
    "The event queue should not be empty.");

stdair::Count_T idx = 1;
while (trademngenService.isQueueDone() == false) {

    // Get the next event from the event queue
    stdair::EventStruct lEventStruct;
    stdair::ProgressStatusSet lPPS = trademngenService.popEvent (lEventStruct);

    // DEBUG
    STDAIR_LOG_DEBUG ("Popped event: '" << lEventStruct.describe() << "'");

    // Extract the corresponding demand/booking request
    const stdair::BookingRequestStruct& lPoppedRequest =
        lEventStruct.getBookingRequest();

    // DEBUG
    STDAIR_LOG_DEBUG ("Popped booking request: '"
        << lPoppedRequest.describe() << "'");

```

```

// Retrieve the corresponding demand stream
const stdair::DemandGeneratorKey_T& lDemandStreamKey =
    lPoppedRequest.getDemandGeneratorKey();

// Check that the number of booking requests to be generated are correct
const NbOfEventsByDemandStreamMap_T::iterator itNbOfEventsMap =
    lNbOfEventsMap.find (lDemandStreamKey);
BOOST_REQUIRE_MESSAGE (itNbOfEventsMap != lNbOfEventsMap.end(),
    "The demand stream key '" << lDemandStreamKey
    << "' is not expected in that test");

const NbOfEventsPair_T& lNbOfEventsPair = itNbOfEventsMap->second;
stdair::Count_T lCurrentNbOfEvents = lNbOfEventsPair.first;
const stdair::Count_T& lExpectedTotalNbOfEvents = lNbOfEventsPair.second;

// Assess whether more events should be generated for that demand stream
const bool stillHavingRequestsToBeGenerated = trademgenService.
    stillHavingRequestsToBeGenerated (lDemandStreamKey, lPPS,
    iDemandGenerationMethod);

if (lCurrentNbOfEvents == 1) {
    const stdair::ProgressStatus& lDemandStreamProgressStatus =
        lPPS.getSpecificGeneratorStatus();
    const stdair::Count_T& lNbOfRequests =
        lDemandStreamProgressStatus.getExpectedNb();

    BOOST_CHECK_EQUAL (lNbOfRequests, lExpectedTotalNbOfEvents);
    BOOST_CHECK_MESSAGE (lNbOfRequests == lExpectedTotalNbOfEvents,
        "[" << lDemandStreamKey
        << "]" Total number of requests to be generated: "
        << lNbOfRequests << "). Expected value: "
        << lExpectedTotalNbOfEvents);
}

// DEBUG
STDAIR_LOG_DEBUG ("=> [" << lDemandStreamKey << "]"[" << lCurrentNbOfEvents
    << "/" << lExpectedTotalNbOfEvents
    << "]" is now processed. "
    << "Still generate events for that demand stream? "
    << stillHavingRequestsToBeGenerated);

// If there are still events to be generated for that demand stream,
// generate and add them to the event queue
if (stillHavingRequestsToBeGenerated == true) {
    const stdair::BookingRequestPtr_T lNextRequest_ptr =
        trademgenService.generateNextRequest (lDemandStreamKey,
        iDemandGenerationMethod);

    assert (lNextRequest_ptr != NULL);

    const stdair::Duration_T lDuration =
        lNextRequest_ptr->getRequestDateTime()
        - lPoppedRequest.getRequestDateTime();
    BOOST_REQUIRE_GT (lDuration.total_milliseconds(), 0);
    BOOST_REQUIRE_MESSAGE (lDuration.total_milliseconds() > 0,
        "[" << lDemandStreamKey
        << "]" The date-time of the generated event ("
        << lNextRequest_ptr->getRequestDateTime()
        << ") is lower than the date-time "
        << "of the current event ("
        << lPoppedRequest.getRequestDateTime() << ")");

    // DEBUG
    STDAIR_LOG_DEBUG ("[" << lDemandStreamKey << "]"[" << lCurrentNbOfEvents
        << "/" << lExpectedTotalNbOfEvents
        << "]" Added request: '" << lNextRequest_ptr->describe()
        << "'. Is queue done? "
        << trademgenService.isQueueDone());
}

```

```

        // Keep, within the dedicated map, the current counters of events updated.
        ++lCurrentNbOfEvents;
        itNbOfEventsMap->second = NbOfEventsPair_T (lCurrentNbOfEvents,
                                                    lExpectedTotalNbOfEvents);
    }

    // Iterate
    ++idx;
}
// Compensate for the last iteration
--idx;

if (iDemandGenerationMethod == stdair::DemandGenerationMethod::STA_ORD) {
    //
    BOOST_CHECK_EQUAL (idx, lRefActualNbOfEvents);
    BOOST_CHECK_MESSAGE (idx == lRefActualNbOfEvents,
                        "The total actual number of events is "
                        << lRefActualNbOfEvents << ", but " << idx
                        << " events have been generated");
}

trademgenService.reset();

// DEBUG
STDAIR_LOG_DEBUG ("End of the simulation");

// Close the log file
logOutputFile.close();
}

// ////////////////////////////////// Main: Unit Test Suite //////////////////////////////////

// Set the UTF configuration (re-direct the output to a specific file)
BOOST_GLOBAL_FIXTURE (UnitTestFixture);

// Start the test suite
BOOST_AUTO_TEST_SUITE (master_test_suite)

BOOST_AUTO_TEST_CASE (trademgen_simple_simulation_test) {

    // Input file name
    const stdair::Filename_T lInputFilename (STDAIR_SAMPLE_DIR "/demand01.csv");

    // Generate the date time of the requests with the statistic order method.
    const stdair::DemandGenerationMethod lDemandGenerationMethod (stdair::DemandGenerationMethod::STA_ORD);

    // State whether the BOM tree should be built-in or parsed from an input file
    const bool isBuiltin = false;
    BOOST_CHECK_NO_THROW (testDemandGenerationHelper(0,
                                                    lInputFilename,
                                                    lDemandGenerationMethod,
                                                    isBuiltin));
}

BOOST_AUTO_TEST_CASE (trademgen_missing_input_file_test) {

    // Input file name
    const stdair::Filename_T lInputFilename (STDAIR_SAMPLE_DIR "/missingFile.csv");

    // Generate the date time of the requests with the statistic order method.

```

```

const stdair::DemandGenerationMethod lDemandGenerationMethod (stdair::DemandGen
erationMethod::STA_ORD);

// State whether the BOM tree should be built-in or parsed from an input file
const bool isBuiltin = false;
BOOST_CHECK_THROW (testDemandGenerationHelper(1,
                                                lInputFilename,
                                                lDemandGenerationMethod,
                                                isBuiltin),
                  TRADEMGEN::DemandInputFileNotFoundException);

}

BOOST_AUTO_TEST_CASE (trademgen_default_bom_simulation_test) {

    // Generate the date time of the requests with the statistic order method.
    const stdair::DemandGenerationMethod lDemandGenerationMethod (stdair::DemandGen
erationMethod::STA_ORD);

    // State whether the BOM tree should be built-in or parsed from an input file
    const bool isBuiltin = true;
    BOOST_CHECK_NO_THROW (testDemandGenerationHelper(2,
                                                    " ",
                                                    lDemandGenerationMethod,
                                                    isBuiltin));

}

BOOST_AUTO_TEST_CASE (trademgen_poisson_process_test) {

    // Generate the date time of the requests with the poisson process.
    const stdair::DemandGenerationMethod lDemandGenerationMethod (stdair::DemandGen
erationMethod::POI_PRO);

    // State whether the BOM tree should be built-in or parsed from an input file
    const bool isBuiltin = true;
    BOOST_CHECK_NO_THROW (testDemandGenerationHelper(3,
                                                    " ",
                                                    lDemandGenerationMethod,
                                                    isBuiltin));

}

// End the test suite
BOOST_AUTO_TEST_SUITE_END()

/*!

```

17 Directory Hierarchy

17.1 Directories

This directory hierarchy is sorted roughly, but not completely, alphabetically:

test	57
trademgen	57
trademgen	57
basic	55

batches	55
bom	56
command	56
factory	56
python	56
service	57
ui	57
cmdline	56

18 Namespace Index

18.1 Namespace List

Here is a list of all namespaces with brief descriptions:

SEVMGR	58
stdair (Forward declarations)	59
TRADEMGEN	59
TRADEMGEN::DemandParserHelper	68

19 Class Index

19.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

```
std::basic_fstream< char >
std::basic_fstream< wchar_t >
std::basic_ifstream< char >
std::basic_ifstream< wchar_t >
std::basic_ios< char >
std::basic_ios< wchar_t >
std::basic_iostream< char >
std::basic_iostream< wchar_t >
std::basic_istream< char >
std::basic_istream< wchar_t >
std::basic_istreamstream< char >
std::basic_istreamstream< wchar_t >
std::basic_ofstream< char >
std::basic_ofstream< wchar_t >
std::basic_ostream< char >
std::basic_ostream< wchar_t >
```


std::basic_ostringstream< char >	
std::basic_ostringstream< wchar_t >	
std::basic_string< char >	
std::basic_string< wchar_t >	
std::basic_stringstream< char >	
std::basic_stringstream< wchar_t >	
BomAbstract	72
TRADEMGEN::DemandStream	114
TRADEMGEN::BomDisplay	73
stdair::CategoricalAttribute< T >	74
TRADEMGEN::CategoricalAttributeLite< T >	76
CmdAbstract	78
TRADEMGEN::DemandFileParser	107
TRADEMGEN::DemandManager	110
TRADEMGEN::DemandParser	111
TRADEMGEN::ContinuousAttribute< T >	79
TRADEMGEN::ContinuousAttributeLite< T >	81
TRADEMGEN::DBManager	83
TRADEMGEN::DefaultMap	88
TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT >	88
TRADEMGEN::DictionaryManager	140
FacServiceAbstract	143
TRADEMGEN::FacTRADEMGENServiceContext	144
FileNotFoundException	145
TRADEMGEN::DemandInputFileNotFoundException	109
TRADEMGEN::FlagSaver	146
grammar	146
TRADEMGEN::DemandParserHelper::DemandParser	112
InputFilePath	147
TRADEMGEN::DemandFilePath	108
KeyAbstract	148
TRADEMGEN::DemandStreamKey	130

TRADEMGEN::DemandParserHelper::ParserSemanticAction	148
TRADEMGEN::DemandParserHelper::doEndDemand	141
TRADEMGEN::DemandParserHelper::storeChannelCode	153
TRADEMGEN::DemandParserHelper::storeChannelProbMass	155
TRADEMGEN::DemandParserHelper::storeDemandChangeFeeDisutility	157
TRADEMGEN::DemandParserHelper::storeDemandChangeFeeProb	158
TRADEMGEN::DemandParserHelper::storeDemandMean	160
TRADEMGEN::DemandParserHelper::storeDemandNonRefundableDisutility	162
TRADEMGEN::DemandParserHelper::storeDemandNonRefundableProb	163
TRADEMGEN::DemandParserHelper::storeDemandStdDev	165
TRADEMGEN::DemandParserHelper::storeDestination	167
TRADEMGEN::DemandParserHelper::storeDow	168
TRADEMGEN::DemandParserHelper::storeDTD	170
TRADEMGEN::DemandParserHelper::storeDTDProbMass	172
TRADEMGEN::DemandParserHelper::storeFFCode	173
TRADEMGEN::DemandParserHelper::storeFFProbMass	175
TRADEMGEN::DemandParserHelper::storeOrigin	176
TRADEMGEN::DemandParserHelper::storePosCode	178
TRADEMGEN::DemandParserHelper::storePosProbMass	180
TRADEMGEN::DemandParserHelper::storePrefCabin	181
TRADEMGEN::DemandParserHelper::storePrefDepDateRangeEnd	183
TRADEMGEN::DemandParserHelper::storePrefDepDateRangeStart	185
TRADEMGEN::DemandParserHelper::storePrefDepTime	187
TRADEMGEN::DemandParserHelper::storePrefDepTimeProbMass	188
TRADEMGEN::DemandParserHelper::storeStayCode	190
TRADEMGEN::DemandParserHelper::storeStayProbMass	192
TRADEMGEN::DemandParserHelper::storeTimeValue	193
TRADEMGEN::DemandParserHelper::storeTimeValueProbMass	195
TRADEMGEN::DemandParserHelper::storeTripCode	197

TRADEMGEN::DemandParserHelper::storeTripProbMass	198
TRADEMGEN::DemandParserHelper::storeWTP	200
RootException	153
TRADEMGEN::TrademgenGenerationException	216
TRADEMGEN::IndexOutOfRangeException	147
ServiceAbstract	153
TRADEMGEN::TRADEMGEN_ServiceContext	214
StructAbstract	202
TRADEMGEN::DemandCharacteristics	100
TRADEMGEN::DemandDistribution	105
TRADEMGEN::DemandStruct	132
TRADEMGEN::RandomGenerationContext	150
TestFixture	202
DemandGenerationTestSuite	108
TRADEMGEN::TRADEMGEN_Abstract	202
TRADEMGEN::DBParams	84
TRADEMGEN::TRADEMGEN_Service	204
TRADEMGEN::Trademgener	215

20 Class Index

20.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

BomAbstract	72
TRADEMGEN::BomDisplay (Utility class to display TraDemGen objects with a pretty format)	73
stdair::CategoricalAttribute< T > (Class modeling the distribution of values that can be taken by a categorical attribute)	74
TRADEMGEN::CategoricalAttributeLite< T > (Class modeling the distribution of values that can be taken by a categorical attribute)	76
CmdAbstract	78
TRADEMGEN::ContinuousAttribute< T >	79

TRADEMGEN::ContinuousAttributeLite< T > (Class modeling the distribution of values that can be taken by a continuous attribute)	81
TRADEMGEN::DBManager	83
TRADEMGEN::DBParams	84
TRADEMGEN::DefaultMap	88
TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT >	88
TRADEMGEN::DemandCharacteristics (Class modeling the characteristics of a demand type)	100
TRADEMGEN::DemandDistribution (Class modeling the distribution of a demand type)	105
TRADEMGEN::DemandFileParser	107
TRADEMGEN::DemandFilePath	108
DemandGenerationTestSuite	108
TRADEMGEN::DemandInputFileNotFoundException	109
TRADEMGEN::DemandManager (Utility class for Demand and DemandStream objects)	110
TRADEMGEN::DemandParser (Class wrapping the parser entry point)	111
TRADEMGEN::DemandParserHelper::DemandParser	112
TRADEMGEN::DemandStream (Class modeling a demand stream)	114
TRADEMGEN::DemandStreamKey	130
TRADEMGEN::DemandStruct	132
TRADEMGEN::DictionaryManager (Class wrapper of dictionary business methods)	140
TRADEMGEN::DemandParserHelper::doEndDemand	141
FacServiceAbstract	143
TRADEMGEN::FacTRADEMGENServiceContext (Factory for creating the TraDemGen service context instance)	144
FileNotFoundException	145
TRADEMGEN::FlagSaver	146
grammar	146
TRADEMGEN::IndexOutOfRangeException	147
InputFilePath	147
KeyAbstract	148
TRADEMGEN::DemandParserHelper::ParserSemanticAction	148

TRADEMGEN::RandomGenerationContext	150
RootException	153
ServiceAbstract	153
TRADEMGEN::DemandParserHelper::storeChannelCode	153
TRADEMGEN::DemandParserHelper::storeChannelProbMass	155
TRADEMGEN::DemandParserHelper::storeDemandChangeFeeDisutility	157
TRADEMGEN::DemandParserHelper::storeDemandChangeFeeProb	158
TRADEMGEN::DemandParserHelper::storeDemandMean	160
TRADEMGEN::DemandParserHelper::storeDemandNonRefundableDisutility	162
TRADEMGEN::DemandParserHelper::storeDemandNonRefundableProb	163
TRADEMGEN::DemandParserHelper::storeDemandStdDev	165
TRADEMGEN::DemandParserHelper::storeDestination	167
TRADEMGEN::DemandParserHelper::storeDow	168
TRADEMGEN::DemandParserHelper::storeDTD	170
TRADEMGEN::DemandParserHelper::storeDTDProbMass	172
TRADEMGEN::DemandParserHelper::storeFFCode	173
TRADEMGEN::DemandParserHelper::storeFFProbMass	175
TRADEMGEN::DemandParserHelper::storeOrigin	176
TRADEMGEN::DemandParserHelper::storePosCode	178
TRADEMGEN::DemandParserHelper::storePosProbMass	180
TRADEMGEN::DemandParserHelper::storePrefCabin	181
TRADEMGEN::DemandParserHelper::storePrefDepDateRangeEnd	183
TRADEMGEN::DemandParserHelper::storePrefDepDateRangeStart	185
TRADEMGEN::DemandParserHelper::storePrefDepTime	187
TRADEMGEN::DemandParserHelper::storePrefDepTimeProbMass	188
TRADEMGEN::DemandParserHelper::storeStayCode	190
TRADEMGEN::DemandParserHelper::storeStayProbMass	192
TRADEMGEN::DemandParserHelper::storeTimeValue	193
TRADEMGEN::DemandParserHelper::storeTimeValueProbMass	195

TRADEMGEN::DemandParserHelper::storeTripCode	197
TRADEMGEN::DemandParserHelper::storeTripProbMass	198
TRADEMGEN::DemandParserHelper::storeWTP	200
StructAbstract	202
TestFixture	202
TRADEMGEN::TRADEMGEN_Abstract	202
TRADEMGEN::TRADEMGEN_Service (Class holding the services related to Travel Demand Generation)	204
TRADEMGEN::TRADEMGEN_ServiceContext (Class holding the context of the Trademgen services)	214
TRADEMGEN::Trademgener (Wrapper structure around the C++ API, so as to expose a Python API)	215
TRADEMGEN::TrademgenGenerationException	216

21 File Index

21.1 File List

Here is a list of all files with brief descriptions:

test/trademgen/DemandGenerationTestSuite.cpp	219
test/trademgen/DemandGenerationTestSuite.hpp	226
test/trademgen/generateEvents.cpp	228
trademgen/DBParams.hpp	387
trademgen/TRADEMGEN_Abstract.hpp	424
trademgen/TRADEMGEN_Exceptions.hpp	426
trademgen/TRADEMGEN_Service.hpp	428
trademgen/TRADEMGEN_Types.hpp	432
trademgen/basic/BasConst.cpp	230
trademgen/basic/BasConst_DemandGeneration.hpp	233
trademgen/basic/BasConst_TRADEMGEN_Service.hpp	235
trademgen/basic/BasParserTypes.hpp	237
trademgen/basic/CategoricalAttribute.hpp	240

trademgen/basic/CategoricalAttributeLite.hpp	244
trademgen/basic/ContinuousAttribute.hpp	248
trademgen/basic/ContinuousAttributeLite.hpp	252
trademgen/basic/DemandCharacteristics.cpp	256
trademgen/basic/DemandCharacteristics.hpp	260
trademgen/basic/DemandCharacteristicsTypes.hpp	263
trademgen/basic/DemandDistribution.cpp	266
trademgen/basic/DemandDistribution.hpp	268
trademgen/basic/DictionaryManager.cpp	270
trademgen/basic/DictionaryManager.hpp	272
trademgen/basic/RandomGenerationContext.cpp	274
trademgen/basic/RandomGenerationContext.hpp	276
trademgen/batches/trademgen_generateDemand.cpp	283
trademgen/batches/trademgen_with_db.cpp	294
trademgen/bom/BomDisplay.cpp	301
trademgen/bom/BomDisplay.hpp	304
trademgen/bom/DemandStream.cpp	306
trademgen/bom/DemandStream.hpp	317
trademgen/bom/DemandStreamKey.cpp	323
trademgen/bom/DemandStreamKey.hpp	325
trademgen/bom/DemandStreamTypes.hpp	328
trademgen/bom/DemandStruct.cpp	330
trademgen/bom/DemandStruct.hpp	334
trademgen/command/DBManager.cpp	337
trademgen/command/DBManager.hpp	341
trademgen/command/DemandManager.cpp	343
trademgen/command/DemandManager.hpp	359
trademgen/command/DemandParser.cpp	362
trademgen/command/DemandParser.hpp	364

trademgen/command/DemandParserHelper.cpp	366
trademgen/command/DemandParserHelper.hpp	382
trademgen/factory/FacTRADEMGENServiceContext.cpp	390
trademgen/factory/FacTRADEMGENServiceContext.hpp	392
trademgen/python/pytrademgen.cpp	395
trademgen/service/TRADEMGEN_Service.cpp	402
trademgen/service/TRADEMGEN_ServiceContext.cpp	418
trademgen/service/TRADEMGEN_ServiceContext.hpp	421
trademgen/ui/cmdline/trademgen.cpp	434

22 Directory Documentation

22.1 trademgen/basic/ Directory Reference

Files

- file [BasConst.cpp](#)
- file [BasConst_DemandGeneration.hpp](#)
- file [BasConst_TRADEMGEN_Service.hpp](#)
- file [BasParserTypes.hpp](#)
- file [CategoricalAttribute.hpp](#)
- file [CategoricalAttributeLite.hpp](#)
- file [ContinuousAttribute.hpp](#)
- file [ContinuousAttributeLite.hpp](#)
- file [DemandCharacteristics.cpp](#)
- file [DemandCharacteristics.hpp](#)
- file [DemandCharacteristicsTypes.hpp](#)
- file [DemandDistribution.cpp](#)
- file [DemandDistribution.hpp](#)
- file [DictionaryManager.cpp](#)
- file [DictionaryManager.hpp](#)
- file [RandomGenerationContext.cpp](#)
- file [RandomGenerationContext.hpp](#)

22.2 trademgen/batches/ Directory Reference

Files

- file [trademgen_generateDemand.cpp](#)
- file [trademgen_with_db.cpp](#)

22.3 trademgen/bom/ Directory Reference

Files

- file [BomDisplay.cpp](#)
- file [BomDisplay.hpp](#)
- file [DemandStream.cpp](#)
- file [DemandStream.hpp](#)
- file [DemandStreamKey.cpp](#)
- file [DemandStreamKey.hpp](#)
- file [DemandStreamTypes.hpp](#)
- file [DemandStruct.cpp](#)
- file [DemandStruct.hpp](#)

22.4 trademgen/ui/cmdline/ Directory Reference

Files

- file [trademgen.cpp](#)

22.5 trademgen/command/ Directory Reference

Files

- file [DBManager.cpp](#)
- file [DBManager.hpp](#)
- file [DemandManager.cpp](#)
- file [DemandManager.hpp](#)
- file [DemandParser.cpp](#)
- file [DemandParser.hpp](#)
- file [DemandParserHelper.cpp](#)
- file [DemandParserHelper.hpp](#)

22.6 trademgen/factory/ Directory Reference

Files

- file [FacTRADEMGENServiceContext.cpp](#)
- file [FacTRADEMGENServiceContext.hpp](#)

22.7 trademgen/python/ Directory Reference

Files

- file [pytrademgen.cpp](#)

22.8 trademgen/service/ Directory Reference

Files

- file [TRADEMGEN_Service.cpp](#)
- file [TRADEMGEN_ServiceContext.cpp](#)
- file [TRADEMGEN_ServiceContext.hpp](#)

22.9 test/ Directory Reference

Directories

- directory [trademgen](#)

22.10 trademgen/ Directory Reference

Directories

- directory [basic](#)
- directory [batches](#)
- directory [bom](#)
- directory [command](#)
- directory [factory](#)
- directory [python](#)
- directory [service](#)
- directory [ui](#)

Files

- file [DBParams.hpp](#)
- file [TRADEMGEN_Abstract.hpp](#)
- file [TRADEMGEN_Exceptions.hpp](#)
- file [TRADEMGEN_Service.hpp](#)
- file [TRADEMGEN_Types.hpp](#)

22.11 test/trademgen/ Directory Reference

Files

- file [DemandGenerationTestSuite.cpp](#)
- file [DemandGenerationTestSuite.hpp](#)
- file [generateEvents.cpp](#)

22.12 trademgen/ui/ Directory Reference

Directories

- directory [cmdline](#)

23 Namespace Documentation

23.1 SEVMGR Namespace Reference

Functions

- `template void SEVMGR_Service::addEventGenerator< TRADEMGEN::DemandStream > (TRADEMGENT::DemandStream &) const`
- `template TRADEMGENT::DemandStream & SEVMGR_Service::getEventGenerator< TRADEMGENT::DemandStream, stdair::DemandStreamKeyStr_T > (const stdair::DemandStreamKeyStr_T &) const`
- `template bool SEVMGR_Service::hasEventGenerator< TRADEMGENT::DemandStream, stdair::DemandStreamKeyStr_T > (const stdair::DemandStreamKeyStr_T &) const`
- `template const TRADEMGENT::DemandStreamList_T SEVMGR_Service::getEventGeneratorList< TRADEMGENT::DemandStream > () const`
- `template bool SEVMGR_Service::hasEventGeneratorList< TRADEMGENT::DemandStream > () const`

23.1.1 Function Documentation

23.1.1.1 `template void SEVMGR::SEVMGR_Service::addEventGenerator< TRADEMGENT::DemandStream > (TRADEMGENT::DemandStream &) const`

Explicit template instantiations with the TraDemGen own EventGenerator type: DemandStream.

23.1.1.2 `template TRADEMGENT::DemandStream& SEVMGR::SEVMGR_Service::getEventGenerator< TRADEMGENT::DemandStream, stdair::DemandStreamKeyStr_T > (const stdair::DemandStreamKeyStr_T &) const`

23.1.1.3 `template bool SEVMGR::SEVMGR_Service::hasEventGenerator< TRADEMGENT::DemandStream, stdair::DemandStreamKeyStr_T > (const stdair::DemandStreamKeyStr_T &) const`

23.1.1.4 `template const TRADEMGENT::DemandStreamList_T SEVMGR::SEVMGR_Service::getEventGeneratorList< TRADEMGENT::DemandStream > () const`

23.1.1.5 `template bool SEVMGR::SEVMGR_Service::hasEventGeneratorList< TRADEMGENT::DemandStream > () const`

23.2 stdair Namespace Reference

Forward declarations.

Classes

- struct [CategoricalAttribute](#)
Class modeling the distribution of values that can be taken by a categorical attribute.

23.2.1 Detailed Description

Forward declarations.

23.3 TRADEMGEN Namespace Reference

Namespaces

- namespace [DemandParserHelper](#)

Classes

- struct [DefaultMap](#)
- struct [CategoricalAttributeLite](#)
Class modeling the distribution of values that can be taken by a categorical attribute.
- struct [ContinuousAttribute](#)
- struct [ContinuousAttributeLite](#)
Class modeling the distribution of values that can be taken by a continuous attribute.
- struct [DemandCharacteristics](#)
Class modeling the characteristics of a demand type.
- struct [DemandDistribution](#)
Class modeling the distribution of a demand type.
- class [DictionaryManager](#)
Class wrapper of dictionary business methods.
- struct [RandomGenerationContext](#)
- struct [FlagSaver](#)
- class [BomDisplay](#)
Utility class to display TraDemGen objects with a pretty format.
- class [DemandStream](#)
Class modeling a demand stream.
- struct [DemandStreamKey](#)
- struct [DemandStruct](#)

- class [DBManager](#)
- class [DemandManager](#)
Utility class for Demand and [DemandStream](#) objects.
- class [DemandParser](#)
Class wrapping the parser entry point.
- class [DemandFileParser](#)
- struct [DBParams](#)
- class [FacTRADEMGENSEerviceContext](#)
Factory for creating the TraDemGen service context instance.
- struct [Trademgener](#)
Wrapper structure around the C++ API, so as to expose a Python API.
- class [TRADEMGEN_ServiceContext](#)
Class holding the context of the Trademgen services.
- struct [TRADEMGEN_Abstract](#)
- class [TrademgenGenerationException](#)
- class [DemandInputFileNotFoundException](#)
- class [IndexOutOfRangeException](#)
- class [TRADEMGEN_Service](#)
class holding the services related to Travel Demand Generation.
- class [DemandFilePath](#)

Typedefs

- typedef char [char_t](#)
- typedef boost::spirit::classic::file_iterator< [char_t](#) > [iterator_t](#)
- typedef boost::spirit::classic::scanner< [iterator_t](#) > [scanner_t](#)
- typedef boost::spirit::classic::rule< [scanner_t](#) > [rule_t](#)
- typedef boost::spirit::classic::int_parser< unsigned int, 10, 1, 1 > [int1_p_t](#)
- typedef boost::spirit::classic::uint_parser< unsigned int, 10, 2, 2 > [uint2_p_t](#)
- typedef boost::spirit::classic::uint_parser< unsigned int, 10, 1, 2 > [uint1_2_p_t](#)
- typedef boost::spirit::classic::uint_parser< unsigned int, 10, 1, 3 > [uint1_3_p_t](#)
- typedef boost::spirit::classic::uint_parser< unsigned int, 10, 4, 4 > [uint4_p_t](#)
- typedef boost::spirit::classic::uint_parser< unsigned int, 10, 1, 4 > [uint1_4_p_t](#)
- typedef boost::spirit::classic::chset< [char_t](#) > [chset_t](#)
- typedef boost::spirit::classic::impl::loop_traits< [chset_t](#), unsigned int, unsigned int >::type [repeat_p_t](#)
- typedef boost::spirit::classic::bounded< [uint2_p_t](#), unsigned int > [bounded2_p_t](#)
- typedef boost::spirit::classic::bounded< [uint1_2_p_t](#), unsigned int > [bounded1_2_p_t](#)
- typedef boost::spirit::classic::bounded< [uint1_3_p_t](#), unsigned int > [bounded1_3_p_t](#)
- typedef boost::spirit::classic::bounded< [uint4_p_t](#), unsigned int > [bounded4_p_t](#)
- typedef boost::spirit::classic::bounded< [uint1_4_p_t](#), unsigned int > [bounded1_4_p_t](#)
- typedef [ContinuousAttributeLite](#)< std::float_duration::FloatDuration_T > [ContinuousFloatDuration_T](#)
- typedef [ContinuousFloatDuration_T](#)::ContinuousDistribution_T [ArrivalPatternCumulativeDistribution_T](#)

- typedef [CategoricalAttributeLite](#)< stdair::AirportCode_T > [POSProbabilityMass_T](#)
- typedef [POSProbabilityMass_T::ProbabilityMassFunction_T](#) [POSProbabilityMassFunction_T](#)
- typedef [CategoricalAttributeLite](#)< stdair::ChannelLabel_T > [ChannelProbabilityMass_T](#)
- typedef [ChannelProbabilityMass_T::ProbabilityMassFunction_T](#) [ChannelProbabilityMassFunction_T](#)
- typedef [CategoricalAttributeLite](#)< stdair::TripType_T > [TripTypeProbabilityMass_T](#)
- typedef [TripTypeProbabilityMass_T::ProbabilityMassFunction_T](#) [TripTypeProbabilityMassFunction_T](#)
- typedef [CategoricalAttributeLite](#)< stdair::DayDuration_T > [StayDurationProbabilityMass_T](#)
- typedef [StayDurationProbabilityMass_T::ProbabilityMassFunction_T](#) [StayDurationProbabilityMassFunction_T](#)
- typedef [CategoricalAttributeLite](#)< stdair::FrequentFlyer_T > [FrequentFlyerProbabilityMass_T](#)
- typedef [FrequentFlyerProbabilityMass_T::ProbabilityMassFunction_T](#) [FrequentFlyerProbabilityMassFunction_T](#)
- typedef [ContinuousAttributeLite](#)< stdair::IntDuration_T > [PreferredDepartureTimeCumulativeDistribution_T](#)
- typedef [PreferredDepartureTimeCumulativeDistribution_T::ContinuousDistribution_T](#) [PreferredDepartureTimeContinuousDistribution_T](#)
- typedef [ContinuousAttributeLite](#)< stdair::PriceValue_T > [ValueOfTimeCumulativeDistribution_T](#)
- typedef [ValueOfTimeCumulativeDistribution_T::ContinuousDistribution_T](#) [ValueOfTimeContinuousDistribution_T](#)
- typedef [ContinuousAttributeLite](#)< stdair::RealNumber_T > [CumulativeDistribution_T](#)
- typedef [CumulativeDistribution_T::ContinuousDistribution_T](#) [FRAT5Pattern_T](#)
- typedef stdair::Probability_T [DictionaryKey_T](#)
- typedef std::list< [DemandStream](#) * > [DemandStreamList_T](#)
- typedef std::map< const stdair::MapKey_T, [DemandStream](#) * > [DemandStreamMap_T](#)
- typedef std::list< std::string > [DBParamsNameList_T](#)
- typedef boost::shared_ptr< [TRADEMGEN_Service](#) > [TRADEMGEN_ServicePtr_T](#)

Functions

- stdair::BaseGenerator_T [DEFAULT_BASE_GENERATOR](#) (stdair::DEFAULT_RANDOM_SEED)
- stdair::UniformGenerator_T [DEFAULT_UNIFORM_GENERATOR](#) ([DEFAULT_BASE_GENERATOR](#), [DEFAULT_UNIFORM_REAL_DISTRIBUTION](#))
- void [stat_display](#) (std::ostream &oStream, const [stat_acc_type](#) &iStatAcc)

Variables

- const [POSProbabilityMassFunction_T](#) [DEFAULT_POS_PROBALILITY_MASS](#)
- const stdair::FloatDuration_T [DEFAULT_LAST_LOWER_BOUND_ARRIVAL_PATTERN](#) = -1
- const [FRAT5Pattern_T](#) [DEFAULT_FRAT5_PATTERN](#) = [DefaultMap::createFRAT5Pattern\(\)](#)
- const double [DEFAULT_MAX_ADVANCE_PURCHASE](#) = 330.0
- const stdair::UniformDistribution_T [DEFAULT_UNIFORM_REAL_DISTRIBUTION](#)
- stdair::BaseGenerator_T [DEFAULT_BASE_GENERATOR](#)
- stdair::UniformGenerator_T [DEFAULT_UNIFORM_GENERATOR](#)

23.3.1 Typedef Documentation

23.3.1.1 `typedef char TRADEMGEN::char_t`

Definition at line 31 of file [BasParserTypes.hpp](#).

23.3.1.2 `typedef boost::spirit::classic::file_iterator<char_t> TRADEMGEN::iterator_t`

Definition at line 35 of file [BasParserTypes.hpp](#).

23.3.1.3 `typedef boost::spirit::classic::scanner<iterator_t> TRADEMGEN::scanner_t`

Definition at line 36 of file [BasParserTypes.hpp](#).

23.3.1.4 `typedef boost::spirit::classic::rule<scanner_t> TRADEMGEN::rule_t`

Definition at line 37 of file [BasParserTypes.hpp](#).

23.3.1.5 `typedef boost::spirit::classic::int_parser<unsigned int, 10, 1, 1> TRADEMGEN::int1_p_t`

1-digit-integer parser

Definition at line 45 of file [BasParserTypes.hpp](#).

23.3.1.6 `typedef boost::spirit::classic::uint_parser<unsigned int, 10, 2, 2> TRADEMGEN::uint2_p_t`

2-digit-integer parser

Definition at line 48 of file [BasParserTypes.hpp](#).

23.3.1.7 `typedef boost::spirit::classic::uint_parser<unsigned int, 10, 1, 2> TRADEMGEN::uint1_2_p_t`

Up-to-2-digit-integer parser

Definition at line 51 of file [BasParserTypes.hpp](#).

23.3.1.8 `typedef boost::spirit::classic::uint_parser<unsigned int, 10, 1, 3> TRADEMGEN::uint1_3_p_t`

Up-to-3-digit-integer parser

Definition at line 54 of file [BasParserTypes.hpp](#).

23.3.1.9 `typedef boost::spirit::classic::uint_parser<unsigned int, 10, 4, 4>
TRADEMGENT::uint4_p_t`

4-digit-integer parser

Definition at line 57 of file [BasParserTypes.hpp](#).

23.3.1.10 `typedef boost::spirit::classic::uint_parser<unsigned int, 10, 1, 4>
TRADEMGENT::uint1_4_p_t`

Up-to-4-digit-integer parser

Definition at line 60 of file [BasParserTypes.hpp](#).

23.3.1.11 `typedef boost::spirit::classic::chset<char_t> TRADEMGENT::chset_t`

character set

Definition at line 63 of file [BasParserTypes.hpp](#).

23.3.1.12 `typedef boost::spirit::classic::impl::loop_traits<chset_t, unsigned int, unsigned
int>::type TRADEMGENT::repeat_p_t`

(Repeating) sequence of a given number of characters: repeat_p(min, max)

Definition at line 69 of file [BasParserTypes.hpp](#).

23.3.1.13 `typedef boost::spirit::classic::bounded<uint2_p_t, unsigned int>
TRADEMGENT::bounded2_p_t`

Bounded-number-of-integers parser

Definition at line 72 of file [BasParserTypes.hpp](#).

23.3.1.14 `typedef boost::spirit::classic::bounded<uint1_2_p_t, unsigned int>
TRADEMGENT::bounded1_2_p_t`

Definition at line 73 of file [BasParserTypes.hpp](#).

23.3.1.15 `typedef boost::spirit::classic::bounded<uint1_3_p_t, unsigned int>
TRADEMGENT::bounded1_3_p_t`

Definition at line 74 of file [BasParserTypes.hpp](#).

23.3.1.16 `typedef boost::spirit::classic::bounded<uint4_p_t, unsigned int>
TRADEMGENT::bounded4_p_t`

Definition at line 75 of file [BasParserTypes.hpp](#).

23.3.1.17 `typedef boost::spirit::classic::bounded<uint1_4_p_t, unsigned int>
TRADEMGENT::bounded1_4_p_t`

Definition at line 76 of file [BasParserTypes.hpp](#).

23.3.1.18 `typedef ContinuousAttributeLite<stdair::FloatDuration_T>
TRADEMGENT::ContinuousFloatDuration_T`

Type definition for the continuous distribution of the duration (as a float number).

Definition at line 19 of file [DemandCharacteristicsTypes.hpp](#).

23.3.1.19 `typedef ContinuousFloatDuration_T::ContinuousDistribution_T
TRADEMGENT::ArrivalPatternCumulativeDistribution_T`

Type definition for the arrival pattern cumulative distribution.

Definition at line 22 of file [DemandCharacteristicsTypes.hpp](#).

23.3.1.20 `typedef CategoricalAttributeLite<stdair::AirportCode_T>
TRADEMGENT::POSProbabilityMass_T`

Define the point-of-sale probability mass.

Definition at line 25 of file [DemandCharacteristicsTypes.hpp](#).

23.3.1.21 `typedef POSProbabilityMass_T::ProbabilityMassFunction_T
TRADEMGENT::POSProbabilityMassFunction_T`

Define the probability mass function type of point-of-sale.

Definition at line 28 of file [DemandCharacteristicsTypes.hpp](#).

23.3.1.22 `typedef CategoricalAttributeLite<stdair::ChannelLabel_T>
TRADEMGENT::ChannelProbabilityMass_T`

Define the booking channel probability mass.

Definition at line 31 of file [DemandCharacteristicsTypes.hpp](#).

23.3.1.23 `typedef ChannelProbabilityMass_T::ProbabilityMassFunction_T
TRADEMGENT::ChannelProbabilityMassFunction_T`

Define the probability mass function type of booking channel.

Definition at line 34 of file [DemandCharacteristicsTypes.hpp](#).

23.3.1.24 `typedef CategoricalAttributeLite<stdair::TripType_T>
TRADEMGENTripTypeProbabilityMass_T`

Define the trip type probability mass.

Definition at line 37 of file [DemandCharacteristicsTypes.hpp](#).

23.3.1.25 `typedef TripTypeProbabilityMass_T::ProbabilityMassFunction_T
TRADEMGENTripTypeProbabilityMassFunction_T`

Define the probability mass function type of trip type.

Definition at line 40 of file [DemandCharacteristicsTypes.hpp](#).

23.3.1.26 `typedef CategoricalAttributeLite<stdair::DayDuration_T>
TRADEMGENTayDurationProbabilityMass_T`

Define the stay duration probability mass.

Definition at line 43 of file [DemandCharacteristicsTypes.hpp](#).

23.3.1.27 `typedef StayDurationProbabilityMass_T::ProbabilityMassFunction_T
TRADEMGENTayDurationProbabilityMassFunction_T`

Define the probability mass function type of stay duration.

Definition at line 46 of file [DemandCharacteristicsTypes.hpp](#).

23.3.1.28 `typedef CategoricalAttributeLite<stdair::FrequentFlyer_T>
TRADEMGENTrequentFlyerProbabilityMass_T`

Define the frequent flyer probability mass.

Definition at line 49 of file [DemandCharacteristicsTypes.hpp](#).

23.3.1.29 `typedef FrequentFlyerProbabilityMass_T::ProbabilityMassFunction_T
TRADEMGENTrequentFlyerProbabilityMassFunction_T`

Define the probability mass function type of frequent flyer.

Definition at line 52 of file [DemandCharacteristicsTypes.hpp](#).

23.3.1.30 `typedef ContinuousAttributeLite<stdair::IntDuration_T>
TRADEMGENTPreferredDepartureTimeCumulativeDistribution_T`

Define the preferred departure time cumulative distribution.

Definition at line 55 of file [DemandCharacteristicsTypes.hpp](#).

23.3.1.31 `typedef PreferredDepartureTimeCumulativeDistribution_T::ContinuousDistribution_T
TRADEMGENTPreferredDepartureTimeContinuousDistribution_T`

Define the preferred departure time continuous distribution.

Definition at line 58 of file [DemandCharacteristicsTypes.hpp](#).

23.3.1.32 `typedef ContinuousAttributeLite<stdair::PriceValue_T>
TRADEMGENT::ValueOfTimeCumulativeDistribution_T`

Define the value of time cumulative distribution.

Definition at line 61 of file [DemandCharacteristicsTypes.hpp](#).

23.3.1.33 `typedef ValueOfTimeCumulativeDistribution_T::ContinuousDistribution_T
TRADEMGENT::ValueOfTimeContinuousDistribution_T`

Define the value of time continuous distribution.

Definition at line 64 of file [DemandCharacteristicsTypes.hpp](#).

23.3.1.34 `typedef ContinuousAttributeLite<stdair::RealNumber_T>
TRADEMGENT::CumulativeDistribution_T`

Define the FRAT5 pattern type.

Definition at line 67 of file [DemandCharacteristicsTypes.hpp](#).

23.3.1.35 `typedef CumulativeDistribution_T::ContinuousDistribution_T
TRADEMGENT::FRAT5Pattern_T`

Definition at line 68 of file [DemandCharacteristicsTypes.hpp](#).

23.3.1.36 `typedef stdair::Probability_T TRADEMGENT::DictionaryKey_T`

Dictionary key.

Definition at line 16 of file [DictionaryManager.hpp](#).

23.3.1.37 `typedef std::list<DemandStream*> TRADEMGENT::DemandStreamList_T`

Define the airline feature list.

Definition at line 16 of file [DemandStreamTypes.hpp](#).

23.3.1.38 `typedef std::map<const stdair::MapKey_T, DemandStream*>
TRADEMGENT::DemandStreamMap_T`

Define the airline feature map.

Definition at line 22 of file [DemandStreamTypes.hpp](#).

23.3.1.39 `typedef std::list<std::string> TRADEMGENT::DBParamsNameList_T`

List of names for a given (geographical) dbparams.

Definition at line 17 of file [DBParams.hpp](#).

23.3.1.40 `typedef boost::shared_ptr<TRADEMGEN_Service> TRADEMGEN::TRADEMGEN_ServicePtr_T`

(Smart) Pointer on the TraDemGen service handler.

Definition at line 17 of file [TRADEMGEN_Types.hpp](#).

23.3.2 Function Documentation

23.3.2.1 `stdair::BaseGenerator_T TRADEMGEN::DEFAULT_BASE_GENERATOR (stdair::DEFAULT_RANDOM_SEED)`

Default base generator.

23.3.2.2 `stdair::UniformGenerator_T TRADEMGEN::DEFAULT_UNIFORM_GENERATOR (DEFAULT_BASE_GENERATOR, DEFAULT_UNIFORM_REAL_DISTRIBUTION)`

Default uniform variate generator.

23.3.2.3 `void TRADEMGEN::stat_display (std::ostream & oStream, const stat_acc_type & iStatAcc)`

Display the statistics held by the dedicated accumulator.

Definition at line 47 of file [pytrademgen.cpp](#).

Referenced by [TRADEMGEN::Trademgener::trademgen\(\)](#).

23.3.3 Variable Documentation

23.3.3.1 `const POSProbabilityMassFunction_T TRADEMGEN::DEFAULT_POS_PROBALILITY_MASS`

Initial value:

```
DefaultMap::createPOSProbMass ()
```

Default name for the [TRADEMGEN_Service](#). Default PoS probability mass.

Default PoS probability mass.

Definition at line 16 of file [BasConst.cpp](#).

23.3.3.2 `const stdair::FloatDuration_T TRADEMGEN::DEFAULT_LAST_LOWER_BOUND_ARRIVAL_PATTERN = -1`

Default last lower bound of daily rate interval in arrival pattern.

Definition at line 35 of file [BasConst.cpp](#).

Referenced by [TRADEMGEN::DemandStream::generateTimeOfRequestPoissonProcess\(\)](#).

23.3.3.3 `const FRAT5Pattern_T TRADEMGEN::DEFAULT_FRAT5_PATTERN = DefaultMap::createFRAT5Pattern()`

Default FRAT5 pattern.

Definition at line 38 of file [BasConst.cpp](#).

23.3.3.4 `const double TRADEMGEN::DEFAULT_MAX_ADVANCE_PURCHASE = 330.0`

Default MAX Advance Purchase.

Definition at line 75 of file [BasConst.cpp](#).

23.3.3.5 `const stdair::UniformDistribution_T TRADEMGEN::DEFAULT_UNIFORM_REAL_DISTRIBUTION`

Default random uniform real distribution.

Definition at line 81 of file [BasConst.cpp](#).

23.3.3.6 `stdair::BaseGenerator_T TRADEMGEN::DEFAULT_BASE_GENERATOR`

Default base generator. Just here to initialise objects (e.g., `stdair::RandomGeneration`) with default generator. They are then replaced by a generator, for which the state can better be tracked/stored.

23.3.3.7 `stdair::UniformGenerator_T TRADEMGEN::DEFAULT_UNIFORM_GENERATOR`

Default uniform generator. Just here to initialise objects (e.g., `stdair::RandomGeneration`) with default generator. They are then replaced by a generator, for which the state can better be tracked/stored.

23.4 TRADEMGEN::DemandParserHelper Namespace Reference

Classes

- struct [ParserSemanticAction](#)
- struct [storePrefDepDateRangeStart](#)
- struct [storePrefDepDateRangeEnd](#)
- struct [storeDow](#)
- struct [storeOrigin](#)
- struct [storeDestination](#)
- struct [storePrefCabin](#)
- struct [storeDemandMean](#)
- struct [storeDemandStdDev](#)
- struct [storeDemandChangeFeeProb](#)
- struct [storeDemandChangeFeeDisutility](#)
- struct [storeDemandNonRefundableProb](#)
- struct [storeDemandNonRefundableDisutility](#)
- struct [storePosCode](#)
- struct [storePosProbMass](#)
- struct [storeChannelCode](#)

- struct [storeChannelProbMass](#)
- struct [storeTripCode](#)
- struct [storeTripProbMass](#)
- struct [storeStayCode](#)
- struct [storeStayProbMass](#)
- struct [storeFFCode](#)
- struct [storeFFProbMass](#)
- struct [storePrefDepTime](#)
- struct [storePrefDepTimeProbMass](#)
- struct [storeWTP](#)
- struct [storeTimeValue](#)
- struct [storeTimeValueProbMass](#)
- struct [storeDTD](#)
- struct [storeDTDProbMass](#)
- struct [doEndDemand](#)
- struct [DemandParser](#)

Functions

- [repeat_p_t airline_code_p](#) ([chset_t](#)("0-9A-Z").derived(), 2, 3)
- [bounded1_4_p_t flight_number_p](#) ([uint1_4_p](#).derived(), 0u, 9999u)
- [bounded4_p_t year_p](#) ([uint4_p](#).derived(), 2000u, 2099u)
- [bounded2_p_t month_p](#) ([uint2_p](#).derived(), 1u, 12u)
- [bounded2_p_t day_p](#) ([uint2_p](#).derived(), 1u, 31u)
- [repeat_p_t dow_p](#) ([chset_t](#)("0-1").derived().derived(), 7, 7)
- [repeat_p_t airport_p](#) ([chset_t](#)("0-9A-Z").derived(), 3, 3)
- [bounded1_2_p_t hours_p](#) ([uint1_2_p](#).derived(), 0u, 23u)
- [bounded2_p_t minutes_p](#) ([uint2_p](#).derived(), 0u, 59u)
- [bounded2_p_t seconds_p](#) ([uint2_p](#).derived(), 0u, 59u)
- [chset_t cabin_code_p](#) ("A-Z")
- [chset_t passenger_type_p](#) ("A-Z")
- [chset_t ff_type_p](#) ("A-Z")
- [repeat_p_t class_code_list_p](#) ([chset_t](#)("A-Z").derived(), 1, 26)
- [bounded1_3_p_t stay_duration_p](#) ([uint1_3_p](#).derived(), 0u, 999u)

Variables

- [int1_p_t int1_p](#)
- [uint2_p_t uint2_p](#)
- [uint1_2_p_t uint1_2_p](#)
- [uint1_3_p_t uint1_3_p](#)
- [uint4_p_t uint4_p](#)
- [uint1_4_p_t uint1_4_p](#)
- [int1_p_t family_code_p](#)

23.4.1 Function Documentation

23.4.1.1 [repeat_p_t TRADEMGEN::DemandParserHelper::airline_code_p](#) ([chset_t](#)("0-9A-Z").derived(), 2, 3)

Airline Code Parser: [repeat_p](#)(2,3)[[chset_p](#)("0-9A-Z")]

**23.4.1.2 bounded1_4_p_t TRADEMGEN::DemandParserHelper::flight_number_p (uint1_4_p.
derived(), 0u, 9999u)**

Flight Number Parser: limit_d(0u, 9999u)[uint1_4_p]

**23.4.1.3 bounded4_p_t TRADEMGEN::DemandParserHelper::year_p (uint4_p.
derived(), 2000u, 2099u)**

Year Parser: limit_d(2000u, 2099u)[uint4_p]

Referenced by [TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT
>::definition\(\)](#).**23.4.1.4 bounded2_p_t TRADEMGEN::DemandParserHelper::month_p (uint2_p.
derived(), 1u, 12u)**

Month Parser: limit_d(1u, 12u)[uint2_p]

Referenced by [TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT
>::definition\(\)](#).**23.4.1.5 bounded2_p_t TRADEMGEN::DemandParserHelper::day_p (uint2_p.
derived(), 1u, 31u)**

Day Parser: limit_d(1u, 31u)[uint2_p]

Referenced by [TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT
>::definition\(\)](#).**23.4.1.6 repeat_p_t TRADEMGEN::DemandParserHelper::dow_p
(chset_t("0-1").derived().derived(), 7, 7)**

DOW (Day-Of-the-Week) Parser: repeat_p(7)[chset_p("0-1")]

Referenced by [TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT
>::definition\(\)](#).**23.4.1.7 repeat_p_t TRADEMGEN::DemandParserHelper::airport_p
(chset_t("0-9A-Z").derived(), 3, 3)**

Airport Parser: repeat_p(3)[chset_p("0-9A-Z")]

Referenced by [TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT
>::definition\(\)](#).**23.4.1.8 bounded1_2_p_t TRADEMGEN::DemandParserHelper::hours_p (uint1_2_p.
derived(), 0u, 23u)**

Hour Parser: limit_d(0u, 23u)[uint2_p]

Referenced by [TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT
>::definition\(\)](#).

23.4.1.9 bounded2_p_t TRADEMGEN::DemandParserHelper::minutes_p (uint2_p. *derived()*, 0u, 59u)

Minute Parser: limit_d(0u, 59u)[uint2_p]

Referenced by [TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT >::definition\(\)](#).

23.4.1.10 bounded2_p_t TRADEMGEN::DemandParserHelper::seconds_p (uint2_p. *derived()*, 0u, 59u)

Second Parser: limit_d(0u, 59u)[uint2_p]

Referenced by [TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT >::definition\(\)](#).

23.4.1.11 chset_t TRADEMGEN::DemandParserHelper::cabin_code_p ("A-Z")

Cabin code parser: chset_p("A-Z")

Referenced by [TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT >::definition\(\)](#).

23.4.1.12 chset_t TRADEMGEN::DemandParserHelper::passenger_type_p ("A-Z")

Passenger type parser: chset_p("A-Z")

23.4.1.13 chset_t TRADEMGEN::DemandParserHelper::ff_type_p ("A-Z")

Frequent flyer type parser: chset_p("A-Z")

Referenced by [TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT >::definition\(\)](#).

23.4.1.14 repeat_p_t TRADEMGEN::DemandParserHelper::class_code_list_p (chset_t("A-Z").*derived()*, 1, 26)

Class Code List Parser: repeat_p(1,26)[chset_p("A-Z")]

23.4.1.15 bounded1_3_p_t TRADEMGEN::DemandParserHelper::stay_duration_p (uint1_3_p. *derived()*, 0u, 999u)

Stay duration Parser: limit_d(0u, 999u)[uint3_p]

Referenced by [TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT >::definition\(\)](#).

23.4.2 Variable Documentation

23.4.2.1 `int1_p_t` TRADEMGEN::DemandParserHelper::int1_p

1-digit-integer parser

Definition at line 497 of file [DemandParserHelper.cpp](#).

23.4.2.2 `uint2_p_t` TRADEMGEN::DemandParserHelper::uint2_p

2-digit-integer parser

Definition at line 500 of file [DemandParserHelper.cpp](#).

23.4.2.3 `uint1_2_p_t` TRADEMGEN::DemandParserHelper::uint1_2_p

Up-to-2-digit-integer parser

Definition at line 503 of file [DemandParserHelper.cpp](#).

23.4.2.4 `uint1_3_p_t` TRADEMGEN::DemandParserHelper::uint1_3_p

Up-to-3-digit-integer parser

Definition at line 506 of file [DemandParserHelper.cpp](#).

23.4.2.5 `uint4_p_t` TRADEMGEN::DemandParserHelper::uint4_p

4-digit-integer parser

Definition at line 509 of file [DemandParserHelper.cpp](#).

23.4.2.6 `uint1_4_p_t` TRADEMGEN::DemandParserHelper::uint1_4_p

Up-to-4-digit-integer parser

Definition at line 512 of file [DemandParserHelper.cpp](#).

23.4.2.7 `int1_p_t` TRADEMGEN::DemandParserHelper::family_code_p

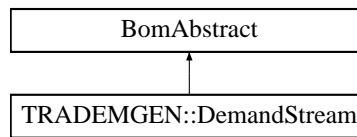
Family code parser

Definition at line 554 of file [DemandParserHelper.cpp](#).

24 Class Documentation

24.1 BomAbstract Class Reference

Inheritance diagram for BomAbstract::



The documentation for this class was generated from the following file:

- trademgen/bom/[DemandStream.hpp](#)

24.2 TRADEMGEN::BomDisplay Class Reference

Utility class to display TraDemGen objects with a pretty format.

```
#include <trademgen/bom/BomDisplay.hpp>
```

Static Public Member Functions

- static std::string [csvDisplay](#) (const SEVMGR::SEVMGR_ServicePtr_T)
- static void [csvDisplay](#) (std::ostream &, const [DemandStream](#) &)

24.2.1 Detailed Description

Utility class to display TraDemGen objects with a pretty format.

Definition at line 23 of file [BomDisplay.hpp](#).

24.2.2 Member Function Documentation

24.2.2.1 std::string TRADEMGEN::BomDisplay::csvDisplay (const SEVMGR::SEVMGR_ServicePtr_T iSEVMGR_ServicePtr) [static]

Recursively display (dump in the underlying output log stream) the objects of the BOM tree.

Parameters:

- std::ostream&* Output stream in which the BOM tree should be logged/dumped.
- const SEVMGR::SEVMGR_ServicePtr_T* Pointer on the SEvMgr service handler to display the queue key and to obtain the demand stream list.

Definition at line 45 of file [BomDisplay.cpp](#).

24.2.2.2 void TRADEMGEN::BomDisplay::csvDisplay (std::ostream & oStream, const DemandStream & iDemandStream) [static]

Recursively display (dump in the underlying output log stream) the objects of the BOM tree.

Parameters:

- std::ostream&* Output stream in which the BOM tree should be logged/dumped.

const [DemandStream](#)& Root of the BOM tree to be displayed.

Definition at line 87 of file [BomDisplay.cpp](#).

References [TRADEMGEN::DemandStream::display\(\)](#).

The documentation for this class was generated from the following files:

- trademgen/bom/[BomDisplay.hpp](#)
- trademgen/bom/[BomDisplay.cpp](#)

24.3 stdair::CategoricalAttribute< T > Struct Template Reference

Class modeling the distribution of values that can be taken by a categorical attribute.

```
#include <trademgen/basic/CategoricalAttribute.hpp>
```

Public Types

- typedef std::map< T, DictionaryKey_T > [ProbabilityMassFunction_T](#)
- typedef std::map< DictionaryKey_T, T > [InverseCumulativeDistribution_T](#)

Public Member Functions

- const T & [getValue](#) (const Probability_T &iCumulativeProbability) const
- const std::string [displayProbabilityMassFunction](#) () const
- const std::string [displayInverseCumulativeDistribution](#) () const
- [CategoricalAttribute](#) (const [ProbabilityMassFunction_T](#) &iProbabilityMassFunction)
- [CategoricalAttribute](#) ()
- [CategoricalAttribute](#) (const [CategoricalAttribute](#) &iCategoricalAttribute)
- virtual [~CategoricalAttribute](#) ()
- void [determineInverseCumulativeDistributionFromProbabilityMassFunction](#) ()

24.3.1 Detailed Description

template<typename T> struct stdair::CategoricalAttribute< T >

Class modeling the distribution of values that can be taken by a categorical attribute.

Definition at line 21 of file [CategoricalAttribute.hpp](#).

24.3.2 Member Typedef Documentation

**24.3.2.1 template<typename T > typedef std::map<T, DictionaryKey_T>
stdair::CategoricalAttribute< T >::ProbabilityMassFunction_T**

Define the probability mass function type.

Definition at line 28 of file [CategoricalAttribute.hpp](#).

24.3.2.2 `template<typename T > typedef std::map<DictionaryKey_T, T>
stdair::CategoricalAttribute< T >::InverseCumulativeDistribution_T`

Define the inverse cumulative distribution type.

Definition at line 33 of file [CategoricalAttribute.hpp](#).

24.3.3 Constructor & Destructor Documentation

24.3.3.1 `template<typename T > stdair::CategoricalAttribute< T >::CategoricalAttribute (const
ProbabilityMassFunction_T & iProbabilityMassFunction) [inline]`

Main constructor.

Definition at line 129 of file [CategoricalAttribute.hpp](#).

References [stdair::CategoricalAttribute< T >::determineInverseCumulativeDistributionFromProbabilityMassFunction\(\)](#).

24.3.3.2 `template<typename T > stdair::CategoricalAttribute< T >::CategoricalAttribute ()
[inline]`

Default constructor.

Definition at line 137 of file [CategoricalAttribute.hpp](#).

24.3.3.3 `template<typename T > stdair::CategoricalAttribute< T >::CategoricalAttribute (const
CategoricalAttribute< T > & iCategoricalAttribute) [inline]`

Copy constructor.

Definition at line 142 of file [CategoricalAttribute.hpp](#).

References [stdair::CategoricalAttribute< T >::determineInverseCumulativeDistributionFromProbabilityMassFunction\(\)](#).

24.3.3.4 `template<typename T > virtual stdair::CategoricalAttribute< T
>::~~CategoricalAttribute () [inline, virtual]`

Destructor.

Definition at line 150 of file [CategoricalAttribute.hpp](#).

24.3.4 Member Function Documentation

24.3.4.1 `template<typename T > const T& stdair::CategoricalAttribute< T >::getValue (const
Probability_T & iCumulativeProbability) const [inline]`

Get value from inverse cumulative distribution.

Definition at line 67 of file [CategoricalAttribute.hpp](#).

References [stdair::CategoricalAttribute< T >::displayInverseCumulativeDistribution\(\)](#).

24.3.4.2 `template<typename T > const std::string stdair::CategoricalAttribute< T >::displayProbabilityMassFunction () const [inline]`

Display probability mass function.

Definition at line 91 of file [CategoricalAttribute.hpp](#).

24.3.4.3 `template<typename T > const std::string stdair::CategoricalAttribute< T >::displayInverseCumulativeDistribution () const [inline]`

Display inverse cumulative distribution.

Definition at line 111 of file [CategoricalAttribute.hpp](#).

Referenced by [stdair::CategoricalAttribute< T >::getValue\(\)](#).

24.3.4.4 `template<typename T > void stdair::CategoricalAttribute< T >::determineInverseCumulativeDistributionFromProbabilityMassFunction () [inline]`

Determine inverse cumulative distribution from probability mass function (initialisation).

Definition at line 157 of file [CategoricalAttribute.hpp](#).

Referenced by [stdair::CategoricalAttribute< T >::CategoricalAttribute\(\)](#).

The documentation for this struct was generated from the following file:

- [trademgen/basic/CategoricalAttribute.hpp](#)

24.4 TRADEMGEN::CategoricalAttributeLite< T > Struct Template Reference

Class modeling the distribution of values that can be taken by a categorical attribute.

```
#include <trademgen/basic/CategoricalAttributeLite.hpp>
```

Public Types

- `typedef std::map< T, stdair::Probability_T > ProbabilityMassFunction_T`

Public Member Functions

- `const T & getValue (const stdair::Probability_T &iCumulativeProbability) const`
- `bool checkValue (const T &iValue) const`
- `const std::string displayProbabilityMass () const`
- `CategoricalAttributeLite (const ProbabilityMassFunction_T &iValueMap)`
- `CategoricalAttributeLite ()`
- `CategoricalAttributeLite (const CategoricalAttributeLite &iCAL)`
- `CategoricalAttributeLite & operator= (const CategoricalAttributeLite &iCAL)`
- `virtual ~CategoricalAttributeLite ()`

24.4.1 Detailed Description

template<typename T> struct TRADEMGEN::CategoricalAttributeLite< T >

Class modeling the distribution of values that can be taken by a categorical attribute.

Definition at line 27 of file [CategoricalAttributeLite.hpp](#).

24.4.2 Member Typedef Documentation

**24.4.2.1 template<typename T> typedef std::map<T, stdair::Probability_T>
TRADEMGENT::CategoricalAttributeLite< T >::ProbabilityMassFunction_T**

Type for the probability mass function.

Definition at line 33 of file [CategoricalAttributeLite.hpp](#).

24.4.3 Constructor & Destructor Documentation

**24.4.3.1 template<typename T> TRADEMGENT::CategoricalAttributeLite< T
>::CategoricalAttributeLite (const ProbabilityMassFunction_T & *iValueMap*)
[inline]**

Main constructor.

Definition at line 95 of file [CategoricalAttributeLite.hpp](#).

**24.4.3.2 template<typename T> TRADEMGENT::CategoricalAttributeLite< T
>::CategoricalAttributeLite () [inline]**

Default constructor.

Definition at line 103 of file [CategoricalAttributeLite.hpp](#).

**24.4.3.3 template<typename T> TRADEMGENT::CategoricalAttributeLite< T
>::CategoricalAttributeLite (const CategoricalAttributeLite< T > & *iCAL*) [inline]**

Copy constructor.

Definition at line 109 of file [CategoricalAttributeLite.hpp](#).

**24.4.3.4 template<typename T> virtual TRADEMGENT::CategoricalAttributeLite< T
>::~~CategoricalAttributeLite () [inline, virtual]**

Destructor.

Definition at line 128 of file [CategoricalAttributeLite.hpp](#).

24.4.4 Member Function Documentation

24.4.4.1 `template<typename T> const T& TRADEMGEN::CategoricalAttributeLite< T >::getValue (const stdair::Probability_T & iCumulativeProbability) const [inline]`

Get value from inverse cumulative distribution.

Definition at line 41 of file [CategoricalAttributeLite.hpp](#).

Referenced by [TRADEMGEN::DemandStream::generateChannel\(\)](#), [TRADEMGEN::DemandStream::generateFrequentFlyer\(\)](#), [TRADEMGEN::DemandStream::generateStayDuration\(\)](#), [TRADEMGEN::DemandStream::generateTripType\(\)](#), and [TRADEMGEN::DemandCharacteristics::getPOSValue\(\)](#).

24.4.4.2 `template<typename T> bool TRADEMGEN::CategoricalAttributeLite< T >::checkValue (const T & iValue) const [inline]`

Check if a value belongs to the value list.

Definition at line 61 of file [CategoricalAttributeLite.hpp](#).

Referenced by [TRADEMGEN::DemandCharacteristics::checkPOSValue\(\)](#).

24.4.4.3 `template<typename T> const std::string TRADEMGEN::CategoricalAttributeLite< T >::displayProbabilityMass () const [inline]`

Display probability mass function.

Definition at line 76 of file [CategoricalAttributeLite.hpp](#).

Referenced by [TRADEMGEN::DemandCharacteristics::describe\(\)](#), [TRADEMGEN::DemandStream::display\(\)](#), and [TRADEMGEN::CategoricalAttributeLite< stdair::TripType_T >::getValue\(\)](#).

24.4.4.4 `template<typename T> CategoricalAttributeLite& TRADEMGEN::CategoricalAttributeLite< T >::operator= (const CategoricalAttributeLite< T > & iCAL) [inline]`

Copy operator.

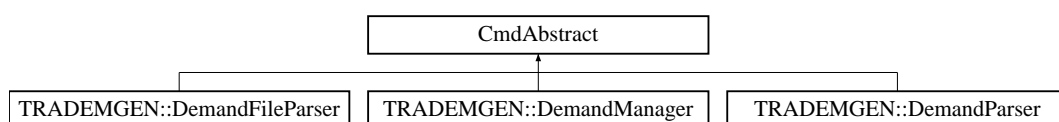
Definition at line 118 of file [CategoricalAttributeLite.hpp](#).

The documentation for this struct was generated from the following file:

- [trademgen/basic/CategoricalAttributeLite.hpp](#)

24.5 CmdAbstract Class Reference

Inheritance diagram for CmdAbstract::



The documentation for this class was generated from the following files:

- trademgen/command/[DemandParser.hpp](#)
- trademgen/command/[DemandManager.hpp](#)
- trademgen/command/[DemandParserHelper.hpp](#)

24.6 TRADEMGEN::ContinuousAttribute< T > Struct Template Reference

```
#include <trademgen/basic/ContinuousAttribute.hpp>
```

Public Types

- typedef std::multimap< T, [DictionaryKey_T](#) > [ContinuousDistribution_T](#)
- typedef std::multimap< [DictionaryKey_T](#), T > [ContinuousInverseDistribution_T](#)

Public Member Functions

- const T [getValue](#) (const stdair::Probability_T &iCumulativeProbability) const
- const std::string [displayCumulativeDistribution](#) () const
- const std::string [displayInverseCumulativeDistribution](#) () const
- [ContinuousAttribute](#) ()
- [ContinuousAttribute](#) (const [ContinuousDistribution_T](#) &iCumulativeDistribution)
- [ContinuousAttribute](#) (const [ContinuousAttribute](#) &iContinuousAttribute)
- virtual [~ContinuousAttribute](#) ()
- void [determineInverseCumulativeDistributionFromCumulativeDistribution](#) ()

24.6.1 Detailed Description

```
template<class T> struct TRADEMGEN::ContinuousAttribute< T >
```

Class modeling the distribution of values that can be taken by a continuous attribute.

Definition at line 21 of file [ContinuousAttribute.hpp](#).

24.6.2 Member Typedef Documentation

24.6.2.1 `template<class T > typedef std::multimap<T, DictionaryKey_T>
TRADEMGEN::ContinuousAttribute< T >::ContinuousDistribution_T`

Definition at line 26 of file [ContinuousAttribute.hpp](#).

24.6.2.2 `template<class T > typedef std::multimap<DictionaryKey_T, T>
TRADEMGEN::ContinuousAttribute< T >::ContinuousInverseDistribution_T`

Definition at line 27 of file [ContinuousAttribute.hpp](#).

24.6.3 Constructor & Destructor Documentation

24.6.3.1 `template<class T > TRADEMGEN::ContinuousAttribute< T >::ContinuousAttribute () [inline]`

Constructor by default

Definition at line 113 of file [ContinuousAttribute.hpp](#).

24.6.3.2 `template<class T > TRADEMGEN::ContinuousAttribute< T >::ContinuousAttribute (const ContinuousDistribution_T & iCumulativeDistribution) [inline]`

Constructor

Definition at line 116 of file [ContinuousAttribute.hpp](#).

References [TRADEMGENT::ContinuousAttribute< T >::determineInverseCumulativeDistributionFromCumulativeDistribution](#)

24.6.3.3 `template<class T > TRADEMGEN::ContinuousAttribute< T >::ContinuousAttribute (const ContinuousAttribute< T > & iContinuousAttribute) [inline]`

Copy constructor

Definition at line 122 of file [ContinuousAttribute.hpp](#).

24.6.3.4 `template<class T > virtual TRADEMGEN::ContinuousAttribute< T >::~~ContinuousAttribute () [inline, virtual]`

Destructor

Definition at line 128 of file [ContinuousAttribute.hpp](#).

24.6.4 Member Function Documentation

24.6.4.1 `template<class T > const T TRADEMGEN::ContinuousAttribute< T >::getValue (const stdair::Probability_T & iCumulativeProbability) const [inline]`

Get value from inverse cumulative distribution.

Definition at line 52 of file [ContinuousAttribute.hpp](#).

References [TRADEMGENT::DictionaryManager::keyToValue\(\)](#), and [TRADEMGENT::DictionaryManager::valueToKey\(\)](#).

24.6.4.2 `template<class T > const std::string TRADEMGEN::ContinuousAttribute< T >::displayCumulativeDistribution () const [inline]`

Display cumulative distribution

Definition at line 83 of file [ContinuousAttribute.hpp](#).

References [TRADEMGENT::DictionaryManager::keyToValue\(\)](#).

24.6.4.3 `template<class T > const std::string TRADEMGEN::ContinuousAttribute< T >::displayInverseCumulativeDistribution () const [inline]`

Display inverse cumulative distribution

Definition at line 99 of file [ContinuousAttribute.hpp](#).

References [TRADEMGENT::DictionaryManager::keyToValue\(\)](#).

24.6.4.4 `template<class T > void TRADEMGEN::ContinuousAttribute< T >::determineInverseCumulativeDistributionFromCumulativeDistribution () [inline]`

Determine inverse cumulative distribution from cumulative distribution (initialisation).

Definition at line 132 of file [ContinuousAttribute.hpp](#).

Referenced by [TRADEMGENT::ContinuousAttribute< T >::ContinuousAttribute\(\)](#).

The documentation for this struct was generated from the following file:

- [trademgen/basic/ContinuousAttribute.hpp](#)

24.7 TRADEMGEN::ContinuousAttributeLite< T > Struct Template Reference

Class modeling the distribution of values that can be taken by a continuous attribute.

```
#include <trademgen/basic/ContinuousAttributeLite.hpp>
```

Public Types

- `typedef std::map< T, stdair::Probability_T > ContinuousDistribution_T`

Public Member Functions

- `const T getValue (const stdair::Probability_T &iCumulativeProbability) const`
- `const double getDerivativeValue (const T iKey) const`
- `const T getUpperBound (const T iKey) const`
- `const std::string displayCumulativeDistribution () const`
- `ContinuousAttributeLite (const ContinuousDistribution_T &iValueMap)`
- `ContinuousAttributeLite (const ContinuousAttributeLite &iCAL)`
- `ContinuousAttributeLite & operator= (const ContinuousAttributeLite &iCAL)`
- `virtual ~ContinuousAttributeLite ()`

24.7.1 Detailed Description

`template<typename T> struct TRADEMGEN::ContinuousAttributeLite< T >`

Class modeling the distribution of values that can be taken by a continuous attribute.

Definition at line 26 of file [ContinuousAttributeLite.hpp](#).

24.7.2 Member Typedef Documentation

24.7.2.1 `template<typename T> typedef std::map<T, stdair::Probability_T>
TRADEMGENT::ContinuousAttributeLite< T >::ContinuousDistribution_T`

Type for the probability mass function.

Definition at line 32 of file [ContinuousAttributeLite.hpp](#).

24.7.3 Constructor & Destructor Documentation

24.7.3.1 `template<typename T> TRADEMGENT::ContinuousAttributeLite< T
>::ContinuousAttributeLite (const ContinuousDistribution_T & iValueMap) [inline]`

Constructor.

Definition at line 157 of file [ContinuousAttributeLite.hpp](#).

24.7.3.2 `template<typename T> TRADEMGENT::ContinuousAttributeLite< T
>::ContinuousAttributeLite (const ContinuousAttributeLite< T > & iCAL) [inline]`

Copy constructor.

Definition at line 165 of file [ContinuousAttributeLite.hpp](#).

24.7.3.3 `template<typename T> virtual TRADEMGENT::ContinuousAttributeLite< T
>::~~ContinuousAttributeLite () [inline, virtual]`

Destructor.

Definition at line 184 of file [ContinuousAttributeLite.hpp](#).

24.7.4 Member Function Documentation

24.7.4.1 `template<typename T> const T TRADEMGENT::ContinuousAttributeLite< T
>::getValue (const stdair::Probability_T & iCumulativeProbability) const [inline]`

Get value from inverse cumulative distribution.

Definition at line 39 of file [ContinuousAttributeLite.hpp](#).

Referenced by [TRADEMGENT::DemandStream::generateTimeOfRequestPoissonProcess\(\)](#), [TRADEMGENT::DemandStream::generateTimeOfRequestStatisticsOrder\(\)](#), [TRADEMGENT::DemandStream::generateValueOfTime\(\)](#), and [TRADEMGENT::DemandStream::generateWTP\(\)](#).

24.7.4.2 `template<typename T> const double TRADEMGENT::ContinuousAttributeLite< T
>::getDerivativeValue (const T iKey) const [inline]`

Get the value of the derivative function in a key point.

Definition at line 82 of file [ContinuousAttributeLite.hpp](#).

Referenced by [TRADEMGENT::DemandStream::generateTimeOfRequestPoissonProcess\(\)](#).

24.7.4.3 `template<typename T> const T TRADEMGEN::ContinuousAttributeLite< T >::getUpperBound (const T iKey) const [inline]`

Get the upper bound.

Definition at line 116 of file [ContinuousAttributeLite.hpp](#).

Referenced by [TRADEMGENT::DemandStream::generateTimeOfRequestPoissonProcess\(\)](#).

24.7.4.4 `template<typename T> const std::string TRADEMGEN::ContinuousAttributeLite< T >::displayCumulativeDistribution () const [inline]`

Display cumulative distribution.

Definition at line 135 of file [ContinuousAttributeLite.hpp](#).

Referenced by [TRADEMGENT::DemandCharacteristics::describe\(\)](#).

24.7.4.5 `template<typename T> ContinuousAttributeLite& TRADEMGEN::ContinuousAttributeLite< T >::operator= (const ContinuousAttributeLite< T > &iCAL) [inline]`

Copy operator.

Definition at line 174 of file [ContinuousAttributeLite.hpp](#).

The documentation for this struct was generated from the following file:

- [trademgen/basic/ContinuousAttributeLite.hpp](#)

24.8 TRADEMGEN::DBManager Class Reference

```
#include <trademgen/command/DBManager.hpp>
```

Static Public Member Functions

- static void [updateAirlineInDB](#) (stdair::DBSession_T &, const stdair::AirlineStruct &)
- static bool [retrieveAirline](#) (stdair::DBSession_T &, const stdair::AirlineCode_T &, stdair::AirlineStruct &)
- static void [prepareSelectStatement](#) (stdair::DBSession_T &, stdair::DBRequestStatement_T &, stdair::AirlineStruct &)
- static bool [iterateOnStatement](#) (stdair::DBRequestStatement_T &, stdair::AirlineStruct &, const bool iShouldDoReset)

24.8.1 Detailed Description

Class building the Business Object Model (BOM) from data retrieved from the database.

Definition at line 20 of file [DBManager.hpp](#).

24.8.2 Member Function Documentation

24.8.2.1 void TRADEMGEN::DBManager::updateAirlineInDB (stdair::DBSession_T & *ioSociSession*, const stdair::AirlineStruct & *iAirline*) [static]

Update the fields of the database row corresponding to the given BOM object.

Definition at line 121 of file [DBManager.cpp](#).

24.8.2.2 bool TRADEMGEN::DBManager::retrieveAirline (stdair::DBSession_T & *ioSociSession*, const stdair::AirlineCode_T & *iAirlineCode*, stdair::AirlineStruct & *ioAirline*) [static]

Retrieve, from the (MySQL) database, the row corresponding to the given BOM code, and fill the given BOM object with that retrieved data.

Definition at line 157 of file [DBManager.cpp](#).

References [iterateOnStatement\(\)](#).

24.8.2.3 void TRADEMGEN::DBManager::prepareSelectStatement (stdair::DBSession_T & *ioSociSession*, stdair::DBRequestStatement_T & *ioSelectStatement*, stdair::AirlineStruct & *ioAirline*) [static]

Prepare (parse and put in cache) the SQL statement.

Definition at line 24 of file [DBManager.cpp](#).

24.8.2.4 bool TRADEMGEN::DBManager::iterateOnStatement (stdair::DBRequestStatement_T & *ioStatement*, stdair::AirlineStruct & *ioAirline*, const bool *iShouldDoReset*) [static]

Iterate on the SQL statement.

The SQL has to be already prepared. const bool Tells whether the Airline object should be reset.

Definition at line 97 of file [DBManager.cpp](#).

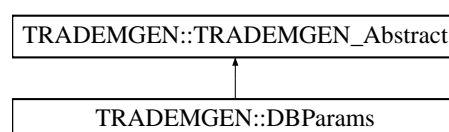
Referenced by [retrieveAirline\(\)](#).

The documentation for this class was generated from the following files:

- [trademgen/command/DBManager.hpp](#)
- [trademgen/command/DBManager.cpp](#)

24.9 TRADEMGEN::DBParams Struct Reference

#include <trademgen/DBParams.hpp> Inheritance diagram for TRADEMGEN::DBParams::



Public Member Functions

- std::string [getUser](#) () const
- std::string [getPassword](#) () const
- std::string [getHost](#) () const
- std::string [getPort](#) () const
- std::string [getDBName](#) () const
- void [setUser](#) (const std::string &iUser)
- void [setPassword](#) (const std::string &iPasswd)
- void [setHost](#) (const std::string &iHost)
- void [setPort](#) (const std::string &iPort)
- void [setDBName](#) (const std::string &iDBName)
- bool [check](#) () const
- void [toStream](#) (std::ostream &ioOut) const
- void [fromStream](#) (std::istream &)
- std::string [toShortString](#) () const
- std::string [toString](#) () const
- [DBParams](#) (const std::string &iDBUser, const std::string &iDBPasswd, const std::string &iDBHost, const std::string &iDBPort, const std::string &iDBName)
- virtual [~DBParams](#) ()

24.9.1 Detailed Description

Structure modelling a (geographical) dbparams.

Definition at line 21 of file [DBParams.hpp](#).

24.9.2 Constructor & Destructor Documentation
24.9.2.1 TRADEMGEN::DBParams::DBParams (const std::string &iDBUser, const std::string &iDBPasswd, const std::string &iDBHost, const std::string &iDBPort, const std::string &iDBName) [inline]

Main Constructor.

Definition at line 119 of file [DBParams.hpp](#).

24.9.2.2 virtual TRADEMGEN::DBParams::~DBParams () [inline, virtual]

Default Constructor. Default copy constructor. Destructor.

Definition at line 132 of file [DBParams.hpp](#).

24.9.3 Member Function Documentation
24.9.3.1 std::string TRADEMGEN::DBParams::getUser () const [inline]

Get the database user name.

Definition at line 25 of file [DBParams.hpp](#).

24.9.3.2 `std::string TRADEMGEN::DBParams::getPassword () const [inline]`

Get the database user password.

Definition at line 30 of file [DBParams.hpp](#).

24.9.3.3 `std::string TRADEMGEN::DBParams::getHost () const [inline]`

Get the database host name.

Definition at line 35 of file [DBParams.hpp](#).

24.9.3.4 `std::string TRADEMGEN::DBParams::getPort () const [inline]`

Get the database port number.

Definition at line 40 of file [DBParams.hpp](#).

24.9.3.5 `std::string TRADEMGEN::DBParams::getDBName () const [inline]`

Get the database name.

Definition at line 45 of file [DBParams.hpp](#).

24.9.3.6 `void TRADEMGEN::DBParams::setUser (const std::string & iUser) [inline]`

Set the database user name.

Definition at line 52 of file [DBParams.hpp](#).

24.9.3.7 `void TRADEMGEN::DBParams::setPassword (const std::string & iPasswd) [inline]`

Set the database password.

Definition at line 57 of file [DBParams.hpp](#).

24.9.3.8 `void TRADEMGEN::DBParams::setHost (const std::string & iHost) [inline]`

Set the database host name.

Definition at line 62 of file [DBParams.hpp](#).

24.9.3.9 `void TRADEMGEN::DBParams::setPort (const std::string & iPort) [inline]`

Set the database port number.

Definition at line 67 of file [DBParams.hpp](#).

24.9.3.10 `void TRADEMGEN::DBParams::setDBName (const std::string & iDBName)
[inline]`

Set the database name.

Definition at line 72 of file [DBParams.hpp](#).

24.9.3.11 bool TRADEMGEN::DBParams::check () const [inline]

Check that all the parameters are fine.

Definition at line 80 of file [DBParams.hpp](#).

24.9.3.12 void TRADEMGEN::DBParams::toStream (std::ostream & *ioOut*) const [inline, virtual]

Dump a structure into an output stream.

Parameters:

ostream& the output stream.

Implements [TRADEMGEN::TRADEMGEN_Abstract](#).

Definition at line 93 of file [DBParams.hpp](#).

References [toString\(\)](#).

24.9.3.13 void TRADEMGEN::DBParams::fromStream (std::istream &) [inline, virtual]

Read a structure from an input stream.

Parameters:

istream& the input stream.

Implements [TRADEMGEN::TRADEMGEN_Abstract](#).

Definition at line 99 of file [DBParams.hpp](#).

24.9.3.14 std::string TRADEMGEN::DBParams::toShortString () const [inline]

Get a short display of the [DBParams](#) structure.

Definition at line 103 of file [DBParams.hpp](#).

24.9.3.15 std::string TRADEMGEN::DBParams::toString () const [inline, virtual]

Get the serialised version of the [DBParams](#) structure.

Implements [TRADEMGEN::TRADEMGEN_Abstract](#).

Definition at line 110 of file [DBParams.hpp](#).

Referenced by [toStream\(\)](#).

The documentation for this struct was generated from the following file:

- [trademgen/DBParams.hpp](#)

24.10 TRADEMGEN::DefaultMap Struct Reference

```
#include <trademgen/basic/BasConst_DemandGeneration.hpp>
```

Static Public Member Functions

- static [POSProbabilityMassFunction_T](#) createPOSProbMass ()
- static [FRAT5Pattern_T](#) createFRAT5Pattern ()

24.10.1 Detailed Description

Default PoS probability mass.

Definition at line 24 of file [BasConst_DemandGeneration.hpp](#).

24.10.2 Member Function Documentation

24.10.2.1 [POSProbabilityMassFunction_T](#) TRADEMGEN::DefaultMap::createPOSProbMass () [static]

Default PoS probability mass.

Definition at line 20 of file [BasConst.cpp](#).

24.10.2.2 [FRAT5Pattern_T](#) TRADEMGEN::DefaultMap::createFRAT5Pattern () [static]

Default FRAT5 pattern.

Definition at line 41 of file [BasConst.cpp](#).

The documentation for this struct was generated from the following files:

- trademgen/basic/[BasConst_DemandGeneration.hpp](#)
- trademgen/basic/[BasConst.cpp](#)

24.11 TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT > Struct Template Reference

```
#include <trademgen/command/DemandParserHelper.hpp>
```

Public Member Functions

- [definition](#) ([DemandParser](#) const &self)
- boost::spirit::classic::rule< ScannerT > const & [start](#) () const

Public Attributes

- boost::spirit::classic::rule< ScannerT > [demand_list](#)
- boost::spirit::classic::rule< ScannerT > [not_to_be_parsed](#)
- boost::spirit::classic::rule< ScannerT > [demand](#)

- boost::spirit::classic::rule< ScannerT > [demand_end](#)
- boost::spirit::classic::rule< ScannerT > [pref_dep_date_range](#)
- boost::spirit::classic::rule< ScannerT > [date](#)
- boost::spirit::classic::rule< ScannerT > [dow](#)
- boost::spirit::classic::rule< ScannerT > [origin](#)
- boost::spirit::classic::rule< ScannerT > [destination](#)
- boost::spirit::classic::rule< ScannerT > [pref_cabin](#)
- boost::spirit::classic::rule< ScannerT > [demand_params](#)
- boost::spirit::classic::rule< ScannerT > [pos_dist](#)
- boost::spirit::classic::rule< ScannerT > [pos_pair](#)
- boost::spirit::classic::rule< ScannerT > [pos_code](#)
- boost::spirit::classic::rule< ScannerT > [pos_share](#)
- boost::spirit::classic::rule< ScannerT > [channel_dist](#)
- boost::spirit::classic::rule< ScannerT > [channel_pair](#)
- boost::spirit::classic::rule< ScannerT > [channel_code](#)
- boost::spirit::classic::rule< ScannerT > [channel_share](#)
- boost::spirit::classic::rule< ScannerT > [trip_dist](#)
- boost::spirit::classic::rule< ScannerT > [trip_pair](#)
- boost::spirit::classic::rule< ScannerT > [trip_code](#)
- boost::spirit::classic::rule< ScannerT > [trip_share](#)
- boost::spirit::classic::rule< ScannerT > [stay_dist](#)
- boost::spirit::classic::rule< ScannerT > [stay_pair](#)
- boost::spirit::classic::rule< ScannerT > [stay_share](#)
- boost::spirit::classic::rule< ScannerT > [ff_dist](#)
- boost::spirit::classic::rule< ScannerT > [ff_pair](#)
- boost::spirit::classic::rule< ScannerT > [ff_code](#)
- boost::spirit::classic::rule< ScannerT > [ff_share](#)
- boost::spirit::classic::rule< ScannerT > [change_fees](#)
- boost::spirit::classic::rule< ScannerT > [non_refundable](#)
- boost::spirit::classic::rule< ScannerT > [pref_dep_time_dist](#)
- boost::spirit::classic::rule< ScannerT > [pref_dep_time_pair](#)
- boost::spirit::classic::rule< ScannerT > [pref_dep_time_share](#)
- boost::spirit::classic::rule< ScannerT > [time](#)
- boost::spirit::classic::rule< ScannerT > [wtp](#)
- boost::spirit::classic::rule< ScannerT > [time_value_dist](#)
- boost::spirit::classic::rule< ScannerT > [time_value_pair](#)
- boost::spirit::classic::rule< ScannerT > [time_value_share](#)
- boost::spirit::classic::rule< ScannerT > [dtd_dist](#)
- boost::spirit::classic::rule< ScannerT > [dtd_pair](#)
- boost::spirit::classic::rule< ScannerT > [dtd_share](#)

24.11.1 Detailed Description

template<typename ScannerT> struct TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT >

Definition at line 391 of file [DemandParserHelper.hpp](#).

24.11.2 Constructor & Destructor Documentation

24.11.2.1 `template<typename ScannerT > TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT >::definition (DemandParser const & self) [inline]`

Definition at line 580 of file [DemandParserHelper.cpp](#).

References

TRADEMGEN::DemandParserHelper::DemandParser::_demand,	TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT >::change_
TRADEMGEN::DemandStruct::_itDay,	TRADEMGEN::DemandStruct::_itHours,
TRADEMGEN::DemandStruct::_itMinutes,	TRADEMGEN::DemandStruct::_itMonth,
TRADEMGEN::DemandStruct::_itSeconds,	TRADEMGEN::DemandStruct::_itYear,
TRADEMGEN::DemandParserHelper::DemandParser::_posProbabilityMass,	TRADEMGEN::DemandParserHelper::DemandParser::_sevmgrServicePtr,
TRADEMGEN::DemandParserHelper::DemandParser::_uniformGenerator,	TRADEMGEN::DemandParserHelper::airport_p(),
TRADEMGEN::DemandParserHelper::cabin_	TRADEMGEN::DemandParserHelper::cabin_
code_p(),	TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT >::channel_
fees,	TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT >::channel_
code,	TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT >::channel_
dist,	TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT >::channel_
pair,	TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT >::channel_
share,	TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT >::date,
TRADEMGEN::DemandParserHelper::day_p(),	TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT >::demand,
TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT >::demand_end,	TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT >::demand_list,
TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT >::demand_params,	TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT >::destination,
TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT >::dow,	TRADEMGEN::DemandParserHelper::dow_p(),
TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT >::dtd_dist,	TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT >::dtd_pair,
TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT >::dtd_share,	TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT >::ff_code,
TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT >::ff_dist,	TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT >::ff_
pair,	TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT >::ff_share,
TRADEMGEN::DemandParserHelper::ff_type_p(),	TRADEMGEN::DemandParserHelper::hours_p(),
TRADEMGEN::DemandParserHelper::minutes_p(),	TRADEMGEN::DemandParserHelper::month_
p(),	TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT >::non_
refundable,	TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT >::not_
to_be_parsed,	TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT >::origin,
TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT >::pos_	TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT >::pos_
code,	TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT >::pos_
dist,	TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT >::pos_
pair,	TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT >::pos_
share,	TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT >::pref_
cabin,	TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT >::pref_
dep_date_range,	TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT >::pref_dep_time_dist,
TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT >::pref_dep_time_pair,	TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT >::pref_dep_time_share,
TRADEMGEN::DemandParserHelper::seconds_	TRADEMGEN::DemandParserHelper::seconds_
p(),	TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT >::stay_dist,
TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT >::stay_duration_p(),	

TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT >::stay_
pair, TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT >::stay_
share, TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT >::time,
TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT >::time_value_dist,
TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT >::time_value_pair,
TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT >::time_value_
share, TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT >::trip_
code, TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT >::trip_
dist, TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT >::trip_
pair, TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT >::trip_
share, TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT >::wtp, and
TRADEMGEN::DemandParserHelper::year_p().

24.11.3 Member Function Documentation

24.11.3.1 `template<typename ScannerT > bsc::rule< ScannerT > const &
TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT >::start
() const [inline]`

Entry point of the parser.

Definition at line 839 of file [DemandParserHelper.cpp](#).

References [TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT >::demand_list](#).

24.11.4 Member Data Documentation

24.11.4.1 `template<typename ScannerT > boost::spirit::classic::rule<ScannerT>
TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT
>::demand_list`

Definition at line 395 of file [DemandParserHelper.hpp](#).

Referenced by [TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT >::definition\(\)](#), and [TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT >::start\(\)](#).

24.11.4.2 `template<typename ScannerT > boost::spirit::classic::rule<ScannerT>
TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT
>::not_to_be_parsed`

Definition at line 395 of file [DemandParserHelper.hpp](#).

Referenced by [TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT >::definition\(\)](#).

24.11.4.3 `template<typename ScannerT > boost::spirit::classic::rule<ScannerT>
TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT
>::demand`

Definition at line 395 of file [DemandParserHelper.hpp](#).

Referenced by [TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT
>::definition\(\)](#).

24.11.4.4 `template<typename ScannerT > boost::spirit::classic::rule<ScannerT>
TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT
>::demand_end`

Definition at line 395 of file [DemandParserHelper.hpp](#).

Referenced by [TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT
>::definition\(\)](#).

24.11.4.5 `template<typename ScannerT > boost::spirit::classic::rule<ScannerT>
TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT
>::pref_dep_date_range`

Definition at line 395 of file [DemandParserHelper.hpp](#).

Referenced by [TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT
>::definition\(\)](#).

24.11.4.6 `template<typename ScannerT > boost::spirit::classic::rule<ScannerT>
TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT >::date`

Definition at line 395 of file [DemandParserHelper.hpp](#).

Referenced by [TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT
>::definition\(\)](#).

24.11.4.7 `template<typename ScannerT > boost::spirit::classic::rule<ScannerT>
TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT >::dow`

Definition at line 395 of file [DemandParserHelper.hpp](#).

Referenced by [TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT
>::definition\(\)](#).

24.11.4.8 `template<typename ScannerT > boost::spirit::classic::rule<ScannerT>
TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT
>::origin`

Definition at line 395 of file [DemandParserHelper.hpp](#).

Referenced by [TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT
>::definition\(\)](#).

24.11.4.9 `template<typename ScannerT > boost::spirit::classic::rule<ScannerT>
TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT
>::destination`

Definition at line 395 of file [DemandParserHelper.hpp](#).

Referenced by [TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT
>::definition\(\)](#).

24.11.4.10 `template<typename ScannerT > boost::spirit::classic::rule<ScannerT>
TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT
>::pref_cabin`

Definition at line 395 of file [DemandParserHelper.hpp](#).

Referenced by [TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT
>::definition\(\)](#).

24.11.4.11 `template<typename ScannerT > boost::spirit::classic::rule<ScannerT>
TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT
>::demand_params`

Definition at line 395 of file [DemandParserHelper.hpp](#).

Referenced by [TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT
>::definition\(\)](#).

24.11.4.12 `template<typename ScannerT > boost::spirit::classic::rule<ScannerT>
TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT
>::pos_dist`

Definition at line 395 of file [DemandParserHelper.hpp](#).

Referenced by [TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT
>::definition\(\)](#).

24.11.4.13 `template<typename ScannerT > boost::spirit::classic::rule<ScannerT>
TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT
>::pos_pair`

Definition at line 395 of file [DemandParserHelper.hpp](#).

Referenced by [TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT
>::definition\(\)](#).

24.11.4.14 `template<typename ScannerT > boost::spirit::classic::rule<ScannerT>
TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT
>::pos_code`

Definition at line 395 of file [DemandParserHelper.hpp](#).

Referenced by [TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT
>::definition\(\)](#).

24.11.4.15 `template<typename ScannerT > boost::spirit::classic::rule<ScannerT>
TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT
>::pos_share`

Definition at line 395 of file [DemandParserHelper.hpp](#).

Referenced by [TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT
>::definition\(\)](#).

24.11.4.16 `template<typename ScannerT > boost::spirit::classic::rule<ScannerT>
TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT
>::channel_dist`

Definition at line 395 of file [DemandParserHelper.hpp](#).

Referenced by [TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT
>::definition\(\)](#).

24.11.4.17 `template<typename ScannerT > boost::spirit::classic::rule<ScannerT>
TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT
>::channel_pair`

Definition at line 395 of file [DemandParserHelper.hpp](#).

Referenced by [TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT
>::definition\(\)](#).

24.11.4.18 `template<typename ScannerT > boost::spirit::classic::rule<ScannerT>
TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT
>::channel_code`

Definition at line 395 of file [DemandParserHelper.hpp](#).

Referenced by [TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT
>::definition\(\)](#).

24.11.4.19 `template<typename ScannerT > boost::spirit::classic::rule<ScannerT>
TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT
>::channel_share`

Definition at line 395 of file [DemandParserHelper.hpp](#).

Referenced by [TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT
>::definition\(\)](#).

24.11.4.20 `template<typename ScannerT > boost::spirit::classic::rule<ScannerT>
TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT
>::trip_dist`

Definition at line 395 of file [DemandParserHelper.hpp](#).

Referenced by [TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT
>::definition\(\)](#).

24.11.4.21 `template<typename ScannerT > boost::spirit::classic::rule<ScannerT>
TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT
>::trip_pair`

Definition at line 395 of file [DemandParserHelper.hpp](#).

Referenced by [TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT
>::definition\(\)](#).

24.11.4.22 `template<typename ScannerT > boost::spirit::classic::rule<ScannerT>
TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT
>::trip_code`

Definition at line 395 of file [DemandParserHelper.hpp](#).

Referenced by [TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT
>::definition\(\)](#).

24.11.4.23 `template<typename ScannerT > boost::spirit::classic::rule<ScannerT>
TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT
>::trip_share`

Definition at line 395 of file [DemandParserHelper.hpp](#).

Referenced by [TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT
>::definition\(\)](#).

24.11.4.24 `template<typename ScannerT > boost::spirit::classic::rule<ScannerT>
TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT
>::stay_dist`

Definition at line 395 of file [DemandParserHelper.hpp](#).

Referenced by [TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT
>::definition\(\)](#).

24.11.4.25 `template<typename ScannerT > boost::spirit::classic::rule<ScannerT>
TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT
>::stay_pair`

Definition at line 395 of file [DemandParserHelper.hpp](#).

Referenced by [TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT
>::definition\(\)](#).

24.11.4.26 `template<typename ScannerT > boost::spirit::classic::rule<ScannerT>
TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT
>::stay_share`

Definition at line 395 of file [DemandParserHelper.hpp](#).

Referenced by [TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT
>::definition\(\)](#).

24.11.4.27 `template<typename ScannerT > boost::spirit::classic::rule<ScannerT>
TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT
>::ff_dist`

Definition at line 395 of file [DemandParserHelper.hpp](#).

Referenced by [TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT
>::definition\(\)](#).

24.11.4.28 `template<typename ScannerT > boost::spirit::classic::rule<ScannerT>
TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT
>::ff_pair`

Definition at line 395 of file [DemandParserHelper.hpp](#).

Referenced by [TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT
>::definition\(\)](#).

24.11.4.29 `template<typename ScannerT > boost::spirit::classic::rule<ScannerT>
TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT
>::ff_code`

Definition at line 395 of file [DemandParserHelper.hpp](#).

Referenced by [TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT
>::definition\(\)](#).

24.11.4.30 `template<typename ScannerT > boost::spirit::classic::rule<ScannerT>
TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT
>::ff_share`

Definition at line 395 of file [DemandParserHelper.hpp](#).

Referenced by [TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT
>::definition\(\)](#).

24.11.4.31 `template<typename ScannerT > boost::spirit::classic::rule<ScannerT>
TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT
>::change_fees`

Definition at line 395 of file [DemandParserHelper.hpp](#).

Referenced by [TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT
>::definition\(\)](#).

24.11.4.32 `template<typename ScannerT > boost::spirit::classic::rule<ScannerT>
TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT
>::non_refundable`

Definition at line 395 of file [DemandParserHelper.hpp](#).

Referenced by [TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT
>::definition\(\)](#).

24.11.4.33 `template<typename ScannerT > boost::spirit::classic::rule<ScannerT>
TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT
>::pref_dep_time_dist`

Definition at line 395 of file [DemandParserHelper.hpp](#).

Referenced by [TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT
>::definition\(\)](#).

24.11.4.34 `template<typename ScannerT > boost::spirit::classic::rule<ScannerT>
TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT
>::pref_dep_time_pair`

Definition at line 395 of file [DemandParserHelper.hpp](#).

Referenced by [TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT
>::definition\(\)](#).

24.11.4.35 `template<typename ScannerT > boost::spirit::classic::rule<ScannerT>
TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT
>::pref_dep_time_share`

Definition at line 395 of file [DemandParserHelper.hpp](#).

Referenced by [TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT
>::definition\(\)](#).

24.11.4.36 `template<typename ScannerT > boost::spirit::classic::rule<ScannerT>
TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT
>::time`

Definition at line 395 of file [DemandParserHelper.hpp](#).

Referenced by [TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT
>::definition\(\)](#).

24.11.4.37 `template<typename ScannerT > boost::spirit::classic::rule<ScannerT>
TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT >::wtp`

Definition at line 395 of file [DemandParserHelper.hpp](#).

Referenced by [TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT
>::definition\(\)](#).

24.11.4.38 `template<typename ScannerT > boost::spirit::classic::rule<ScannerT>
TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT
>::time_value_dist`

Definition at line 395 of file [DemandParserHelper.hpp](#).

Referenced by [TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT
>::definition\(\)](#).

24.11.4.39 `template<typename ScannerT > boost::spirit::classic::rule<ScannerT>
TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT
>::time_value_pair`

Definition at line 395 of file [DemandParserHelper.hpp](#).

Referenced by [TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT
>::definition\(\)](#).

24.11.4.40 `template<typename ScannerT > boost::spirit::classic::rule<ScannerT>
TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT
>::time_value_share`

Definition at line 395 of file [DemandParserHelper.hpp](#).

Referenced by [TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT
>::definition\(\)](#).

24.11.4.41 `template<typename ScannerT > boost::spirit::classic::rule<ScannerT>
TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT
>::dtd_dist`

Definition at line 395 of file [DemandParserHelper.hpp](#).

Referenced by [TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT
>::definition\(\)](#).

24.11.4.42 `template<typename ScannerT > boost::spirit::classic::rule<ScannerT>
TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT
>::dtd_pair`

Definition at line 395 of file [DemandParserHelper.hpp](#).

Referenced by [TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT
>::definition\(\)](#).

24.11.4.43 `template<typename ScannerT> boost::spirit::classic::rule<ScannerT> TRADEMGENT::DemandParserHelper::DemandParser::definition< ScannerT >::dtd_share`

Definition at line 395 of file [DemandParserHelper.hpp](#).

Referenced by [TRADEMGENT::DemandParserHelper::DemandParser::definition< ScannerT >::definition\(\)](#).

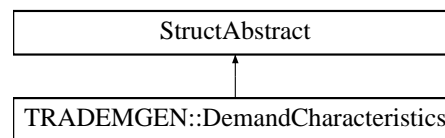
The documentation for this struct was generated from the following files:

- [trademgen/command/DemandParserHelper.hpp](#)
- [trademgen/command/DemandParserHelper.cpp](#)

24.12 TRADEMGEN::DemandCharacteristics Struct Reference

Class modeling the characteristics of a demand type.

`#include <trademgen/basic/DemandCharacteristics.hpp>` Inheritance diagram for TRADEMGENT::DemandCharacteristics:



Public Member Functions

- `const stdair::AirportCode_T & getPOSValue (const stdair::Probability_T &iCumulativeProbability) const`
- `bool checkPOSValue (const stdair::AirportCode_T &iPOS) const`
- `const std::string describe () const`
- `DemandCharacteristics (const ArrivalPatternCumulativeDistribution_T &, const POSProbabilityMassFunction_T &, const ChannelProbabilityMassFunction_T &, const TripTypeProbabilityMassFunction_T &, const StayDurationProbabilityMassFunction_T &, const FrequentFlyerProbabilityMassFunction_T &, const stdair::ChangeFeesRatio_T &, const stdair::Disutility_T &, const stdair::NonRefundableRatio_T &, const stdair::Disutility_T &, const PreferredDepartureTimeContinuousDistribution_T &, const stdair::WTP_T &, const ValueOfTimeContinuousDistribution_T &)`
- `DemandCharacteristics ()`
- `DemandCharacteristics (const DemandCharacteristics &)`
- `~DemandCharacteristics ()`

Public Attributes

- `ContinuousFloatDuration_T _arrivalPattern`
- `POSProbabilityMass_T _posProbabilityMass`
- `ChannelProbabilityMass_T _channelProbabilityMass`
- `TripTypeProbabilityMass_T _tripTypeProbabilityMass`

- [StayDurationProbabilityMass_T _stayDurationProbabilityMass](#)
- [FrequentFlyerProbabilityMass_T _frequentFlyerProbabilityMass](#)
- [stdair::ChangeFeesRatio_T _changeFeeProb](#)
- [stdair::Disutility_T _changeFeeDisutility](#)
- [stdair::NonRefundableRatio_T _nonRefundableProb](#)
- [stdair::Disutility_T _nonRefundableDisutility](#)
- [PreferredDepartureTimeCumulativeDistribution_T _preferredDepartureTimeCumulativeDistribution](#)
- [stdair::WTP_T _minWTP](#)
- [CumulativeDistribution_T _frat5Pattern](#)
- [ValueOfTimeCumulativeDistribution_T _valueOfTimeCumulativeDistribution](#)

24.12.1 Detailed Description

Class modeling the characteristics of a demand type.

Definition at line 21 of file [DemandCharacteristics.hpp](#).

24.12.2 Constructor & Destructor Documentation

24.12.2.1 TRADEMGEN::DemandCharacteristics::DemandCharacteristics
 (const ArrivalPatternCumulativeDistribution_T & *iArrivalPattern*,
 const POSProbabilityMassFunction_T & *iPOSProbMass*, const
 ChannelProbabilityMassFunction_T & *iChannelProbMass*, const
 TripTypeProbabilityMassFunction_T & *iTripTypeProbMass*, const
 StayDurationProbabilityMassFunction_T & *iStayDurationProbMass*, const
 FrequentFlyerProbabilityMassFunction_T & *iFrequentFlyerProbMass*, const
 stdair::ChangeFeesRatio_T & *iChangeFeeProb*, const stdair::Disutility_T
 & *iChangeFeeDisutility*, const stdair::NonRefundableRatio_T &
iNonRefundableProb, const stdair::Disutility_T & *iNonRefundableDisutility*,
 const PreferredDepartureTimeContinuousDistribution_T &
iPreferredDepartureTimeContinuousDistribution, const stdair::WTP_T & *iMinWTP*,
 const ValueOfTimeContinuousDistribution_T & *iValueOfTimeContinuousDistribution*)

Constructor.

Definition at line 50 of file [DemandCharacteristics.cpp](#).

24.12.2.2 TRADEMGEN::DemandCharacteristics::DemandCharacteristics ()

Default constructor.

Definition at line 16 of file [DemandCharacteristics.cpp](#).

**24.12.2.3 TRADEMGEN::DemandCharacteristics::DemandCharacteristics (const
 DemandCharacteristics & *iDC*)**

Copy constructor.

Definition at line 32 of file [DemandCharacteristics.cpp](#).

24.12.2.4 TRADEMGEN::DemandCharacteristics::~~DemandCharacteristics ()

Destructor.

Definition at line 79 of file [DemandCharacteristics.cpp](#).**24.12.3 Member Function Documentation****24.12.3.1 const stdair::AirportCode_T & TRADEMGEN::DemandCharacteristics::getPOSValue (const stdair::Probability_T & iCumulativeProbability) const**

Get the POS corresponding to the cumulative probability

Definition at line 84 of file [DemandCharacteristics.cpp](#).References [_posProbabilityMass](#), and [TRADEMGENT::CategoricalAttributeLite< T >::getValue\(\)](#).Referenced by [TRADEMGENT::DemandStream::generatePOS\(\)](#).**24.12.3.2 bool TRADEMGEN::DemandCharacteristics::checkPOSValue (const stdair::AirportCode_T & iPOS) const**

Check that the POS is within the distribution.

Definition at line 90 of file [DemandCharacteristics.cpp](#).References [_posProbabilityMass](#), and [TRADEMGENT::CategoricalAttributeLite< T >::checkValue\(\)](#).**24.12.3.3 const std::string TRADEMGEN::DemandCharacteristics::describe () const**

Give a description of the structure (for display purposes).

Definition at line 95 of file [DemandCharacteristics.cpp](#).

References [_arrivalPattern](#), [_changeFeeDisutility](#), [_changeFeeProb](#), [_channelProbabilityMass](#), [_frequentFlyerProbabilityMass](#), [_minWTP](#), [_nonRefundableDisutility](#), [_nonRefundableProb](#), [_posProbabilityMass](#), [_preferredDepartureTimeCumulativeDistribution](#), [_stayDurationProbabilityMass](#), [_tripTypeProbabilityMass](#), [_valueOfTimeCumulativeDistribution](#), [TRADEMGENT::ContinuousAttributeLite< T >::displayCumulativeDistribution\(\)](#), and [TRADEMGENT::CategoricalAttributeLite< T >::displayProbabilityMass\(\)](#).

Referenced by [TRADEMGENT::DemandStream::display\(\)](#).**24.12.4 Member Data Documentation****24.12.4.1 ContinuousFloatDuration_T TRADEMGEN::DemandCharacteristics::_arrivalPattern**

Arrival pattern (cumulative distribution of timing of arrival of requests (negative number of days between departure date and request date).

Definition at line 87 of file [DemandCharacteristics.hpp](#).Referenced by [describe\(\)](#), [TRADEMGENT::DemandStream::generateTimeOfRequestPoissonProcess\(\)](#), and [TRADEMGENT::DemandStream::generateTimeOfRequestStatisticsOrder\(\)](#).

24.12.4.2 POSProbabilityMass_T TRADEMGEN::DemandCharacteristics::_posProbabilityMass

POS probability mass.

Definition at line 92 of file [DemandCharacteristics.hpp](#).

Referenced by [checkPOSValue\(\)](#), [describe\(\)](#), and [getPOSValue\(\)](#).

24.12.4.3 ChannelProbabilityMass_T TRADEMGEN::DemandCharacteristics::_channelProbabilityMass

Channel probability mass.

Definition at line 97 of file [DemandCharacteristics.hpp](#).

Referenced by [describe\(\)](#), and [TRADEMGEN::DemandStream::generateChannel\(\)](#).

24.12.4.4 TripTypeProbabilityMass_T TRADEMGEN::DemandCharacteristics::_tripTypeProbabilityMass

Trip type probability mass.

Definition at line 102 of file [DemandCharacteristics.hpp](#).

Referenced by [describe\(\)](#), and [TRADEMGEN::DemandStream::generateTripType\(\)](#).

24.12.4.5 StayDurationProbabilityMass_T TRADEMGEN::DemandCharacteristics::_stayDurationProbabilityMass

Stay duration probability mass.

Definition at line 107 of file [DemandCharacteristics.hpp](#).

Referenced by [describe\(\)](#), and [TRADEMGEN::DemandStream::generateStayDuration\(\)](#).

24.12.4.6 FrequentFlyerProbabilityMass_T TRADEMGEN::DemandCharacteristics::_frequentFlyerProbabilityMass

Frequent flyer probability mass.

Definition at line 112 of file [DemandCharacteristics.hpp](#).

Referenced by [describe\(\)](#), and [TRADEMGEN::DemandStream::generateFrequentFlyer\(\)](#).

24.12.4.7 stdair::ChangeFeesRatio_T TRADEMGEN::DemandCharacteristics::_changeFeeProb

Change fee restriction acceptance probability.

Definition at line 117 of file [DemandCharacteristics.hpp](#).

Referenced by [describe\(\)](#), and [TRADEMGEN::DemandStream::generateChangeFees\(\)](#).

24.12.4.8 stdair::Disutility_T TRADEMGEN::DemandCharacteristics::_changeFeeDisutility

Change fee disutility.

Definition at line 122 of file [DemandCharacteristics.hpp](#).

Referenced by [describe\(\)](#), [TRADEMGEN::DemandStream::generateNextRequest\(\)](#), and [TRADEMGEN::DemandStream::getChangeFeeDisutility\(\)](#).

24.12.4.9 `stdair::NonRefundableRatio_T TRADEMGEN::DemandCharacteristics::_nonRefundableProb`

Non refundable restriction acceptance probability.

Definition at line 127 of file [DemandCharacteristics.hpp](#).

Referenced by [describe\(\)](#), and [TRADEMGEN::DemandStream::generateNonRefundable\(\)](#).

24.12.4.10 `stdair::Disutility_T TRADEMGEN::DemandCharacteristics::_nonRefundableDisutility`

Non refundable disutility.

Definition at line 132 of file [DemandCharacteristics.hpp](#).

Referenced by [describe\(\)](#), [TRADEMGEN::DemandStream::generateNextRequest\(\)](#), and [TRADEMGEN::DemandStream::getNonRefundableDisutility\(\)](#).

24.12.4.11 `PreferredDepartureTimeCumulativeDistribution_T TRADEMGEN::DemandCharacteristics::_preferredDepartureTimeCumulativeDistribution`

Preferred departure time cumulative distribution.

Definition at line 137 of file [DemandCharacteristics.hpp](#).

Referenced by [describe\(\)](#).

24.12.4.12 `stdair::WTP_T TRADEMGEN::DemandCharacteristics::_minWTP`

Min Willingness-to-pay, used for the computation of the WTP of each request.

Definition at line 143 of file [DemandCharacteristics.hpp](#).

Referenced by [describe\(\)](#), and [TRADEMGEN::DemandStream::generateWTP\(\)](#).

24.12.4.13 `CumulativeDistribution_T TRADEMGEN::DemandCharacteristics::_frat5Pattern`

FRAT5 pattern, used for the computation of WTP.

Definition at line 148 of file [DemandCharacteristics.hpp](#).

Referenced by [TRADEMGEN::DemandStream::generateWTP\(\)](#).

24.12.4.14 `ValueOfTimeCumulativeDistribution_T TRADEMGEN::DemandCharacteristics::_valueOfTimeCumulativeDistribution`

Value of time cumulative distribution.

Definition at line 153 of file [DemandCharacteristics.hpp](#).

Referenced by [describe\(\)](#), and [TRADEMGEN::DemandStream::generateValueOfTime\(\)](#).

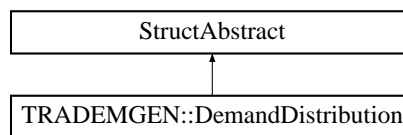
The documentation for this struct was generated from the following files:

- [trademgen/basic/DemandCharacteristics.hpp](#)
- [trademgen/basic/DemandCharacteristics.cpp](#)

24.13 TRADEMGEN::DemandDistribution Struct Reference

Class modeling the distribution of a demand type.

`#include <trademgen/basic/DemandDistribution.hpp>`
 Inheritance diagram for TRADEMGEN::DemandDistribution::



Public Member Functions

- [DemandDistribution](#) (const stdair::NbOfRequests_T &iMean, const stdair::StdDevValue_T &iStdDev)
- [DemandDistribution](#) ()
- [DemandDistribution](#) (const [DemandDistribution](#) &)
- [~DemandDistribution](#) ()
- void [fromStream](#) (std::istream &ioIn)
- const std::string [describe](#) () const
- std::string [display](#) () const

Public Attributes

- stdair::NbOfRequests_T [_meanNumberOfRequests](#)
- stdair::StdDevValue_T [_stdDevNumberOfRequests](#)

24.13.1 Detailed Description

Class modeling the distribution of a demand type.

Definition at line 20 of file [DemandDistribution.hpp](#).

24.13.2 Constructor & Destructor Documentation

24.13.2.1 TRADEMGEN::DemandDistribution::DemandDistribution (const stdair::NbOfRequests_T &iMean, const stdair::StdDevValue_T &iStdDev)

Constructor.

Definition at line 15 of file [DemandDistribution.cpp](#).

24.13.2.2 TRADEMGEN::DemandDistribution::DemandDistribution ()

Default constructor.

Definition at line 22 of file [DemandDistribution.cpp](#).

24.13.2.3 TRADEMGEN::DemandDistribution::DemandDistribution (const DemandDistribution & iDemandDistribution)

Copy constructor.

Definition at line 31 of file [DemandDistribution.cpp](#).

24.13.2.4 TRADEMGEN::DemandDistribution::~~DemandDistribution ()

Destructor.

Definition at line 26 of file [DemandDistribution.cpp](#).

24.13.3 Member Function Documentation

24.13.3.1 void TRADEMGEN::DemandDistribution::fromStream (std::istream & ioln)

Read a Business Object from an input stream.

Parameters:

istream& the input stream.

Definition at line 37 of file [DemandDistribution.cpp](#).

24.13.3.2 const std::string TRADEMGEN::DemandDistribution::describe () const

Display of the structure.

Definition at line 41 of file [DemandDistribution.cpp](#).

References [_meanNumberOfRequests](#), and [_stdDevNumberOfRequests](#).

Referenced by [TRADEMGEN::DemandStream::display\(\)](#), and [display\(\)](#).

24.13.3.3 std::string TRADEMGEN::DemandDistribution::display () const

Display demand distribution.

Definition at line 49 of file [DemandDistribution.cpp](#).

References [describe\(\)](#).

24.13.4 Member Data Documentation

24.13.4.1 stdair::NbOfRequests_T TRADEMGEN::DemandDistribution::_meanNumberOfRequests

Mean number of requests.

Definition at line 67 of file [DemandDistribution.hpp](#).

Referenced by [describe\(\)](#), [TRADEMGENT::DemandStream::generateTimeOfRequestPoissonProcess\(\)](#), and [TRADEMGENT::DemandStream::getMeanNumberOfRequests\(\)](#).

24.13.4.2 stdair::StdDevValue_T TRADEMGENT::DemandDistribution::-stdDevNumberOfRequests

Standard deviation of number of requests.

Definition at line 72 of file [DemandDistribution.hpp](#).

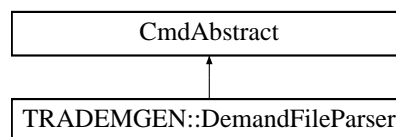
Referenced by [describe\(\)](#), and [TRADEMGENT::DemandStream::getStdDevNumberOfRequests\(\)](#).

The documentation for this struct was generated from the following files:

- [trademgen/basic/DemandDistribution.hpp](#)
- [trademgen/basic/DemandDistribution.cpp](#)

24.14 TRADEMGENT::DemandFileParser Class Reference

`#include <trademgen/command/DemandParserHelper.hpp>` Inheritance diagram for TRADEMGENT::DemandFileParser::



Public Member Functions

- [DemandFileParser](#) (SEVMGR::SEVMGR_ServicePtr_T, stdair::RandomGeneration &, const POSProbabilityMass_T &, const stdair::Filename_T &iDemandInputFilename)
- bool [generateDemand](#) ()

24.14.1 Detailed Description

Class wrapping the initialisation and entry point of the parser.

The seemingly redundancy is used to force the instantiation of the actual parser, which is a templatised Boost Spirit grammar. Hence, the actual parser is instantiated within that class object code.

Definition at line 434 of file [DemandParserHelper.hpp](#).

24.14.2 Constructor & Destructor Documentation

24.14.2.1 TRADEMGENT::DemandFileParser::DemandFileParser (SEVMGR::SEVMGR_ServicePtr_T, stdair::RandomGeneration &, const POSProbabilityMass_T &, const stdair::Filename_T &iDemandInputFilename)

Constructor.

24.14.3 Member Function Documentation

24.14.3.1 bool TRADEMGEN::DemandFileParser::generateDemand ()

Parse the demand input file.

Definition at line 897 of file [DemandParserHelper.cpp](#).

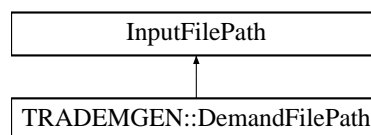
Referenced by [TRADEMGEN::DemandParser::generateDemand\(\)](#).

The documentation for this class was generated from the following files:

- trademgen/command/[DemandParserHelper.hpp](#)
- trademgen/command/[DemandParserHelper.cpp](#)

24.15 TRADEMGEN::DemandFilePath Class Reference

`#include <trademgen/TRADEMGEN_Types.hpp>`Inheritance diagram for TRADEMGEN::DemandFilePath::



Public Member Functions

- [DemandFilePath](#) (const stdair::Filename_T &iFilename)

24.15.1 Detailed Description

Demand input file.

Definition at line 30 of file [TRADEMGEN_Types.hpp](#).

24.15.2 Constructor & Destructor Documentation

24.15.2.1 TRADEMGEN::DemandFilePath::DemandFilePath (const stdair::Filename_T &iFilename) [inline, explicit]

Constructor.

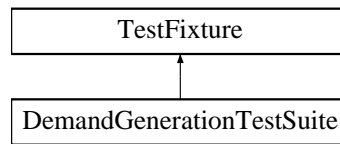
Definition at line 35 of file [TRADEMGEN_Types.hpp](#).

The documentation for this class was generated from the following file:

- trademgen/[TRADEMGEN_Types.hpp](#)

24.16 DemandGenerationTestSuite Class Reference

`#include <test/trademgen/DemandGenerationTestSuite.hpp>`Inheritance diagram for DemandGenerationTestSuite::



Public Member Functions

- void [simpleEventGeneration](#) ()
- [DemandGenerationTestSuite](#) ()

Protected Attributes

- std::stringstream [_describeKey](#)

24.16.1 Detailed Description

Definition at line 6 of file [DemandGenerationTestSuite.hpp](#).

24.16.2 Constructor & Destructor Documentation

24.16.2.1 DemandGenerationTestSuite::DemandGenerationTestSuite ()

Test some error detection functionalities. Constructor.

24.16.3 Member Function Documentation

24.16.3.1 void DemandGenerationTestSuite::simpleEventGeneration ()

Test a simple event generation functionality.

24.16.4 Member Data Documentation

24.16.4.1 std::stringstream DemandGenerationTestSuite::_describeKey [protected]

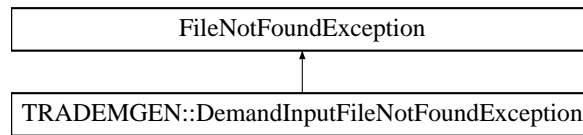
Definition at line 27 of file [DemandGenerationTestSuite.hpp](#).

The documentation for this class was generated from the following file:

- test/trademgen/[DemandGenerationTestSuite.hpp](#)

24.17 TRADEMGEN::DemandInputFileNotFoundException Class Reference

`#include <trademgen/TRADEMGEN_Exceptions.hpp>`Inheritance diagram for TRADEMGEN::DemandInputFileNotFoundException::



Public Member Functions

- [DemandInputFileNotFoundException](#) (const std::string &iWhat)

24.17.1 Detailed Description

Exception when no demand input file can be found

Definition at line 30 of file [TRADEMGEN_Exceptions.hpp](#).

24.17.2 Constructor & Destructor Documentation

24.17.2.1 TRADEMGEN::DemandInputFileNotFoundException::DemandInputFileNotFoundException (const std::string & iWhat) [inline]

Constructor.

Definition at line 36 of file [TRADEMGEN_Exceptions.hpp](#).

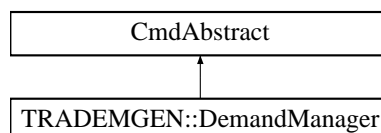
The documentation for this class was generated from the following file:

- trademgen/[TRADEMGEN_Exceptions.hpp](#)

24.18 TRADEMGEN::DemandManager Class Reference

Utility class for Demand and [DemandStream](#) objects.

#include <trademgen/command/DemandManager.hpp>Inheritance diagram for TRADEMGEN::DemandManager:



Friends

- struct [DemandParserHelper::doEndDemand](#)
- class [TRADEMGEN_Service](#)

24.18.1 Detailed Description

Utility class for Demand and [DemandStream](#) objects.

Definition at line 40 of file [DemandManager.hpp](#).

24.18.2 Friends And Related Function Documentation

24.18.2.1 friend struct DemandParserHelper::doEndDemand [friend]

Definition at line 41 of file [DemandManager.hpp](#).

24.18.2.2 friend class TRADEMGEN_Service [friend]

Definition at line 42 of file [DemandManager.hpp](#).

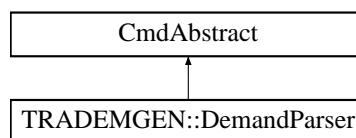
The documentation for this class was generated from the following files:

- [trademgen/command/DemandManager.hpp](#)
- [trademgen/command/DemandManager.cpp](#)

24.19 TRADEMGEN::DemandParser Class Reference

Class wrapping the parser entry point.

`#include <trademgen/command/DemandParser.hpp>`Inheritance diagram for TRADEMGEN::DemandParser::



Static Public Member Functions

- static void [generateDemand](#) (const [DemandFilePath](#) &, SEVMGR::SEVMGR_ServicePtr_T, stdair::RandomGeneration &, const [POSProbabilityMass_T](#) &)

24.19.1 Detailed Description

Class wrapping the parser entry point.

Definition at line 28 of file [DemandParser.hpp](#).

24.19.2 Member Function Documentation

24.19.2.1 `void TRADEMGEN::DemandParser::generateDemand (const DemandFilePath & iDemandFilename, SEVMGR::SEVMGR_ServicePtr_T ioSEVMGR_ServicePtr, stdair::RandomGeneration & ioSharedGenerator, const POSProbabilityMass_T & iDefaultPOSProbabilityMass) [static]`

Parse the CSV file describing travel demand, for instance for generating simulated booking request in a simulator.

The state of the random generator, given as parameter, evolves each time a demand request is generated.

Parameters:

const DemandFilePath& The file-name of the CSV-formatted demand input file.

SEVMGR::SEVMGR_ServicePtr_T Pointer on the SEvMgr service handler to update the queue with the parsed information.

stdair::RandomGeneration& Random generator.

Definition at line 18 of file [DemandParser.cpp](#).

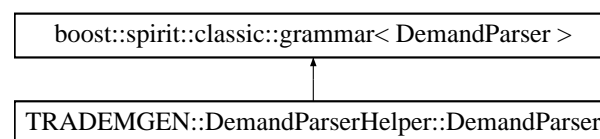
References [TRADEMGEM::DemandFileParser::generateDemand\(\)](#).

The documentation for this class was generated from the following files:

- [trademgen/command/DemandParser.hpp](#)
- [trademgen/command/DemandParser.cpp](#)

24.20 TRADEMGEN::DemandParserHelper::DemandParser Struct Reference

`#include <trademgen/command/DemandParserHelper.hpp>`Inheritance diagram for TRADEMGEM::DemandParserHelper::DemandParser::



Classes

- struct [definition](#)

Public Member Functions

- [DemandParser](#) (SEVMGR::SEVMGR_ServicePtr_T, stdair::RandomGeneration &, const POSProbabilityMass_T &, [DemandStruct](#) &)

Public Attributes

- SEVMGR::SEVMGR_ServicePtr_T [_sevmgrServicePtr](#)

- [stdair::RandomGeneration](#) & [_uniformGenerator](#)
- [const POSProbabilityMass_T](#) & [_posProbabilityMass](#)
- [DemandStruct](#) & [_demand](#)

24.20.1 Detailed Description

PrefDepDate; Origin; Destination; PassengerType; Mean; StdDev; PosDist; ChannelDist; TripTypeDist; StayDurationDist; FrequentFlyerDist; PrefDepTimeDist; min WTP; (PrefArrivalDate; PrefArrivalTime;) TimeValueDist; ValueOfTimeDist; ArrivalPatternDist; 2010-02-08; SIN; BKK; L; 10.0; 1.0; SIN:0.7, BKK:0.2, row:0.1; DF:0.1, DN:0.3, IF:0.4, IN:0.2; RO:0.6, RI:0.2, OW:0.2; 0:0.1, 1:0.1, 2:0.15, 3:0.15, 4:0.15, 5:0.35; P:0.01, G:0.05, S:0.15, M:0.3, N:0.49; 06:0, 07:0.1, 09:0.3, 17:0.4, 19:0.8, 20:0.95, 22:1; 100:0, 500:0.8, 2000:1; 15:0, 60:1; 330:0, 40:0.2, 20:0.6, 1:1;

Fixed: Preferred departure date (yyyy-mm-dd) Origin (3-char airport code) Destination (3-char airport code) PassengerType (1-char, e.g., 'L' for Leisure, 'B' for Business) Observable: Mean StdDev Distribution with Probability Masses: POS Channel (D=direct, I=indirect, N=online, F=offline) Trip type (RO=outbound of round-trip, RI=inbound of round-trip, OW=one way) Stay duration (number of days) Frequent flyer (P=Platinum, G=Gold, S=Silver, M=Member, N=None) Change fees restriction. 'True' for accepting the restriction (for hard-restriction customer choice model) Change fees disutility (for disutility customer choice model) Non refundable restriction. 'True' for accepting the restriction (for hard-restriction customer choice model) Non refundable disutility (for disutility customer choice model) Continuous cumulative distribution: Preferred departure time (hh:mm:ss) Preferred arrival date (equal to preferred departure date) Preferred arrival time (equal to preferred departure time) Value of time Arrival pattern (DTD as a positive value) The main fields are separated by ';' Probability mass distributions are defined by comma-separated 'value:probability' pairs Continuous cumulative distribution are defined by comma-separated 'value:probability' pairs, sorted in increasing order of values. The meaning of probability is P(random variable <= value) = probability.

Grammar: Demand ::= PrefDepDate ';' Origin ';' Destination ';' PassengerType ';' DemandParams ';' PosDist ';' ChannelDist ';' TripDist ';' StayDist ';' FFDist ';' PrefDepTimeDist ';' minWTP ';' TimeValueDist ';' DtdDist EndOfDemand PrefDepDate ::= date PassengerType ::= 'L' | 'B' | 'F' DemandParams ::= DemandMean ';' DemandStdDev PosDist ::= PosPair (';' PosPair)* PosPair ::= PosCode ';' PosShare PosCode ::= AirportCode | "row" PosShare ::= real ChannelDist ::= ChannelPair (';' ChannelPair)* ChannelPair ::= Channel_Code ';' ChannelShare ChannelCode ::= "DF" | "DN" | "IF" | "IN" ChannelShare ::= real TripDist ::= TripPair (';' TripPair)* TripPair ::= TripCode ';' TripShare TripCode ::= "RO" | "RI" | "OW" TripShare ::= real StayDist ::= StayPair (';' StayPair)* StayPair ::= [0;3]-digit-integer ';' stay_share StayShare ::= real FFDist ::= FF_Pair (';' FF_Pair)* FFPair ::= FFCODE ';' FFShare FFCODE ::= 'P' | 'G' | 'S' | 'M' | 'N' FFShare ::= real ChangeFeeProb ::= real NonRefundableProb ::= real PrefDepTimeDist ::= PrefDepTimePair (';' PrefDepTimePair)* PrefDepTimePair ::= time ';' PrefDepTimeShare PrefDepTimeShare ::= real minWTP ::= real TimeValueDist ::= TimeValuePair (';' TimeValuePair)* TimeValuePair ::= [0;2]-digit-integer ';' TimeValueShare TimeValueShare ::= real DTDDist ::= DTDPair (';' DTDPair)* DTDPair ::= real ';' DTDDShare DTDDShare ::= real EndOfDemand ::= ';' Grammar for the demand parser.

Definition at line 384 of file [DemandParserHelper.hpp](#).

24.20.2 Constructor & Destructor Documentation

24.20.2.1 TRADEMGEN::DemandParserHelper::DemandParser::DemandParser (SEVMGR::SEVMGR_ServicePtr_T ioSEVMGR_ServicePtr, stdair::RandomGeneration & ioSharedGenerator, const POSProbabilityMass_T & iPOSProbMass, DemandStruct & ioDemand)

Definition at line 568 of file [DemandParserHelper.cpp](#).

24.20.3 Member Data Documentation

24.20.3.1 SEVMGR::SEVMGR_ServicePtr_T TRADEMGENT::DemandParserHelper::DemandParser::_sevmgrServicePtr

Definition at line 415 of file [DemandParserHelper.hpp](#).

Referenced by [TRADEMGENT::DemandParserHelper::DemandParser::definition< ScannerT >::definition\(\)](#).

24.20.3.2 stdair::RandomGeneration& TRADEMGENT::DemandParserHelper::DemandParser::_uniformGenerator

Definition at line 416 of file [DemandParserHelper.hpp](#).

Referenced by [TRADEMGENT::DemandParserHelper::DemandParser::definition< ScannerT >::definition\(\)](#).

24.20.3.3 const POSProbabilityMass_T& TRADEMGENT::DemandParserHelper::DemandParser::_posProbabilityMass

Definition at line 417 of file [DemandParserHelper.hpp](#).

Referenced by [TRADEMGENT::DemandParserHelper::DemandParser::definition< ScannerT >::definition\(\)](#).

24.20.3.4 DemandStruct& TRADEMGENT::DemandParserHelper::DemandParser::_demand

Definition at line 418 of file [DemandParserHelper.hpp](#).

Referenced by [TRADEMGENT::DemandParserHelper::DemandParser::definition< ScannerT >::definition\(\)](#).

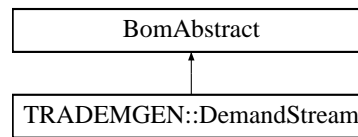
The documentation for this struct was generated from the following files:

- [trademgen/command/DemandParserHelper.hpp](#)
- [trademgen/command/DemandParserHelper.cpp](#)

24.21 TRADEMGENT::DemandStream Class Reference

Class modeling a demand stream.

`#include <trademgen/bom/DemandStream.hpp>` Inheritance diagram for TRADEMGENT::DemandStream::



Public Types

- typedef [DemandStreamKey](#) [Key_T](#)

Public Member Functions

- const [Key_T](#) & [getKey](#) () const
- [BomAbstract](#) *const [getParent](#) () const
- const stdair::AirportCode_T & [getOrigin](#) () const
- const stdair::AirportCode_T & [getDestination](#) () const
- const stdair::Date_T & [getPreferredDepartureDate](#) () const
- const stdair::CabinCode_T & [getPreferredCabin](#) () const
- const stdair::HolderMap_T & [getHolderMap](#) () const
- const [DemandCharacteristics](#) & [getDemandCharacteristics](#) () const
- const [DemandDistribution](#) & [getDemandDistribution](#) () const
- const stdair::NbOfRequests_T & [getTotalNumberOfRequestsToBeGenerated](#) () const
- const stdair::NbOfRequests_T & [getMeanNumberOfRequests](#) () const
- const stdair::StdDevValue_T & [getStdDevNumberOfRequests](#) () const
- const stdair::Count_T & [getNumberOfRequestsGeneratedSoFar](#) () const
- const stdair::Disutility_T & [getChangeFeeDisutility](#) () const
- const stdair::Disutility_T & [getNonRefundableDisutility](#) () const
- const [POSProbabilityMass_T](#) & [getPOSProbabilityMass](#) () const
- void [setNumberOfRequestsGeneratedSoFar](#) (const stdair::Count_T & iCount)
- void [setDemandDistribution](#) (const [DemandDistribution](#) & iDemandDistribution)
- void [setDemandCharacteristics](#) (const [ArrivalPatternCumulativeDistribution_T](#) & iArrivalPattern, const [POSProbabilityMassFunction_T](#) & iPOSProbMass, const [ChannelProbabilityMassFunction_T](#) & iChannelProbMass, const [TripTypeProbabilityMassFunction_T](#) & iTripTypeProbMass, const [StayDurationProbabilityMassFunction_T](#) & iStayDurationProbMass, const [FrequentFlyerProbabilityMassFunction_T](#) & iFrequentFlyerProbMass, const stdair::ChangeFeesRatio_T & iChangeFeeProb, const stdair::Disutility_T & iChangeFeeDisutility, const stdair::NonRefundableRatio_T & iNonRefundableProb, const stdair::Disutility_T & iNonRefundableDisutility, const [PreferredDepartureTimeContinuousDistribution_T](#) & iPreferredDepartureTimeContinuousDistribution, const stdair::WTP_T & iMinWTP, const [ValueOfTimeContinuousDistribution_T](#) & iValueOfTimeContinuousDistribution)
- void [setTotalNumberOfRequestsToBeGenerated](#) (const stdair::NbOfRequests_T & iNbOfRequests)
- void [setRequestDateTimeRandomGeneratorSeed](#) (const stdair::RandomSeed_T & iSeed)
- void [setDemandCharacteristicsRandomGeneratorSeed](#) (const stdair::RandomSeed_T & iSeed)
- void [setPOSProbabilityMass](#) (const [POSProbabilityMass_T](#) & iProbMass)
- void [setAll](#) (const [ArrivalPatternCumulativeDistribution_T](#) &, const [POSProbabilityMassFunction_T](#) &, const [ChannelProbabilityMassFunction_T](#) &, const [TripTypeProbabilityMassFunction_T](#) &, const [StayDurationProbabilityMassFunction_T](#) &, const [FrequentFlyerProbabilityMassFunction_T](#) &, const stdair::ChangeFeesRatio_T &, const stdair::Disutility_T &, const stdair::NonRefundableRatio_T &, const stdair::Disutility_T &, const [PreferredDepartureTimeContinuousDistribution_T](#) &, const stdair::WTP_T &, const

- [ValueOfTimeContinuousDistribution_T](#) &, const [DemandDistribution](#) &, stdair::BaseGenerator_T &ioSharedGenerator, const stdair::RandomSeed_T &iRequestDateTimeSeed, const stdair::RandomSeed_T &iDemandCharacteristicsSeed, const [POSProbabilityMass_T](#) &)
- void [setBoolFirstDateTimeRequest](#) (const bool &iFirstDateTimeRequest)
- void [incrementGeneratedRequestsCounter](#) ()
- const bool [stillHavingRequestsToBeGenerated](#) (const stdair::DemandGenerationMethod &iDemandGenerationMethod) const
- const stdair::DateTime_T [generateTimeOfRequestPoissonProcess](#) ()
- const stdair::DateTime_T [generateTimeOfRequestStatisticsOrder](#) ()
- const stdair::AirportCode_T [generatePOS](#) ()
- const stdair::ChannelLabel_T [generateChannel](#) ()
- const stdair::TripType_T [generateTripType](#) ()
- const stdair::DayDuration_T [generateStayDuration](#) ()
- const stdair::FrequentFlyer_T [generateFrequentFlyer](#) ()
- const stdair::ChangeFees_T [generateChangeFees](#) ()
- const stdair::NonRefundable_T [generateNonRefundable](#) ()
- const stdair::Duration_T [generatePreferredDepartureTime](#) ()
- const stdair::WTP_T [generateWTP](#) (stdair::RandomGeneration &, const stdair::Date_T &, const stdair::DateTime_T &, const stdair::DayDuration_T &)
- const stdair::PriceValue_T [generateValueOfTime](#) ()
- stdair::BookingRequestPtr_T [generateNextRequest](#) (stdair::RandomGeneration &, const stdair::DemandGenerationMethod &)
- void [reset](#) (stdair::BaseGenerator_T &ioSharedGenerator)
- void [toStream](#) (std::ostream &ioOut) const
- void [fromStream](#) (std::istream &ioIn)
- std::string [toString](#) () const
- const std::string [describeKey](#) () const
- std::string [display](#) () const
- const stdair::Duration_T [convertFloatIntoDuration](#) (const stdair::FloatDuration_T)

Protected Member Functions

- [DemandStream](#) (const [Key_T](#) &)
- virtual [~DemandStream](#) ()

Protected Attributes

- [Key_T_key](#)
- [BomAbstract](#) * [_parent](#)
- stdair::HolderMap_T [_holderMap](#)
- [DemandCharacteristics](#) [_demandCharacteristics](#)
- [DemandDistribution](#) [_demandDistribution](#)
- stdair::NbOfRequests_T [_totalNumberOfRequestsToBeGenerated](#)
- [RandomGenerationContext](#) [_randomGenerationContext](#)
- stdair::RandomGeneration [_requestDateTimeRandomGenerator](#)
- stdair::RandomGeneration [_demandCharacteristicsRandomGenerator](#)
- [POSProbabilityMass_T](#) [_posProMass](#)

Friends

- class [stdair::FacBom](#)
- class [stdair::FacBomManager](#)

24.21.1 Detailed Description

Class modeling a demand stream.

Definition at line 30 of file [DemandStream.hpp](#).

24.21.2 Member Typedef Documentation

24.21.2.1 typedef DemandStreamKey TRADEMGEN::DemandStream::Key_T

Definition allowing to retrieve the associated BOM key type.

Definition at line 39 of file [DemandStream.hpp](#).

24.21.3 Constructor & Destructor Documentation

24.21.3.1 TRADEMGEN::DemandStream::DemandStream (const Key_T & iKey) [protected]

Main constructor.

Definition at line 64 of file [DemandStream.cpp](#).

24.21.3.2 TRADEMGEN::DemandStream::~~DemandStream () [protected, virtual]

Destructor.

Definition at line 69 of file [DemandStream.cpp](#).

24.21.4 Member Function Documentation

24.21.4.1 const Key_T& TRADEMGEN::DemandStream::getKey () const [inline]

Get the key

Definition at line 45 of file [DemandStream.hpp](#).

References [_key](#).

24.21.4.2 BomAbstract* const TRADEMGEN::DemandStream::getParent () const [inline]

Get the parent object (EventQueue).

Definition at line 50 of file [DemandStream.hpp](#).

References [_parent](#).

24.21.4.3 `const stdair::AirportCode_T& TRADEMGEN::DemandStream::getOrigin () const [inline]`

Get the origin (part of the primary key).

Definition at line 55 of file [DemandStream.hpp](#).

References [_key](#), and [TRADEMGENT::DemandStreamKey::getOrigin\(\)](#).

24.21.4.4 `const stdair::AirportCode_T& TRADEMGEN::DemandStream::getDestination () const [inline]`

Get the destination (part of the primary key).

Definition at line 60 of file [DemandStream.hpp](#).

References [_key](#), and [TRADEMGENT::DemandStreamKey::getDestination\(\)](#).

24.21.4.5 `const stdair::Date_T& TRADEMGEN::DemandStream::getPreferredDepartureDate () const [inline]`

Get the preferred departure date (part of the primary key).

Definition at line 65 of file [DemandStream.hpp](#).

References [_key](#), and [TRADEMGENT::DemandStreamKey::getPreferredDepartureDate\(\)](#).

24.21.4.6 `const stdair::CabinCode_T& TRADEMGEN::DemandStream::getPreferredCabin () const [inline]`

Get the preferred cabin (part of the primary key).

Definition at line 70 of file [DemandStream.hpp](#).

References [_key](#), and [TRADEMGENT::DemandStreamKey::getPreferredCabin\(\)](#).

24.21.4.7 `const stdair::HolderMap_T& TRADEMGEN::DemandStream::getHolderMap () const [inline]`

Get the map of children holders.

Definition at line 75 of file [DemandStream.hpp](#).

References [_holderMap](#).

24.21.4.8 `const DemandCharacteristics& TRADEMGEN::DemandStream::getDemandCharacteristics () const [inline]`

Get the demand characteristics.

Definition at line 80 of file [DemandStream.hpp](#).

References [_demandCharacteristics](#).

24.21.4.9 `const DemandDistribution& TRADEMGEN::DemandStream::getDemandDistribution () const [inline]`

Get the demand distribution.

Definition at line 85 of file [DemandStream.hpp](#).

References [_demandDistribution](#).

24.21.4.10 `const stdair::NbOfRequests_T& TRADEMGEN::DemandStream::getTotalNumberOfRequestsToBeGenerated () const [inline]`

Get the total number of requests to be generated.

Definition at line 90 of file [DemandStream.hpp](#).

References [_totalNumberOfRequestsToBeGenerated](#).

24.21.4.11 `const stdair::NbOfRequests_T& TRADEMGEN::DemandStream::getMeanNumberOfRequests () const [inline]`

Get the mean (expected) number of requests.

Definition at line 95 of file [DemandStream.hpp](#).

References [_demandDistribution](#), and [TRADEMGENT::DemandDistribution::_meanNumberOfRequests](#).

24.21.4.12 `const stdair::StdDevValue_T& TRADEMGEN::DemandStream::getStdDevNumberOfRequests () const [inline]`

Get the standard deviation of number of requests.

Definition at line 100 of file [DemandStream.hpp](#).

References [_demandDistribution](#), and [TRADEMGENT::DemandDistribution::_stdDevNumberOfRequests](#).

24.21.4.13 `const stdair::Count_T& TRADEMGEN::DemandStream::getNumberOfRequestsGeneratedSoFar () const [inline]`

Get the number of requests generated so far.

Definition at line 105 of file [DemandStream.hpp](#).

References [_randomGenerationContext](#), and [TRADEMGENT::RandomGenerationContext::getNumberOfRequestsGeneratedSoFar](#).

24.21.4.14 `const stdair::Disutility_T& TRADEMGEN::DemandStream::getChangeFeeDisutility () const [inline]`

Get the change fee disutility.

Definition at line 110 of file [DemandStream.hpp](#).

References [TRADEMGEN::DemandCharacteristics::_changeFeeDisutility](#), and [_demandCharacteristics](#).

24.21.4.15 `const stdair::Disutility_T& TRADEMGEN::DemandStream::getNonRefundableDisutility () const [inline]`

Get the non refundable disutility.

Definition at line 115 of file [DemandStream.hpp](#).

References [_demandCharacteristics](#), and [TRADEMGEN::DemandCharacteristics::_nonRefundableDisutility](#).

24.21.4.16 `const POSProbabilityMass_T& TRADEMGEN::DemandStream::getPOSProbabilityMass () const [inline]`

Get the default POS probablity mass, used when "row" (rest of the world) is drawn.

Definition at line 123 of file [DemandStream.hpp](#).

References [_posProMass](#).

24.21.4.17 `void TRADEMGEN::DemandStream::setNumberOfRequestsGeneratedSoFar (const stdair::Count_T & iCount) [inline]`

Set the number of requests generated so far.

Definition at line 131 of file [DemandStream.hpp](#).

References [_randomGenerationContext](#), and [TRADEMGEN::RandomGenerationContext::setNumberOfRequestsGeneratedSoFar](#).

24.21.4.18 `void TRADEMGEN::DemandStream::setDemandDistribution (const DemandDistribution & iDemandDistribution) [inline]`

Set the demand distribution.

Definition at line 136 of file [DemandStream.hpp](#).

References [_demandDistribution](#).

Referenced by [setAll\(\)](#).

24.21.4.19 void TRADEMGEN::DemandStream::setDemandCharacteristics (const ArrivalPatternCumulativeDistribution_T & *iArrivalPattern*, const POSProbabilityMassFunction_T & *iPOSProbMass*, const ChannelProbabilityMassFunction_T & *iChannelProbMass*, const TripTypeProbabilityMassFunction_T & *iTripTypeProbMass*, const StayDurationProbabilityMassFunction_T & *iStayDurationProbMass*, const FrequentFlyerProbabilityMassFunction_T & *iFrequentFlyerProbMass*, const stdair::ChangeFeesRatio_T & *iChangeFeeProb*, const stdair::Disutility_T & *iChangeFeeDisutility*, const stdair::NonRefundableRatio_T & *iNonRefundableProb*, const stdair::Disutility_T & *iNonRefundableDisutility*, const PreferredDepartureTimeContinuousDistribution_T & *iPreferredDepartureTimeContinuousDistribution*, const stdair::WTP_T & *iMinWTP*, const ValueOfTimeContinuousDistribution_T & *iValueOfTimeContinuousDistribution*) [inline]

Set the demand characteristics.

Definition at line 142 of file [DemandStream.hpp](#).

References [_demandCharacteristics](#).

Referenced by [setAll\(\)](#).

24.21.4.20 void TRADEMGEN::DemandStream::setTotalNumberOfRequestsToBeGenerated (const stdair::NbOfRequests_T & *iNbOfRequests*) [inline]

Set the total number of requests to be generated.

Definition at line 166 of file [DemandStream.hpp](#).

References [_totalNumberOfRequestsToBeGenerated](#).

Referenced by [setAll\(\)](#).

24.21.4.21 void TRADEMGEN::DemandStream::setRequestDateTimeRandomGeneratorSeed (const stdair::RandomSeed_T & *iSeed*) [inline]

Set the seed of the random generator for the request datetime.

Definition at line 171 of file [DemandStream.hpp](#).

References [_requestDateTimeRandomGenerator](#).

Referenced by [setAll\(\)](#).

24.21.4.22 void TRADEMGEN::DemandStream::setDemandCharacteristicsRandomGeneratorSeed (const stdair::RandomSeed_T & *iSeed*) [inline]

Set the seed of the random generator for the demand characteristics.

Definition at line 176 of file [DemandStream.hpp](#).

References [_demandCharacteristicsRandomGenerator](#).

Referenced by [setAll\(\)](#).

24.21.4.23 void TRADEMGEN::DemandStream::setPOSProbabilityMass (const POSProbabilityMass_T & iProbMass) [inline]

Set the default POS probability mass, used when "row" (rest of the world) is drawn.

Definition at line 184 of file [DemandStream.hpp](#).

References [_posProMass](#).

Referenced by [setAll\(\)](#).

24.21.4.24 void TRADEMGEN::DemandStream::setAll (const ArrivalPatternCumulativeDistribution_T & iArrivalPattern, const POSProbabilityMassFunction_T & iPOSProbMass, const ChannelProbabilityMassFunction_T & iChannelProbMass, const TripTypeProbabilityMassFunction_T & iTripTypeProbMass, const StayDurationProbabilityMassFunction_T & iStayDurationProbMass, const FrequentFlyerProbabilityMassFunction_T & iFrequentFlyerProbMass, const stdair::ChangeFeesRatio_T & iChangeFeeProb, const stdair::Disutility_T & iChangeFeeDisutility, const stdair::NonRefundableRatio_T & iNonRefundableProb, const stdair::Disutility_T & iNonRefundableDisutility, const PreferredDepartureTimeContinuousDistribution_T & iPreferredDepartureTimeContinuousDistribution, const stdair::WTP_T & iMinWTP, const ValueOfTimeContinuousDistribution_T & iValueOfTimeContinuousDistribution, const DemandDistribution & iDemandDistribution, stdair::BaseGenerator_T & ioSharedGenerator, const stdair::RandomSeed_T & iRequestDateTimeSeed, const stdair::RandomSeed_T & iDemandCharacteristicsSeed, const POSProbabilityMass_T & iDefaultPOSProbabilityMass)

Initialisation.

Definition at line 81 of file [DemandStream.cpp](#).

References [setDemandCharacteristics\(\)](#), [setDemandCharacteristicsRandomGeneratorSeed\(\)](#), [setDemandDistribution\(\)](#), [setPOSProbabilityMass\(\)](#), [setRequestDateTimeRandomGeneratorSeed\(\)](#), and [setTotalNumberOfRequestsToBeGenerated\(\)](#).

24.21.4.25 void TRADEMGEN::DemandStream::setBoolFirstDateTimeRequest (const bool & iFirstDateTimeRequest) [inline]

Set the boolean describing if it is the first time we generate a request for a demand stream.

Definition at line 214 of file [DemandStream.hpp](#).

24.21.4.26 void TRADEMGEN::DemandStream::incrementGeneratedRequestsCounter () [inline]

Increment counter of requests generated so far

Definition at line 222 of file [DemandStream.hpp](#).

References [_randomGenerationContext](#), and [TRADEMGENT::RandomGenerationContext::incrementGeneratedRequestsCounter\(\)](#).

Referenced by [generateTimeOfRequestPoissonProcess\(\)](#), and [generateTimeOfRequestStatisticsOrder\(\)](#).

24.21.4.27 `const bool TRADEMGEN::DemandStream::stillHavingRequestsToBeGenerated (const stdair::DemandGenerationMethod & iDemandGenerationMethod) const`

Check whether enough requests have already been generated.

Definition at line 172 of file [DemandStream.cpp](#).

References [_randomGenerationContext](#), [_totalNumberOfRequestsToBeGenerated](#), and [TRADEMG-
GEN::RandomGenerationContext::getNumberOfRequestsGeneratedSoFar\(\)](#).

24.21.4.28 `const stdair::DateTime_T TRADEMG- GEN::DemandStream::generateTimeOfRequestPoissonProcess ()`

Generate the time of the next request with poisson process.

Definition at line 197 of file [DemandStream.cpp](#).

References [TRADEMG-
GEN::DemandCharacteristics::_arrivalPattern](#), [_demandCharacteristics](#), [_demandDistribution](#), [_key](#), [TRADEMG-
GEN::DemandDistribution::_meanNumberOfRequests](#), [_requestDateTimeRandomGenerator](#), [convertFloatIntoDuration\(\)](#), [TRADEMG-
GEN::DEFAULT_LAST_LOWER_BOUND_ARRIVAL_PATTERN](#), [TRADEMG-
GEN::ContinuousAttributeLite< T >::getDerivativeValue\(\)](#), [TRADEMG-
GEN::DemandStreamKey::getPreferredDepartureDate\(\)](#), [TRADEMG-
GEN::ContinuousAttributeLite< T >::getUpperBound\(\)](#), [TRADEMG-
GEN::ContinuousAttributeLite< T >::getValue\(\)](#), and [incrementGen-
eratedRequestsCounter\(\)](#).

Referenced by [generateNextRequest\(\)](#).

24.21.4.29 `const stdair::DateTime_T TRADEMG- GEN::DemandStream::generateTimeOfRequestStatisticsOrder ()`

Generate the time of the next request with statistics order

Definition at line 299 of file [DemandStream.cpp](#).

References [TRADEMG-
GEN::DemandCharacteristics::_arrivalPattern](#), [_demandCharacteristics](#), [_key](#), [_randomGenerationContext](#), [_requestDateTimeRandomGenerator](#), [_totalNumberOfRequestsToBeGenerated](#), [convertFloatIntoDuration\(\)](#), [TRADEMG-
GEN::RandomGenerationContext::getCumulativeProbabilitySoFar\(\)](#), [TRADEMG-
GEN::RandomGenerationContext::getNumberOfRequestsGeneratedSoFar\(\)](#), [TRADEMG-
GEN::DemandStreamKey::getPreferredDepartureDate\(\)](#), [TRADEMG-
GEN::ContinuousAttributeLite< T >::getValue\(\)](#), [incrementGeneratedRequestsCounter\(\)](#), and [TRADEMG-
GEN::RandomGenerationContext::setCumulativeProbabilitySoFar\(\)](#).

Referenced by [generateNextRequest\(\)](#).

24.21.4.30 `const stdair::AirportCode_T TRADEMG- GEN::DemandStream::generatePOS ()`

Generate the POS.

Definition at line 430 of file [DemandStream.cpp](#).

References [_demandCharacteristics](#), [_demandCharacteristicsRandomGenerator](#), and [TRADEMG-
GEN::DemandCharacteristics::getPOSValue\(\)](#).

Referenced by [generateNextRequest\(\)](#).

24.21.4.31 `const stdair::ChannelLabel_T TRADEMGEN::DemandStream::generateChannel ()`

Generate the reservation channel.

Definition at line 440 of file [DemandStream.cpp](#).

References [TRADEMGEN::DemandCharacteristics::_channelProbabilityMass](#), [_demandCharacteristics](#), [_demandCharacteristicsRandomGenerator](#), and [TRADEMGEN::CategoricalAttributeLite< T >::getValue\(\)](#).

Referenced by [generateNextRequest\(\)](#).

24.21.4.32 `const stdair::TripType_T TRADEMGEN::DemandStream::generateTripType ()`

Generate the trip type.

Definition at line 449 of file [DemandStream.cpp](#).

References [_demandCharacteristics](#), [_demandCharacteristicsRandomGenerator](#), [TRADEMGEN::DemandCharacteristics::_tripTypeProbabilityMass](#), and [TRADEMGEN::CategoricalAttributeLite< T >::getValue\(\)](#).

Referenced by [generateNextRequest\(\)](#).

24.21.4.33 `const stdair::DayDuration_T TRADEMGEN::DemandStream::generateStayDuration ()`

Generate the stay duration.

Definition at line 458 of file [DemandStream.cpp](#).

References [_demandCharacteristics](#), [_demandCharacteristicsRandomGenerator](#), [TRADEMGEN::DemandCharacteristics::_stayDurationProbabilityMass](#), and [TRADEMGEN::CategoricalAttributeLite< T >::getValue\(\)](#).

Referenced by [generateNextRequest\(\)](#).

24.21.4.34 `const stdair::FrequentFlyer_T TRADEMGEN::DemandStream::generateFrequentFlyer ()`

Generate the frequent flyer type.

Definition at line 467 of file [DemandStream.cpp](#).

References [_demandCharacteristics](#), [_demandCharacteristicsRandomGenerator](#), [TRADEMGEN::DemandCharacteristics::_frequentFlyerProbabilityMass](#), and [TRADEMGEN::CategoricalAttributeLite< T >::getValue\(\)](#).

Referenced by [generateNextRequest\(\)](#).

24.21.4.35 `const stdair::ChangeFees_T TRADEMGEN::DemandStream::generateChangeFees ()`

Generate the change fee acceptance.

Definition at line 476 of file [DemandStream.cpp](#).

References [TRADEMGEN::DemandCharacteristics::_changeFeeProb](#), [_demandCharacteristics](#), and [_demandCharacteristicsRandomGenerator](#).

Referenced by [generateNextRequest\(\)](#).

24.21.4.36 `const stdair::NonRefundable_T TRADEMGEN::DemandStream::generateNonRefundable ()`

Generate the non refundable acceptance.

Definition at line 487 of file [DemandStream.cpp](#).

References [_demandCharacteristics](#), [_demandCharacteristicsRandomGenerator](#), and [TRADEMGEN::DemandCharacteristics::_nonRefundableProb](#).

Referenced by [generateNextRequest\(\)](#).

24.21.4.37 `const stdair::Duration_T TRADEMGEN::DemandStream::generatePreferredDepartureTime ()`

Generate the preferred departure time.

Definition at line 498 of file [DemandStream.cpp](#).

References [_demandCharacteristics](#), and [_demandCharacteristicsRandomGenerator](#).

Referenced by [generateNextRequest\(\)](#).

24.21.4.38 `const stdair::WTP_T TRADEMGEN::DemandStream::generateWTP (stdair::RandomGeneration & ioGenerator, const stdair::Date_T & iDepartureDate, const stdair::DateTime_T & iDateTimeThisRequest, const stdair::DayDuration_T & iDurationOfStay)`

Generate the WTP.

Definition at line 512 of file [DemandStream.cpp](#).

References [_demandCharacteristics](#), [TRADEMGEN::DemandCharacteristics::_frat5Pattern](#), [TRADEMGEN::DemandCharacteristics::_minWTP](#), and [TRADEMGEN::ContinuousAttributeLite< T >::getValue\(\)](#).

Referenced by [generateNextRequest\(\)](#).

24.21.4.39 `const stdair::PriceValue_T TRADEMGEN::DemandStream::generateValueOfTime ()`

Generate the value of time.

Definition at line 531 of file [DemandStream.cpp](#).

References [_demandCharacteristics](#), [_demandCharacteristicsRandomGenerator](#), [TRADEMGEN::DemandCharacteristics::_valueOfTimeCumulativeDistribution](#), and [TRADEMGEN::ContinuousAttributeLite< T >::getValue\(\)](#).

Referenced by [generateNextRequest\(\)](#).

24.21.4.40 `stdair::BookingRequestPtr_T TRADEMGEN::DemandStream::generateNextRequest (stdair::RandomGeneration & ioGenerator, const stdair::DemandGenerationMethod & iDemandGenerationMethod)`

Generate the next request.

Parameters:

stdair::RandomGeneration Random generator.

const stdair::DemandGenerationMethod::EN_DemandGenerationMethod Method used to generate the date time of the next booking request: statistic order or poisson process.

Returns:

`stdair::BookingRequestPtr_T` Next request to be simulate.

Definition at line 541 of file [DemandStream.cpp](#).

References [TRADEMGEN::DemandCharacteristics::_changeFeeDisutility](#), [_demandCharacteristics](#), [_key](#), [TRADEMGEN::DemandCharacteristics::_nonRefundableDisutility](#), [describeKey\(\)](#), [generateChangeFees\(\)](#), [generateChannel\(\)](#), [generateFrequentFlyer\(\)](#), [generateNonRefundable\(\)](#), [generatePOS\(\)](#), [generatePreferredDepartureTime\(\)](#), [generateStayDuration\(\)](#), [generateTimeOfRequestPoissonProcess\(\)](#), [generateTimeOfRequestStatisticsOrder\(\)](#), [generateTripType\(\)](#), [generateValueOfTime\(\)](#), [generateWTP\(\)](#), [TRADEMGEN::DemandStreamKey::getDestination\(\)](#), [TRADEMGEN::DemandStreamKey::getOrigin\(\)](#), [TRADEMGEN::DemandStreamKey::getPreferredCabin\(\)](#), and [TRADEMGEN::DemandStreamKey::getPreferredDepartureDate\(\)](#).

24.21.4.41 `void TRADEMGEN::DemandStream::reset (stdair::BaseGenerator_T & ioSharedGenerator)`

Reset all the contexts of the demand stream.

Definition at line 625 of file [DemandStream.cpp](#).

References [_randomGenerationContext](#), and [TRADEMGEN::RandomGenerationContext::reset\(\)](#).

24.21.4.42 `void TRADEMGEN::DemandStream::toStream (std::ostream & ioOut) const [inline]`

Dump a Business Object into an output stream.

Parameters:

ostream& the output stream.

Definition at line 292 of file [DemandStream.hpp](#).

References [toString\(\)](#).

24.21.4.43 `void TRADEMGEN::DemandStream::fromStream (std::istream & ioIn) [inline]`

Read a Business Object from an input stream.

Parameters:

istream& the input stream.

Definition at line 300 of file [DemandStream.hpp](#).

24.21.4.44 std::string TRADEMGEN::DemandStream::toString () const

Get the serialised version of the Business Object.

Definition at line 73 of file [DemandStream.cpp](#).

References [_key](#), and [TRADEMGENT::DemandStreamKey::toString\(\)](#).

Referenced by [toStream\(\)](#).

24.21.4.45 const std::string TRADEMGEN::DemandStream::describeKey () const [inline]

Get a string describing the key.

Definition at line 311 of file [DemandStream.hpp](#).

References [_key](#), and [TRADEMGENT::DemandStreamKey::toString\(\)](#).

Referenced by [TRADEMGENT::TRADEMGENT_Service::displayDemandStream\(\)](#), and [generateNextRequest\(\)](#).

24.21.4.46 std::string TRADEMGEN::DemandStream::display () const

Dump recursively the content of the [DemandStream](#) object.

Definition at line 119 of file [DemandStream.cpp](#).

References [_demandCharacteristics](#), [_demandCharacteristicsRandomGenerator](#), [_demandDistribution](#), [_key](#), [_posProMass](#), [_randomGenerationContext](#), [_requestDateTimeRandomGenerator](#), [_totalNumberOfRequestsToBeGenerated](#), [TRADEMGENT::DemandDistribution::describe\(\)](#), [TRADEMGENT::DemandCharacteristics::describe\(\)](#), [TRADEMGENT::CategoricalAttributeLite< T >::displayProbabilityMass\(\)](#), and [TRADEMGENT::DemandStreamKey::toString\(\)](#).

Referenced by [TRADEMGENT::BomDisplay::csvDisplay\(\)](#).

24.21.4.47 const std::air::Duration_T TRADEMGEN::DemandStream::convertFloatIntoDuration (const std::air::FloatDuration_T iNumberOfDays)

Definition at line 401 of file [DemandStream.cpp](#).

Referenced by [generateTimeOfRequestPoissonProcess\(\)](#), and [generateTimeOfRequestStatisticsOrder\(\)](#).

24.21.5 Friends And Related Function Documentation

24.21.5.1 friend class std::air::FacBom [friend]

Definition at line 31 of file [DemandStream.hpp](#).

24.21.5.2 friend class stdair::FacBomManager [friend]

Definition at line 32 of file [DemandStream.hpp](#).

24.21.6 Member Data Documentation

24.21.6.1 Key_T TRADEMGEN::DemandStream::_key [protected]

Primary key (string gathering the origin, destination, POS and date).

Definition at line 346 of file [DemandStream.hpp](#).

Referenced by [describeKey\(\)](#), [display\(\)](#), [generateNextRequest\(\)](#), [generateTimeOfRequestPoissonProcess\(\)](#), [generateTimeOfRequestStatisticsOrder\(\)](#), [getDestination\(\)](#), [getKey\(\)](#), [getOrigin\(\)](#), [getPreferredCabin\(\)](#), [getPreferredDepartureDate\(\)](#), and [toString\(\)](#).

24.21.6.2 BomAbstract* TRADEMGEN::DemandStream::_parent [protected]

Pointer on the parent class (EventQueue).

Definition at line 351 of file [DemandStream.hpp](#).

Referenced by [getParent\(\)](#).

24.21.6.3 stdair::HolderMap_T TRADEMGEN::DemandStream::_holderMap [protected]

Map holding the children (not used for now).

Definition at line 356 of file [DemandStream.hpp](#).

Referenced by [getHolderMap\(\)](#).

24.21.6.4 DemandCharacteristics TRADEMGEN::DemandStream::_demandCharacteristics [protected]

Demand characteristics.

Definition at line 361 of file [DemandStream.hpp](#).

Referenced by [display\(\)](#), [generateChangeFees\(\)](#), [generateChannel\(\)](#), [generateFrequentFlyer\(\)](#), [generateNextRequest\(\)](#), [generateNonRefundable\(\)](#), [generatePOS\(\)](#), [generatePreferredDepartureTime\(\)](#), [generateStayDuration\(\)](#), [generateTimeOfRequestPoissonProcess\(\)](#), [generateTimeOfRequestStatisticsOrder\(\)](#), [generateTripType\(\)](#), [generateValueOfTime\(\)](#), [generateWTP\(\)](#), [getChangeFeeDisutility\(\)](#), [getDemandCharacteristics\(\)](#), [getNonRefundableDisutility\(\)](#), and [setDemandCharacteristics\(\)](#).

24.21.6.5 DemandDistribution TRADEMGEN::DemandStream::_demandDistribution [protected]

Demand distribution.

Definition at line 366 of file [DemandStream.hpp](#).

Referenced by [display\(\)](#), [generateTimeOfRequestPoissonProcess\(\)](#), [getDemandDistribution\(\)](#), [getMeanNumberOfRequests\(\)](#), [getStdDevNumberOfRequests\(\)](#), and [setDemandDistribution\(\)](#).

24.21.6.6 stdair::NbOfRequests_T TRADEMGEN::DemandStream::_-totalNumberOfRequestsToBeGenerated [protected]

Total number of requests to be generated.

Definition at line 371 of file [DemandStream.hpp](#).

Referenced by [display\(\)](#), [generateTimeOfRequestStatisticsOrder\(\)](#), [getTotalNumberOfRequestsToBeGenerated\(\)](#), [setTotalNumberOfRequestsToBeGenerated\(\)](#), and [stillHavingRequestsToBeGenerated\(\)](#).

24.21.6.7 RandomGenerationContext TRADEMGEN::DemandStream::_-randomGenerationContext [protected]

Random generation context.

Definition at line 376 of file [DemandStream.hpp](#).

Referenced by [display\(\)](#), [generateTimeOfRequestStatisticsOrder\(\)](#), [getNumberOfRequestsGeneratedSoFar\(\)](#), [incrementGeneratedRequestsCounter\(\)](#), [reset\(\)](#), [setNumberOfRequestsGeneratedSoFar\(\)](#), and [stillHavingRequestsToBeGenerated\(\)](#).

24.21.6.8 stdair::RandomGeneration TRADEMGEN::DemandStream::_-requestDateTimeRandomGenerator [protected]

Random generator for request date-time.

Definition at line 381 of file [DemandStream.hpp](#).

Referenced by [display\(\)](#), [generateTimeOfRequestPoissonProcess\(\)](#), [generateTimeOfRequestStatisticsOrder\(\)](#), and [setRequestDateTimeRandomGeneratorSeed\(\)](#).

24.21.6.9 stdair::RandomGeneration TRADEMGEN::DemandStream::_-demandCharacteristicsRandomGenerator [protected]

Random generator for demand characteristics.

Definition at line 386 of file [DemandStream.hpp](#).

Referenced by [display\(\)](#), [generateChangeFees\(\)](#), [generateChannel\(\)](#), [generateFrequentFlyer\(\)](#), [generateNonRefundable\(\)](#), [generatePOS\(\)](#), [generatePreferredDepartureTime\(\)](#), [generateStayDuration\(\)](#), [generateTripType\(\)](#), [generateValueOfTime\(\)](#), and [setDemandCharacteristicsRandomGeneratorSeed\(\)](#).

24.21.6.10 POSProbabilityMass_T TRADEMGEN::DemandStream::_-posProMass [protected]

Default POS probability mass, used when "row" (rest of the world) is drawn.

Definition at line 392 of file [DemandStream.hpp](#).

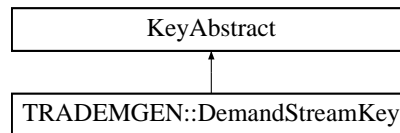
Referenced by [display\(\)](#), [getPOSProbabilityMass\(\)](#), and [setPOSProbabilityMass\(\)](#).

The documentation for this class was generated from the following files:

- [trademgen/bom/DemandStream.hpp](#)
- [trademgen/bom/DemandStream.cpp](#)

24.22 TRADEMGEN::DemandStreamKey Struct Reference

#include <trademgen/bom/DemandStreamKey.hpp> Inheritance diagram for TRADEMGEN::DemandStreamKey:



Public Member Functions

- [DemandStreamKey](#) (const stdair::AirportCode_T &iOrigin, const stdair::AirportCode_T &iDestination, const stdair::Date_T &iPreferredDepartureDate, const stdair::CabinCode_T &iPreferredCabin)
- [DemandStreamKey](#) (const [DemandStreamKey](#) &)
- [~DemandStreamKey](#) ()
- const stdair::AirportCode_T & [getOrigin](#) () const
- const stdair::AirportCode_T & [getDestination](#) () const
- const stdair::Date_T & [getPreferredDepartureDate](#) () const
- const stdair::CabinCode_T & [getPreferredCabin](#) () const
- void [toStream](#) (std::ostream &ioOut) const
- void [fromStream](#) (std::istream &ioIn)
- const std::string [toString](#) () const

24.22.1 Detailed Description

Key of a given demand-stream, made of a pair of required airports/cities (origin and destination), a preferred departure date and a preferred cabin. Those attributes correspond to a the travel requirements of a simulated traveller.

Definition at line 20 of file [DemandStreamKey.hpp](#).

24.22.2 Constructor & Destructor Documentation

24.22.2.1 TRADEMGEN::DemandStreamKey::DemandStreamKey (const stdair::AirportCode_T & iOrigin, const stdair::AirportCode_T & iDestination, const stdair::Date_T & iPreferredDepartureDate, const stdair::CabinCode_T & iPreferredCabin)

Constructor.

Definition at line 25 of file [DemandStreamKey.cpp](#).

24.22.2.2 TRADEMGEN::DemandStreamKey::DemandStreamKey (const DemandStreamKey & iKey)

Default copy constructor.

Definition at line 35 of file [DemandStreamKey.cpp](#).

24.22.2.3 TRADEMGEN::DemandStreamKey::~~DemandStreamKey ()

Destructor.

Definition at line 42 of file [DemandStreamKey.cpp](#).**24.22.3 Member Function Documentation****24.22.3.1 const stdair::AirportCode_T& TRADEMGEN::DemandStreamKey::getOrigin () const [inline]**

Get the origin.

Definition at line 43 of file [DemandStreamKey.hpp](#).Referenced by [TRADEMGEN::DemandStream::generateNextRequest\(\)](#), and [TRADEMGEN::DemandStream::getOrigin\(\)](#).**24.22.3.2 const stdair::AirportCode_T& TRADEMGEN::DemandStreamKey::getDestination () const [inline]**

Get the destination.

Definition at line 48 of file [DemandStreamKey.hpp](#).Referenced by [TRADEMGEN::DemandStream::generateNextRequest\(\)](#), and [TRADEMGEN::DemandStream::getDestination\(\)](#).**24.22.3.3 const stdair::Date_T& TRADEMGEN::DemandStreamKey::getPreferredDepartureDate () const [inline]**

Get the preferred departure date.

Definition at line 53 of file [DemandStreamKey.hpp](#).Referenced by [TRADEMGEN::DemandStream::generateNextRequest\(\)](#), [TRADEMGEN::DemandStream::generateTimeOfRequestPoissonProcess\(\)](#), [TRADEMGEN::DemandStream::generateTimeOfRequestStatisticsOrder\(\)](#), and [TRADEMGEN::DemandStream::getPreferredDepartureDate\(\)](#).**24.22.3.4 const stdair::CabinCode_T& TRADEMGEN::DemandStreamKey::getPreferredCabin () const [inline]**

Get the preferred cabin.

Definition at line 58 of file [DemandStreamKey.hpp](#).Referenced by [TRADEMGEN::DemandStream::generateNextRequest\(\)](#), and [TRADEMGEN::DemandStream::getPreferredCabin\(\)](#).**24.22.3.5 void TRADEMGEN::DemandStreamKey::toStream (std::ostream & ioOut) const**

Dump a Business Object Key into an output stream.

Parameters:

ostream& the output stream.

Definition at line 46 of file [DemandStreamKey.cpp](#).

References [toString\(\)](#).

24.22.3.6 void TRADEMGEN::DemandStreamKey::fromStream (std::istream & ioIn)

Read a Business Object Key from an input stream.

Parameters:

istream& the input stream.

Definition at line 51 of file [DemandStreamKey.cpp](#).

24.22.3.7 const std::string TRADEMGEN::DemandStreamKey::toString () const

Get the serialised version of the Business Object Key.

That string is unique, at the level of a given Business Object, when among children of a given parent Business Object.

For instance, "H" and "K" allow to differentiate among two marketing classes for the same segment-stream.

Definition at line 55 of file [DemandStreamKey.cpp](#).

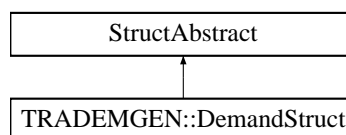
Referenced by [TRADEMGEN::DemandStream::describeKey\(\)](#), [TRADEMGEN::DemandStream::display\(\)](#), [toStream\(\)](#), and [TRADEMGEN::DemandStream::toString\(\)](#).

The documentation for this struct was generated from the following files:

- [trademgen/bom/DemandStreamKey.hpp](#)
- [trademgen/bom/DemandStreamKey.cpp](#)

24.23 TRADEMGEN::DemandStruct Struct Reference

`#include <trademgen/bom/DemandStruct.hpp>` Inheritance diagram for TRADEMGEN::DemandStruct:

**Public Member Functions**

- `stdair::Date_T` [getDate \(\)](#) const
- `stdair::Duration_T` [getTime \(\)](#) const
- `const std::string` [describe \(\)](#) const
- [DemandStruct \(\)](#)
- [~DemandStruct \(\)](#)

Public Attributes

- stdair::DatePeriod_T [_dateRange](#)
- stdair::DoWStruct [_dow](#)
- stdair::AirportCode_T [_origin](#)
- stdair::AirportCode_T [_destination](#)
- stdair::CabinCode_T [_prefCabin](#)
- stdair::MeanValue_T [_demandMean](#)
- stdair::StdDevValue_T [_demandStdDev](#)
- stdair::ChangeFeesRatio_T [_changeFeeProb](#)
- stdair::Disutility_T [_changeFeeDisutility](#)
- stdair::NonRefundableRatio_T [_nonRefundableProb](#)
- stdair::Disutility_T [_nonRefundableDisutility](#)
- POSProbabilityMassFunction_T [_posProbDist](#)
- ChannelProbabilityMassFunction_T [_channelProbDist](#)
- TripTypeProbabilityMassFunction_T [_tripProbDist](#)
- StayDurationProbabilityMassFunction_T [_stayProbDist](#)
- FrequentFlyerProbabilityMassFunction_T [_ffProbDist](#)
- PreferredDepartureTimeContinuousDistribution_T [_prefDepTimeProbDist](#)
- stdair::WTP_T [_minWTP](#)
- ValueOfTimeContinuousDistribution_T [_timeValueProbDist](#)
- ArrivalPatternCumulativeDistribution_T [_dtdProbDist](#)
- stdair::Date_T [_prefDepDateStart](#)
- stdair::Date_T [_prefDepDateEnd](#)
- unsigned int [_itYear](#)
- unsigned int [_itMonth](#)
- unsigned int [_itDay](#)
- long [_itHours](#)
- long [_itMinutes](#)
- long [_itSeconds](#)
- stdair::AirportCode_T [_itPosCode](#)
- stdair::ChannelLabel_T [_itChannelCode](#)
- stdair::TripType_T [_itTripCode](#)
- stdair::DayDuration_T [_itStayDuration](#)
- stdair::FrequentFlyer_T [_itFFCode](#)
- stdair::Duration_T [_itPrefDepTime](#)
- stdair::PriceValue_T [_itTimeValue](#)
- stdair::DayDuration_T [_itDTD](#)

24.23.1 Detailed Description

Utility Structure for the parsing of Demand structures.

Definition at line 21 of file [DemandStruct.hpp](#).

24.23.2 Constructor & Destructor Documentation**24.23.2.1 TRADEMGEN::DemandStruct::DemandStruct ()**

Default constructor.

Definition at line 18 of file [DemandStruct.cpp](#).

24.23.2.2 TRADEMGEN::DemandStruct::~~DemandStruct ()

Destructor

Definition at line 26 of file [DemandStruct.cpp](#).

24.23.3 Member Function Documentation**24.23.3.1 stdair::Date_T TRADEMGEN::DemandStruct::getDate () const**

Get the date from the staging details.

Definition at line 30 of file [DemandStruct.cpp](#).

References [_itDay](#), [_itMonth](#), and [_itYear](#).

Referenced by [TRADEMGENT::DemandParserHelper::storePrefDepDateRangeEnd::operator\(\)\(\)](#), and [TRADEMGENT::DemandParserHelper::storePrefDepDateRangeStart::operator\(\)\(\)](#).

24.23.3.2 stdair::Duration_T TRADEMGEN::DemandStruct::getTime () const

Get the time from the staging details.

Definition at line 35 of file [DemandStruct.cpp](#).

References [_itHours](#), [_itMinutes](#), and [_itSeconds](#).

Referenced by [TRADEMGENT::DemandParserHelper::storePrefDepTime::operator\(\)\(\)](#).

24.23.3.3 const std::string TRADEMGEN::DemandStruct::describe () const

Give a description of the structure (for display purposes).

Definition at line 42 of file [DemandStruct.cpp](#).

References [_changeFeeProb](#), [_channelProbDist](#), [_dateRange](#), [_demandMean](#), [_demandStdDev](#), [_destination](#), [_dow](#), [_dtdProbDist](#), [_ffProbDist](#), [_minWTP](#), [_nonRefundableProb](#), [_origin](#), [_posProbDist](#), [_prefCabin](#), [_prefDepTimeProbDist](#), [_stayProbDist](#), [_timeValueProbDist](#), and [_tripProbDist](#).

24.23.4 Member Data Documentation**24.23.4.1 stdair::DatePeriod_T TRADEMGEN::DemandStruct::_dateRange**

Definition at line 51 of file [DemandStruct.hpp](#).

Referenced by [describe\(\)](#), and [TRADEMGENT::DemandParserHelper::storePrefDepDateRangeEnd::operator\(\)\(\)](#).

24.23.4.2 stdair::DoWStruct TRADEMGEN::DemandStruct::_dow

Definition at line 52 of file [DemandStruct.hpp](#).

Referenced by [describe\(\)](#), and [TRADEMGENT::DemandParserHelper::storeDow::operator\(\)\(\)](#).

24.23.4.3 stdair::AirportCode_T TRADEMGEN::DemandStruct::_origin

Definition at line 53 of file [DemandStruct.hpp](#).

Referenced by [describe\(\)](#), and [TRADEMGEN::DemandParserHelper::storeOrigin::operator\(\)](#).

24.23.4.4 stdair::AirportCode_T TRADEMGEN::DemandStruct::_destination

Definition at line 54 of file [DemandStruct.hpp](#).

Referenced by [describe\(\)](#), and [TRADEMGEN::DemandParserHelper::storeDestination::operator\(\)](#).

24.23.4.5 stdair::CabinCode_T TRADEMGEN::DemandStruct::_prefCabin

Definition at line 55 of file [DemandStruct.hpp](#).

Referenced by [describe\(\)](#), and [TRADEMGEN::DemandParserHelper::storePrefCabin::operator\(\)](#).

24.23.4.6 stdair::MeanValue_T TRADEMGEN::DemandStruct::_demandMean

Definition at line 56 of file [DemandStruct.hpp](#).

Referenced by [describe\(\)](#), and [TRADEMGEN::DemandParserHelper::storeDemandMean::operator\(\)](#).

24.23.4.7 stdair::StdDevValue_T TRADEMGEN::DemandStruct::_demandStdDev

Definition at line 57 of file [DemandStruct.hpp](#).

Referenced by [describe\(\)](#), and [TRADEMGEN::DemandParserHelper::storeDemandStdDev::operator\(\)](#).

24.23.4.8 stdair::ChangeFeesRatio_T TRADEMGEN::DemandStruct::_changeFeeProb

Definition at line 58 of file [DemandStruct.hpp](#).

Referenced by [describe\(\)](#), and [TRADEMGEN::DemandParserHelper::storeDemandChangeFeeProb::operator\(\)](#).

24.23.4.9 stdair::Disutility_T TRADEMGEN::DemandStruct::_changeFeeDisutility

Definition at line 59 of file [DemandStruct.hpp](#).

Referenced by [TRADEMGEN::DemandParserHelper::storeDemandChangeFeeDisutility::operator\(\)](#).

24.23.4.10 stdair::NonRefundableRatio_T TRADEMGEN::DemandStruct::_nonRefundableProb

Definition at line 60 of file [DemandStruct.hpp](#).

Referenced by [describe\(\)](#), and [TRADEMGENT::DemandParserHelper::storeDemandNonRefundableProb::operator\(\)](#).

24.23.4.11 stdair::Disutility_T TRADEMGEN::DemandStruct::_nonRefundableDisutility

Definition at line 61 of file [DemandStruct.hpp](#).

Referenced by [TRADEMGENT::DemandParserHelper::storeDemandNonRefundableDisutility::operator\(\)](#).

24.23.4.12 POSProbabilityMassFunction_T TRADEMGEN::DemandStruct::_posProbDist

Definition at line 62 of file [DemandStruct.hpp](#).

Referenced by [describe\(\)](#), [TRADEMGENT::DemandParserHelper::doEndDemand::operator\(\)](#), and [TRADEMGENT::DemandParserHelper::storePosProbMass::operator\(\)](#).

24.23.4.13 ChannelProbabilityMassFunction_T TRADEMGEN::DemandStruct::_channelProbDist

Definition at line 63 of file [DemandStruct.hpp](#).

Referenced by [describe\(\)](#), [TRADEMGENT::DemandParserHelper::doEndDemand::operator\(\)](#), and [TRADEMGENT::DemandParserHelper::storeChannelProbMass::operator\(\)](#).

24.23.4.14 TripTypeProbabilityMassFunction_T TRADEMGEN::DemandStruct::_tripProbDist

Definition at line 64 of file [DemandStruct.hpp](#).

Referenced by [describe\(\)](#), [TRADEMGENT::DemandParserHelper::doEndDemand::operator\(\)](#), and [TRADEMGENT::DemandParserHelper::storeTripProbMass::operator\(\)](#).

24.23.4.15 StayDurationProbabilityMassFunction_T TRADEMGEN::DemandStruct::_stayProbDist

Definition at line 65 of file [DemandStruct.hpp](#).

Referenced by [describe\(\)](#), [TRADEMGENT::DemandParserHelper::doEndDemand::operator\(\)](#), and [TRADEMGENT::DemandParserHelper::storeStayProbMass::operator\(\)](#).

24.23.4.16 FrequentFlyerProbabilityMassFunction_T TRADEMGEN::DemandStruct::_ffProbDist

Definition at line 66 of file [DemandStruct.hpp](#).

Referenced by [describe\(\)](#), [TRADEMGENT::DemandParserHelper::doEndDemand::operator\(\)](#), and [TRADEMGENT::DemandParserHelper::storeFFProbMass::operator\(\)](#).

24.23.4.17 PreferredDepartureTimeContinuousDistribution_T TRADEMGEN::DemandStruct::_prefDepTimeProbDist

Definition at line 67 of file [DemandStruct.hpp](#).

Referenced by [describe\(\)](#), [TRADEMGENT::DemandParserHelper::doEndDemand::operator\(\)](#), and [TRADEMGENT::DemandParserHelper::storePrefDepTimeProbMass::operator\(\)](#).

24.23.4.18 stdair::WTP_T TRADEMGEN::DemandStruct::_minWTP

Definition at line 68 of file [DemandStruct.hpp](#).

Referenced by [describe\(\)](#), and [TRADEMGENT::DemandParserHelper::storeWTP::operator\(\)](#).

24.23.4.19 ValueOfTimeContinuousDistribution_T TRADEMGEN::DemandStruct::_timeValueProbDist

Definition at line 69 of file [DemandStruct.hpp](#).

Referenced by [describe\(\)](#), [TRADEMGENT::DemandParserHelper::doEndDemand::operator\(\)](#), and [TRADEMGENT::DemandParserHelper::storeTimeValueProbMass::operator\(\)](#).

24.23.4.20 ArrivalPatternCumulativeDistribution_T TRADEMGEN::DemandStruct::_dtdProbDist

Definition at line 70 of file [DemandStruct.hpp](#).

Referenced by [describe\(\)](#), [TRADEMGENT::DemandParserHelper::doEndDemand::operator\(\)](#), and [TRADEMGENT::DemandParserHelper::storeDTDProbMass::operator\(\)](#).

24.23.4.21 stdair::Date_T TRADEMGEN::DemandStruct::_prefDepDateStart

Staging Date.

Definition at line 75 of file [DemandStruct.hpp](#).

Referenced by [TRADEMGENT::DemandParserHelper::storePrefDepDateRangeEnd::operator\(\)](#), and [TRADEMGENT::DemandParserHelper::storePrefDepDateRangeStart::operator\(\)](#).

24.23.4.22 stdair::Date_T TRADEMGEN::DemandStruct::_prefDepDateEnd

Definition at line 76 of file [DemandStruct.hpp](#).

Referenced by [TRADEMGENT::DemandParserHelper::storePrefDepDateRangeEnd::operator\(\)\(\)](#).

24.23.4.23 unsigned int TRADEMGEN::DemandStruct::_itYear

Definition at line 77 of file [DemandStruct.hpp](#).

Referenced by [TRADEMGENT::DemandParserHelper::DemandParser::definition< ScannerT >::definition\(\)](#), and [getDate\(\)](#).

24.23.4.24 unsigned int TRADEMGEN::DemandStruct::_itMonth

Definition at line 78 of file [DemandStruct.hpp](#).

Referenced by [TRADEMGENT::DemandParserHelper::DemandParser::definition< ScannerT >::definition\(\)](#), and [getDate\(\)](#).

24.23.4.25 unsigned int TRADEMGEN::DemandStruct::_itDay

Definition at line 79 of file [DemandStruct.hpp](#).

Referenced by [TRADEMGENT::DemandParserHelper::DemandParser::definition< ScannerT >::definition\(\)](#), and [getDate\(\)](#).

24.23.4.26 long TRADEMGEN::DemandStruct::_itHours

Staging Time.

Definition at line 82 of file [DemandStruct.hpp](#).

Referenced by [TRADEMGENT::DemandParserHelper::DemandParser::definition< ScannerT >::definition\(\)](#), and [getTime\(\)](#).

24.23.4.27 long TRADEMGEN::DemandStruct::_itMinutes

Definition at line 83 of file [DemandStruct.hpp](#).

Referenced by [TRADEMGENT::DemandParserHelper::DemandParser::definition< ScannerT >::definition\(\)](#), [getTime\(\)](#), and [TRADEMGENT::DemandParserHelper::storePrefDepTime::operator\(\)\(\)](#).

24.23.4.28 long TRADEMGEN::DemandStruct::_itSeconds

Definition at line 84 of file [DemandStruct.hpp](#).

Referenced by [TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT >::definition\(\)](#), [getTime\(\)](#), [TRADEMGEN::DemandParserHelper::storePrefDepTime::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storePrefDepDateRangeEnd::operator\(\)](#), and [TRADEMGEN::DemandParserHelper::storePrefDepDateRangeStart::operator\(\)](#).

24.23.4.29 stdair::AirportCode_T TRADEMGEN::DemandStruct::_itPosCode

Staging Point-Of-Sale (POS) code.

Definition at line 87 of file [DemandStruct.hpp](#).

Referenced by [TRADEMGEN::DemandParserHelper::storePosProbMass::operator\(\)](#), and [TRADEMGEN::DemandParserHelper::storePosCode::operator\(\)](#).

24.23.4.30 stdair::ChannelLabel_T TRADEMGEN::DemandStruct::_itChannelCode

Staging channel type code.

Definition at line 90 of file [DemandStruct.hpp](#).

Referenced by [TRADEMGEN::DemandParserHelper::storeChannelProbMass::operator\(\)](#), and [TRADEMGEN::DemandParserHelper::storeChannelCode::operator\(\)](#).

24.23.4.31 stdair::TripType_T TRADEMGEN::DemandStruct::_itTripCode

Staging trip type code.

Definition at line 93 of file [DemandStruct.hpp](#).

Referenced by [TRADEMGEN::DemandParserHelper::storeTripProbMass::operator\(\)](#), and [TRADEMGEN::DemandParserHelper::storeTripCode::operator\(\)](#).

24.23.4.32 stdair::DayDuration_T TRADEMGEN::DemandStruct::_itStayDuration

Staging stay duration.

Definition at line 96 of file [DemandStruct.hpp](#).

Referenced by [TRADEMGEN::DemandParserHelper::storeStayProbMass::operator\(\)](#), and [TRADEMGEN::DemandParserHelper::storeStayCode::operator\(\)](#).

24.23.4.33 stdair::FrequentFlyer_T TRADEMGEN::DemandStruct::_itFFCode

Staging Frequent Flyer code.

Definition at line 99 of file [DemandStruct.hpp](#).

Referenced by [TRADEMGEN::DemandParserHelper::storeFFProbMass::operator\(\)](#), and [TRADEMGEN::DemandParserHelper::storeFFCode::operator\(\)](#).

24.23.4.34 stdair::Duration_T TRADEMGEN::DemandStruct::_itPrefDepTime

Staging preferred departure time.

Definition at line 102 of file [DemandStruct.hpp](#).

Referenced by [TRADEMGEN::DemandParserHelper::storePrefDepTimeProbMass::operator\(\)](#), and [TRADEMGEN::DemandParserHelper::storePrefDepTime::operator\(\)](#).

24.23.4.35 stdair::PriceValue_T TRADEMGEN::DemandStruct::_itTimeValue

Staging time value.

Definition at line 105 of file [DemandStruct.hpp](#).

Referenced by [TRADEMGEN::DemandParserHelper::storeTimeValueProbMass::operator\(\)](#), and [TRADEMGEN::DemandParserHelper::storeTimeValue::operator\(\)](#).

24.23.4.36 stdair::DayDuration_T TRADEMGEN::DemandStruct::_itDTD

Staging DTD (Days-To-Departure).

Definition at line 108 of file [DemandStruct.hpp](#).

Referenced by [TRADEMGEN::DemandParserHelper::storeDTDProbMass::operator\(\)](#), and [TRADEMGEN::DemandParserHelper::storeDTD::operator\(\)](#).

The documentation for this struct was generated from the following files:

- trademgen/bom/[DemandStruct.hpp](#)
- trademgen/bom/[DemandStruct.cpp](#)

24.24 TRADEMGEN::DictionaryManager Class Reference

Class wrapper of dictionary business methods.

```
#include <trademgen/basic/DictionaryManager.hpp>
```

Static Public Member Functions

- static const stdair::Probability_T [keyToValue](#) (const [DictionaryKey_T](#))
- static const [DictionaryKey_T](#) [valueToKey](#) (const stdair::Probability_T)

24.24.1 Detailed Description

Class wrapper of dictionary business methods.

Definition at line 21 of file [DictionaryManager.hpp](#).

24.24.2 Member Function Documentation

24.24.2.1 const stdair::Probability_T TRADEMGEN::DictionaryManager::keyToValue (const DictionaryKey_T iKey) [static]

Convert from key to value.

Definition at line 10 of file [DictionaryManager.cpp](#).

Referenced by [TRADEMGEN::ContinuousAttributeLite< stdair::FloatDuration_T >::displayCumulativeDistribution\(\)](#), [TRADEMGEN::ContinuousAttribute< T >::displayCumulativeDistribution\(\)](#), [TRADEMGEN::ContinuousAttribute< T >::displayInverseCumulativeDistribution\(\)](#), [TRADEMGEN::CategoricalAttributeLite< stdair::TripType_T >::displayProbabilityMass\(\)](#), [TRADEMGEN::ContinuousAttributeLite< stdair::FloatDuration_T >::getDerivativeValue\(\)](#), [TRADEMGEN::ContinuousAttributeLite< stdair::FloatDuration_T >::getValue\(\)](#), and [TRADEMGEN::ContinuousAttribute< T >::getValue\(\)](#).

24.24.2.2 const DictionaryKey_T TRADEMGEN::DictionaryManager::valueToKey (const stdair::Probability_T iValue) [static]

Convert from value to key.

Definition at line 17 of file [DictionaryManager.cpp](#).

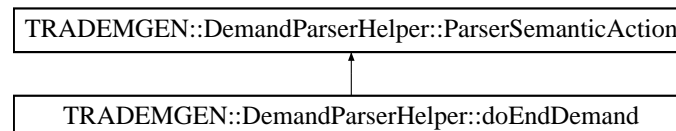
Referenced by [TRADEMGEN::ContinuousAttributeLite< stdair::FloatDuration_T >::getValue\(\)](#), [TRADEMGEN::ContinuousAttribute< T >::getValue\(\)](#), and [TRADEMGEN::CategoricalAttributeLite< stdair::TripType_T >::getValue\(\)](#).

The documentation for this class was generated from the following files:

- [trademgen/basic/DictionaryManager.hpp](#)
- [trademgen/basic/DictionaryManager.cpp](#)

24.25 TRADEMGEN::DemandParserHelper::doEndDemand Struct Reference

`#include <trademgen/command/DemandParserHelper.hpp>`Inheritance diagram for TRADEMGEN::DemandParserHelper::doEndDemand::



Public Member Functions

- [doEndDemand](#) (SEVMGR::SEVMGR_ServicePtr_T, stdair::RandomGeneration &, const POSProbabilityMass_T &, DemandStruct &)
- [void operator\(\)](#) (iterator_t iStr, iterator_t iStrEnd) const

Public Attributes

- SEVMGR::SEVMGR_ServicePtr_T [_sevmgrServicePtr](#)
- stdair::RandomGeneration & [_uniformGenerator](#)
- const POSProbabilityMass_T & [_posProbabilityMass](#)
- DemandStruct & [_demand](#)

24.25.1 Detailed Description

Mark the end of the demand parsing.

Definition at line 275 of file [DemandParserHelper.hpp](#).

24.25.2 Constructor & Destructor Documentation

24.25.2.1 TRADEMGEN::DemandParserHelper::doEndDemand::doEndDemand
(SEVMGR::SEVMGR_ServicePtr_T *ioSEVMGR_ServicePtr*,
stdair::RandomGeneration & *ioSharedGenerator*, const POSProbabilityMass_T &
iPOSProbMass, DemandStruct & *ioDemand*)

Actor Constructor.

Definition at line 457 of file [DemandParserHelper.cpp](#).

24.25.3 Member Function Documentation

24.25.3.1 void TRADEMGEN::DemandParserHelper::doEndDemand::operator() (*iterator_t iStr*,
iterator_t iStrEnd) const

Actor Function (functor).

Definition at line 469 of file [DemandParserHelper.cpp](#).

References [TRADEMGEN::DemandStruct::_channelProbDist](#), [TRADEMGEN::DemandParserHelper::ParserSemanticAction::demand](#), [TRADEMGEN::DemandStruct::_dtdProbDist](#), [TRADEMGEN::DemandStruct::_ffProbDist](#), [TRADEMGEN::DemandStruct::_posProbabilityMass](#), [TRADEMGEN::DemandStruct::_posProbDist](#), [TRADEMGEN::DemandStruct::_prefDepTimeProbDist](#), [TRADEMGEN::DemandStruct::_sevmgrServicePtr](#), [TRADEMGEN::DemandStruct::_stayProbDist](#), [TRADEMGEN::DemandStruct::_timeValueProbDist](#), [TRADEMGEN::DemandStruct::_tripProbDist](#), and [_uniformGenerator](#).

24.25.4 Member Data Documentation

24.25.4.1 SEVMGR::SEVMGR_ServicePtr_T
TRADEMGEN::DemandParserHelper::doEndDemand::_sevmgrServicePtr

Actor Specific Context.

Definition at line 282 of file [DemandParserHelper.hpp](#).

Referenced by [operator\(\)](#).

24.25.4.2 stdair::RandomGeneration&
TRADEMGEN::DemandParserHelper::doEndDemand::_uniformGenerator

Definition at line 283 of file [DemandParserHelper.hpp](#).

Referenced by [operator\(\)](#).

24.25.4.3 const POSProbabilityMass_T& TRADEMGEN::DemandParserHelper::doEndDemand::_posProbabilityMass

Definition at line 284 of file [DemandParserHelper.hpp](#).

Referenced by [operator\(\)](#).

24.25.4.4 DemandStruct& TRADEMGEN::DemandParserHelper::ParserSemanticAction::_demand [inherited]

Actor Context.

Definition at line 35 of file [DemandParserHelper.hpp](#).

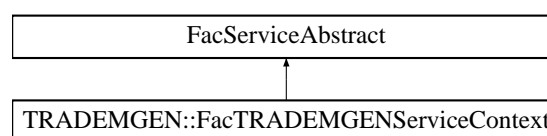
Referenced by [operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDTDProbMass::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDTD::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeTimeValueProbMass::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeTimeValue::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeWTP::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storePrefDepTimeProbMass::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storePrefDepTime::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeFFProbMass::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeFFCode::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeStayProbMass::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeStayCode::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeTripProbMass::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeTripCode::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeChannelProbMass::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeChannelCode::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storePosProbMass::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storePosCode::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDemandNonRefundableProb::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDemandChangeFeeDisutility::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDemandChangeFeeProb::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDemandStdDev::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDemandMean::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storePrefCabin::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDestination::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeOrigin::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDow::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storePrefDepDateRangeEnd::operator\(\)](#) and [TRADEMGEN::DemandParserHelper::storePrefDepDateRangeStart::operator\(\)](#).

The documentation for this struct was generated from the following files:

- [trademgen/command/DemandParserHelper.hpp](#)
- [trademgen/command/DemandParserHelper.cpp](#)

24.26 FacServiceAbstract Class Reference

Inheritance diagram for FacServiceAbstract:



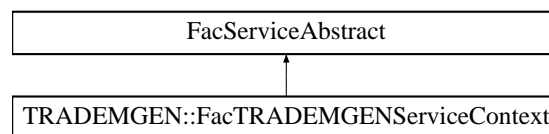
The documentation for this class was generated from the following file:

- trademgen/factory/[FacTRADEMGENSEerviceContext.hpp](#)

24.27 TRADEMGEN::FacTRADEMGENSEerviceContext Class Reference

Factory for creating the TraDemGen service context instance.

#include <trademgen/factory/FacTRADEMGENSEerviceContext.hpp>Inheritance diagram for TRADEMGEN::FacTRADEMGENSEerviceContext::



Public Member Functions

- [~FacTRADEMGENSEerviceContext\(\)](#)
- [TRADEMGEN_ServiceContext](#) & [create](#) (const stdair::RandomSeed_T &)

Static Public Member Functions

- static [FacTRADEMGENSEerviceContext](#) & [instance](#) ()

Protected Member Functions

- [FacTRADEMGENSEerviceContext\(\)](#)

24.27.1 Detailed Description

Factory for creating the TraDemGen service context instance.

Definition at line 21 of file [FacTRADEMGENSEerviceContext.hpp](#).

24.27.2 Constructor & Destructor Documentation

24.27.2.1 TRADEMGEN::FacTRADEMGENSEerviceContext::~~FacTRADEMGENSEerviceContext()

Destructor.

The Destruction put the `_instance` to NULL in order to be clean for the next [FacTRADEMGENSEerviceContext::instance\(\)](#).

Definition at line 17 of file [FacTRADEMGENSEerviceContext.cpp](#).

24.27.2.2 TRADEMGEN::FacTRADEMGENSEerviceContext::FacTRADEMGENSEerviceContext () [*inline, protected*]

Default Constructor.

This constructor is protected in order to ensure the singleton pattern.

Definition at line 57 of file [FacTRADEMGENSEerviceContext.hpp](#).

Referenced by [instance\(\)](#).

24.27.3 Member Function Documentation

24.27.3.1 FacTRADEMGENSEerviceContext & TRADEMGEN::FacTRADEMGENSEerviceContext::instance () [*static*]

Provide the unique instance.

The singleton is instantiated when first used.

Returns:

[FacTRADEMGENSEerviceContext&](#)

Definition at line 22 of file [FacTRADEMGENSEerviceContext.cpp](#).

References [FacTRADEMGENSEerviceContext\(\)](#).

24.27.3.2 TRADEMGEN_ServiceContext & TRADEMGEN::FacTRADEMGENSEerviceContext::create (const stdair::RandomSeed_T & iRandomSeed)

Create a new [TRADEMGEN_ServiceContext](#) object.

This new object is added to the list of instantiated objects.

Parameters:

const stdair::RandomSeed_T& Seed for the random generation.

Returns:

[TRADEMGEN_ServiceContext&](#) The newly created object.

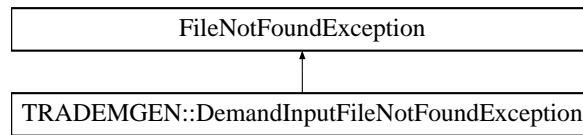
Definition at line 35 of file [FacTRADEMGENSEerviceContext.cpp](#).

The documentation for this class was generated from the following files:

- trademgen/factory/[FacTRADEMGENSEerviceContext.hpp](#)
- trademgen/factory/[FacTRADEMGENSEerviceContext.cpp](#)

24.28 FileNotFoundException Class Reference

Inheritance diagram for FileNotFoundException::



The documentation for this class was generated from the following file:

- [trademgen/TRADEMGEN_Exceptions.hpp](#)

24.29 TRADEMGEN::FlagSaver Struct Reference

Public Member Functions

- [FlagSaver](#) (std::ostream &oStream)
- [~FlagSaver](#) ()

24.29.1 Detailed Description

Helper singleton structure to store the current formatting flags of any given output stream. The flags are re-set at the structure destruction.

Definition at line 24 of file [BomDisplay.cpp](#).

24.29.2 Constructor & Destructor Documentation

24.29.2.1 TRADEMGEN::FlagSaver::FlagSaver (std::ostream & oStream) [inline]

Constructor.

Definition at line 27 of file [BomDisplay.cpp](#).

24.29.2.2 TRADEMGEN::FlagSaver::~~FlagSaver () [inline]

Destructor.

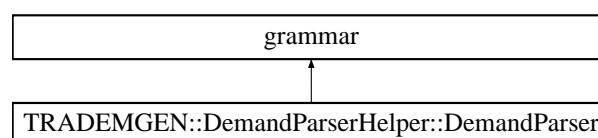
Definition at line 32 of file [BomDisplay.cpp](#).

The documentation for this struct was generated from the following file:

- [trademgen/bom/BomDisplay.cpp](#)

24.30 grammar Class Reference

Inheritance diagram for grammar::

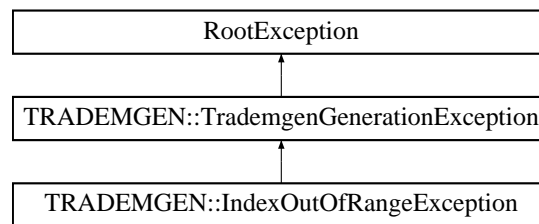


The documentation for this class was generated from the following file:

- trademgen/command/[DemandParserHelper.hpp](#)

24.31 TRADEMGEN::IndexOutOfRangeException Class Reference

#include <trademgen/TRADEMGEN_Exceptions.hpp> Inheritance diagram for TRADEMGEN::IndexOutOfRangeException:



Public Member Functions

- [IndexOutOfRangeException](#) (const std::string &iWhat)

24.31.1 Detailed Description

Exception when index out of range

Definition at line 43 of file [TRADEMGEN_Exceptions.hpp](#).

24.31.2 Constructor & Destructor Documentation

24.31.2.1 TRADEMGEN::IndexOutOfRangeException::IndexOutOfRangeException (const std::string &iWhat) [inline]

Constructor.

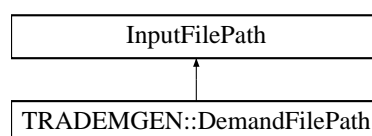
Definition at line 48 of file [TRADEMGEN_Exceptions.hpp](#).

The documentation for this class was generated from the following file:

- trademgen/[TRADEMGEN_Exceptions.hpp](#)

24.32 InputFilePath Class Reference

Inheritance diagram for InputFilePath:

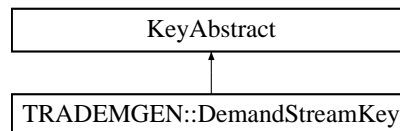


The documentation for this class was generated from the following file:

- [trademgen/TRADEMGEN_Types.hpp](#)

24.33 KeyAbstract Class Reference

Inheritance diagram for KeyAbstract::

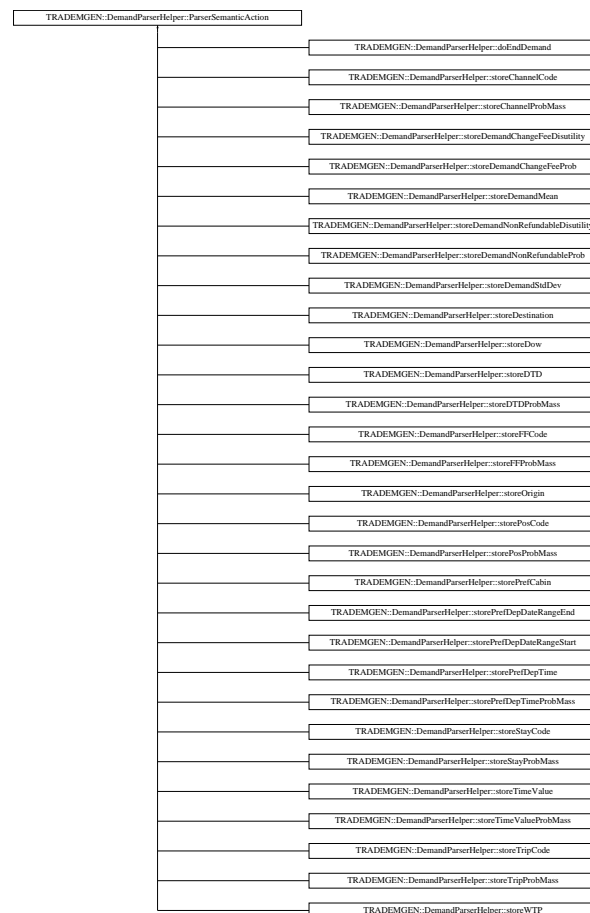


The documentation for this class was generated from the following file:

- [trademgen/bom/DemandStreamKey.hpp](#)

24.34 TRADEMGEN::DemandParserHelper::ParserSemanticAction Struct Reference

`#include <trademgen/command/DemandParserHelper.hpp>`Inheritance diagram for TRADEMGEN::DemandParserHelper::ParserSemanticAction::



Public Member Functions

- [ParserSemanticAction](#) ([DemandStruct](#) &)

Public Attributes

- [DemandStruct](#) & [_demand](#)

24.34.1 Detailed Description

Generic Semantic Action (Actor / Functor) for the Demand Parser.

Definition at line 31 of file [DemandParserHelper.hpp](#).

24.34.2 Constructor & Destructor Documentation

24.34.2.1 TRADEMGEN::DemandParserHelper::ParserSemanticAction::ParserSemanticAction (DemandStruct & *ioDemand*)

Actor Constructor.

Definition at line 26 of file [DemandParserHelper.cpp](#).

24.34.3 Member Data Documentation

24.34.3.1 DemandStruct& TRADEMGEN::DemandParserHelper::ParserSemanticAction::_-demand

Actor Context.

Definition at line 35 of file [DemandParserHelper.hpp](#).

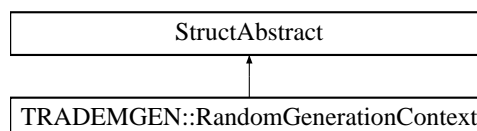
Referenced by [TRADEMGEN::DemandParserHelper::doEndDemand::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDTDProbMass::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDTD::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeTimeValueProbMass::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeTimeValue::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeWTP::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storePrefDepTimeProbMass::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storePrefDepTime::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeFFProbMass::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeFFCode::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeStayProbMass::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeStayCode::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeTripProbMass::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeTripCode::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeChannelProbMass::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeChannelCode::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storePosProbMass::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storePosCode::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDemandNonRefundableProb::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDemandChangeFeeDisutility::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDemandChangeFeeProb::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDemandStdDev::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDemandMean::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storePrefCabin::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDestination::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeOrigin::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDow::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storePrefDepDateRangeEnd::operator\(\)](#) and [TRADEMGEN::DemandParserHelper::storePrefDepDateRangeStart::operator\(\)](#).

The documentation for this struct was generated from the following files:

- [trademgen/command/DemandParserHelper.hpp](#)
- [trademgen/command/DemandParserHelper.cpp](#)

24.35 TRADEMGEN::RandomGenerationContext Struct Reference

`#include <trademgen/basic/RandomGenerationContext.hpp>` Inheritance diagram for TRADEMGEN::RandomGenerationContext:



Public Member Functions

- `const stdair::Count_T & getNumberOfRequestsGeneratedSoFar () const`

- const stdair::Probability_T & [getCumulativeProbabilitySoFar](#) () const
- void [setNumberOfRequestsGeneratedSoFar](#) (const stdair::Count_T &iCount)
- void [setCumulativeProbabilitySoFar](#) (const stdair::Probability_T &iProb)
- [RandomGenerationContext](#) ()
- [RandomGenerationContext](#) (const [RandomGenerationContext](#) &)
- [~RandomGenerationContext](#) ()
- void [incrementGeneratedRequestsCounter](#) ()
- void [reset](#) ()
- const std::string [describe](#) () const

24.35.1 Detailed Description

Structure holding the context necessary for demand random generation.

Definition at line 20 of file [RandomGenerationContext.hpp](#).

24.35.2 Constructor & Destructor Documentation

24.35.2.1 TRADEMGEN::RandomGenerationContext::RandomGenerationContext ()

Default constructor.

Definition at line 13 of file [RandomGenerationContext.cpp](#).

24.35.2.2 TRADEMGEN::RandomGenerationContext::RandomGenerationContext (const RandomGenerationContext & iRGC)

Default constructors.

Definition at line 20 of file [RandomGenerationContext.cpp](#).

24.35.2.3 TRADEMGEN::RandomGenerationContext::~~RandomGenerationContext ()

Destructor.

Definition at line 26 of file [RandomGenerationContext.cpp](#).

24.35.3 Member Function Documentation

24.35.3.1 const stdair::Count_T& TRADEMGEN::RandomGenerationContext::getNumberOfRequestsGeneratedSoFar () const [inline]

Get the number of requests generated so far.

Definition at line 26 of file [RandomGenerationContext.hpp](#).

Referenced by [TRADEMGENT::DemandStream::generateTimeOfRequestStatisticsOrder\(\)](#), [TRADEMGENT::DemandStream::getNumberOfRequestsGeneratedSoFar\(\)](#), and [TRADEMGENT::DemandStream::stillHavingRequestsToBeGenerated\(\)](#).

24.35.3.2 `const stdair::Probability_T& TRADEMGEN::RandomGenerationContext::getCumulativeProbabilitySoFar () const [inline]`

Get the cumulative probability in arrival pattern for last request generated so far (needed for sequential generation).

Definition at line 34 of file [RandomGenerationContext.hpp](#).

Referenced by [TRADEMGENT::DemandStream::generateTimeOfRequestStatisticsOrder\(\)](#).

24.35.3.3 `void TRADEMGEN::RandomGenerationContext::setNumberOfRequestsGeneratedSoFar (const stdair::Count_T & iCount) [inline]`

Set the number of requests generated so far.

Definition at line 43 of file [RandomGenerationContext.hpp](#).

Referenced by [TRADEMGENT::DemandStream::setNumberOfRequestsGeneratedSoFar\(\)](#).

24.35.3.4 `void TRADEMGENT::RandomGenerationContext::setCumulativeProbabilitySoFar (const stdair::Probability_T & iProb) [inline]`

Set the cumulative probability in arrival pattern for last request generated so far (needed for sequential generation).

Definition at line 51 of file [RandomGenerationContext.hpp](#).

Referenced by [TRADEMGENT::DemandStream::generateTimeOfRequestStatisticsOrder\(\)](#).

24.35.3.5 `void TRADEMGEN::RandomGenerationContext::incrementGeneratedRequestsCounter ()`

Increment counter of requests generated so far.

Definition at line 38 of file [RandomGenerationContext.cpp](#).

Referenced by [TRADEMGENT::DemandStream::incrementGeneratedRequestsCounter\(\)](#).

24.35.3.6 `void TRADEMGENT::RandomGenerationContext::reset ()`

Reset the counters.

Definition at line 43 of file [RandomGenerationContext.cpp](#).

Referenced by [TRADEMGENT::DemandStream::reset\(\)](#).

24.35.3.7 `const std::string TRADEMGENT::RandomGenerationContext::describe () const`

Give a description of the structure (for display purposes).

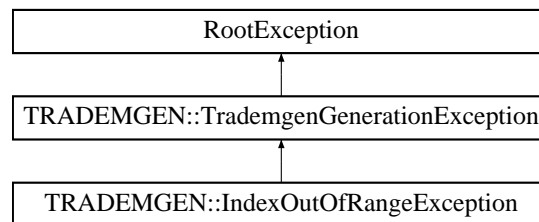
Definition at line 30 of file [RandomGenerationContext.cpp](#).

The documentation for this struct was generated from the following files:

- [trademgen/basic/RandomGenerationContext.hpp](#)
- [trademgen/basic/RandomGenerationContext.cpp](#)

24.36 RootException Class Reference

Inheritance diagram for RootException::

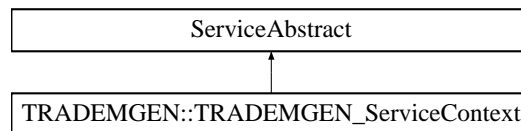


The documentation for this class was generated from the following file:

- [trademgen/TRADEMGEN_Exceptions.hpp](#)

24.37 ServiceAbstract Class Reference

Inheritance diagram for ServiceAbstract::

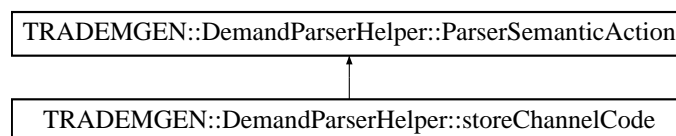


The documentation for this class was generated from the following file:

- [trademgen/service/TRADEMGEN_ServiceContext.hpp](#)

24.38 TRADEMGEN::DemandParserHelper::storeChannelCode Struct Reference

`#include <trademgen/command/DemandParserHelper.hpp>` Inheritance diagram for TRADEMGEN::DemandParserHelper::storeChannelCode::



Public Member Functions

- [storeChannelCode](#) ([DemandStruct](#) &)
- [void operator\(\)](#) ([iterator_t](#) iStr, [iterator_t](#) iStrEnd) const

Public Attributes

- [DemandStruct](#) & [_demand](#)

24.38.1 Detailed Description

Store the channel type code.

Definition at line 151 of file [DemandParserHelper.hpp](#).

24.38.2 Constructor & Destructor Documentation**24.38.2.1 TRADEMGEN::DemandParserHelper::storeChannelCode::storeChannelCode**
([DemandStruct](#) & *ioDemand*)

Actor Constructor.

Definition at line 219 of file [DemandParserHelper.cpp](#).

24.38.3 Member Function Documentation**24.38.3.1 void TRADEMGEN::DemandParserHelper::storeChannelCode::operator()** ([iterator_t](#) *iStr*, [iterator_t](#) *iStrEnd*) const

Actor Function (functor).

Definition at line 224 of file [DemandParserHelper.cpp](#).

References [TRADEMGEM::DemandParserHelper::ParserSemanticAction::_demand](#), and [TRADEMGEM::DemandStruct::_itChannelCode](#).

24.38.4 Member Data Documentation**24.38.4.1 DemandStruct& TRADEMGEN::DemandParserHelper::ParserSemanticAction::_demand** [*inherited*]

Actor Context.

Definition at line 35 of file [DemandParserHelper.hpp](#).

Referenced by [TRADEMGEM::DemandParserHelper::doEndDemand::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storeDTDProbMass::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storeDTD::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storeTimeValueProbMass::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storeTimeValue::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storeWTP::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storePrefDepTimeProbMass::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storePrefDepTime::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storeFFProbMass::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storeFFCode::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storeStayProbMass::operator\(\)](#)

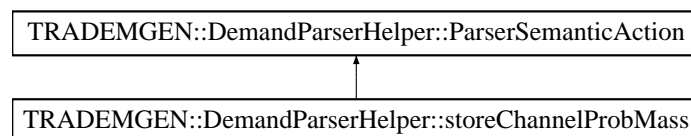
[TRADEMGEN::DemandParserHelper::storeStayCode::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeTripProbMass::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeTripCode::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeChannelProbMass::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storePosProbMass::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storePosCode::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDemandNonRefundableDisutility::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDemandNonRefundableProb::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDemandChangeFeeDisutility::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDemandChangeFeeProb::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDemandStdDev::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDemandMean::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storePrefCabin::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDestination::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeOrigin::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDow::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storePrefDepDateRangeEnd::operator\(\)](#) and [TRADEMGEN::DemandParserHelper::storePrefDepDateRangeStart::operator\(\)](#).

The documentation for this struct was generated from the following files:

- [trademgen/command/DemandParserHelper.hpp](#)
- [trademgen/command/DemandParserHelper.cpp](#)

24.39 TRADEMGEN::DemandParserHelper::storeChannelProbMass Struct Reference

`#include <trademgen/command/DemandParserHelper.hpp>` Inheritance diagram for TRADEMGEN::DemandParserHelper::storeChannelProbMass::



Public Member Functions

- [storeChannelProbMass](#) ([DemandStruct](#) &)
- void [operator\(\)](#) (double iReal) const

Public Attributes

- [DemandStruct](#) & [_demand](#)

24.39.1 Detailed Description

Store the channel type probability mass.

Definition at line 159 of file [DemandParserHelper.hpp](#).

24.39.2 Constructor & Destructor Documentation

24.39.2.1 TRADEMGEN::DemandParserHelper::storeChannelProbMass::storeChannelProbMass (DemandStruct & *ioDemand*)

Actor Constructor.

Definition at line 231 of file [DemandParserHelper.cpp](#).

24.39.3 Member Function Documentation

24.39.3.1 void TRADEMGEN::DemandParserHelper::storeChannelProbMass::operator() (double *iReal*) const

Actor Function (functor).

Definition at line 236 of file [DemandParserHelper.cpp](#).

References [TRADEMGEN::DemandStruct::_channelProbDist](#), [TRADEMGEN::DemandParserHelper::ParserSemanticAction::demand](#), and [TRADEMGEN::DemandStruct::_itChannelCode](#).

24.39.4 Member Data Documentation

24.39.4.1 DemandStruct& TRADEMGEN::DemandParserHelper::ParserSemanticAction::_demand [inherited]

Actor Context.

Definition at line 35 of file [DemandParserHelper.hpp](#).

Referenced by [TRADEMGEN::DemandParserHelper::doEndDemand::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDTDProbMass::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDTD::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeTimeValueProbMass::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeTimeValue::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeWTP::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storePrefDepTimeProbMass::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storePrefDepTime::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeFFProbMass::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeFFCode::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeStayProbMass::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeStayCode::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeTripProbMass::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeTripCode::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeChannelCode::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storePosProbMass::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storePosCode::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDemandNonRefundable::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDemandNonRefundableProb::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDemandChangeFeeDisutility::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDemandChangeFeeProb::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDemandStdDev::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDemandMean::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storePrefCabin::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDestination::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeOrigin::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDow::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storePrefDepDateRangeEnd::operator\(\)](#) and [TRADEMGEN::DemandParserHelper::storePrefDepDateRangeStart::operator\(\)](#).

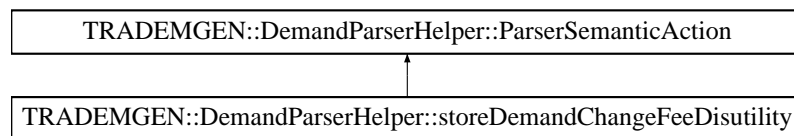
24.40 TRADEMGEN::DemandParserHelper::storeDemandChangeFeeDisutility Struct Reference 158

The documentation for this struct was generated from the following files:

- [trademgen/command/DemandParserHelper.hpp](#)
- [trademgen/command/DemandParserHelper.cpp](#)

24.40 TRADEMGEN::DemandParserHelper::storeDemandChangeFeeDisutility Struct Reference

`#include <trademgen/command/DemandParserHelper.hpp>` Inheritance diagram for TRADEMGEN::DemandParserHelper::storeDemandChangeFeeDisutility::



Public Member Functions

- [storeDemandChangeFeeDisutility](#) ([DemandStruct](#) &)
- [void operator\(\)](#) (double iReal) const

Public Attributes

- [DemandStruct](#) & [_demand](#)

24.40.1 Detailed Description

Store the demand change fee disutility.

Definition at line 111 of file [DemandParserHelper.hpp](#).

24.40.2 Constructor & Destructor Documentation

24.40.2.1 TRADEMGEN::DemandParserHelper::storeDemandChangeFeeDisutility::storeDemandChangeFeeDisutility([DemandStruct](#) & *ioDemand*)

Actor Constructor.

Definition at line 150 of file [DemandParserHelper.cpp](#).

24.40.3 Member Function Documentation

24.40.3.1 void TRADEMGEN::DemandParserHelper::storeDemandChangeFeeDisutility::operator() (double *iReal*) const

Actor Function (functor).

Definition at line 155 of file [DemandParserHelper.cpp](#).

References [TRADEMGEN::DemandStruct::_changeFeeDisutility](#), and
[TRADEMGEN::DemandParserHelper::ParserSemanticAction::_demand](#).

24.40.4 Member Data Documentation

24.40.4.1 DemandStruct& TRADEMGEN::DemandParserHelper::ParserSemanticAction::_demand [inherited]

Actor Context.

Definition at line 35 of file [DemandParserHelper.hpp](#).

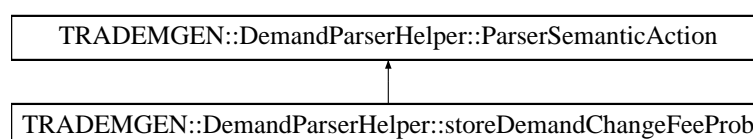
Referenced by [TRADEMGEN::DemandParserHelper::doEndDemand::operator\(\)](#),
[TRADEMGEN::DemandParserHelper::storeDTDProbMass::operator\(\)](#),
[TRADEMGEN::DemandParserHelper::storeDTD::operator\(\)](#), [TRADEM-](#)
[GEN::DemandParserHelper::storeTimeValueProbMass::operator\(\)](#), [TRADEM-](#)
[GEN::DemandParserHelper::storeTimeValue::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeWTP::operator\(\)](#),
[TRADEMGEN::DemandParserHelper::storePrefDepTimeProbMass::operator\(\)](#),
[TRADEMGEN::DemandParserHelper::storePrefDepTime::operator\(\)](#), [TRADEM-](#)
[GEN::DemandParserHelper::storeFFProbMass::operator\(\)](#), [TRADEM-](#)
[GEN::DemandParserHelper::storeFFCode::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeStayProbMass::operator\(\)](#),
[TRADEMGEN::DemandParserHelper::storeStayCode::operator\(\)](#), [TRADEM-](#)
[GEN::DemandParserHelper::storeTripProbMass::operator\(\)](#), [TRADEM-](#)
[GEN::DemandParserHelper::storeTripCode::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeChannelProbMass::operator\(\)](#),
[TRADEMGEN::DemandParserHelper::storeChannelCode::operator\(\)](#), [TRADEM-](#)
[GEN::DemandParserHelper::storePosProbMass::operator\(\)](#), [TRADEM-](#)
[GEN::DemandParserHelper::storePosCode::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDemandNonRefundableProb::operator\(\)](#),
[TRADEMGEN::DemandParserHelper::storeDemandNonRefundableProb::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDemandChangeFeeProb::operator\(\)](#),
[TRADEMGEN::DemandParserHelper::storeDemandStdDev::operator\(\)](#), [TRADEM-](#)
[GEN::DemandParserHelper::storeDemandMean::operator\(\)](#), [TRADEM-](#)
[GEN::DemandParserHelper::storePrefCabin::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDestination::operator\(\)](#),
[TRADEMGEN::DemandParserHelper::storeOrigin::operator\(\)](#), [TRADEM-](#)
[GEN::DemandParserHelper::storeDow::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storePrefDepDateRangeEnd::operator\(\)](#),
and [TRADEMGEN::DemandParserHelper::storePrefDepDateRangeStart::operator\(\)](#).

The documentation for this struct was generated from the following files:

- [trademgen/command/DemandParserHelper.hpp](#)
- [trademgen/command/DemandParserHelper.cpp](#)

24.41 TRADEMGEN::DemandParserHelper::storeDemandChangeFeeProb Struct Reference

#include <[trademgen/command/DemandParserHelper.hpp](#)> Inheritance diagram for TRADEMGEN::DemandParserHelper::storeDemandChangeFeeProb::



Public Member Functions

- [storeDemandChangeFeeProb](#) ([DemandStruct](#) &)
- void [operator\(\)](#) (double *iReal*) const

Public Attributes

- [DemandStruct](#) & [_demand](#)

24.41.1 Detailed Description

Store the demand change fee probability.

Definition at line 103 of file [DemandParserHelper.hpp](#).

24.41.2 Constructor & Destructor Documentation

24.41.2.1 TRADEMGEN::DemandParserHelper::storeDemandChangeFeeProb::storeDemandChangeFeeProb ([DemandStruct](#) & *ioDemand*)

Actor Constructor.

Definition at line 139 of file [DemandParserHelper.cpp](#).

24.41.3 Member Function Documentation

24.41.3.1 void TRADEMGEN::DemandParserHelper::storeDemandChangeFeeProb::operator() (double *iReal*) const

Actor Function (functor).

Definition at line 144 of file [DemandParserHelper.cpp](#).

References [TRADEMGEN::DemandStruct::_changeFeeProb](#), and [TRADEMGEN::DemandParserHelper::ParserSemanticAction::demand](#).

24.41.4 Member Data Documentation

24.41.4.1 [DemandStruct](#)& TRADEMGEN::DemandParserHelper::ParserSemanticAction::_demand [\[inherited\]](#)

Actor Context.

Definition at line 35 of file [DemandParserHelper.hpp](#).

Referenced by [TRADEMGEN::DemandParserHelper::doEndDemand::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDTDProbMass::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDTD::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeTimeValueProbMass::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeTimeValue::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeWTP::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storePrefDepTimeProbMass::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storePrefDepTime::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeFFProbMass::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeFFCode::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeStayProbMass::operator\(\)](#)

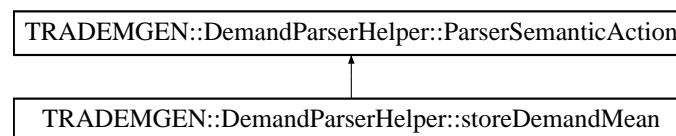
[TRADEMGEN::DemandParserHelper::storeStayCode::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeTripProbMass::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeTripCode::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeChannelProbMass::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeChannelCode::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storePosProbMass::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storePosCode::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDemandNonRefundableProb::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDemandChangeFeeDisutility::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDemandStdDev::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDemandMean::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storePrefCabin::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDestination::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeOrigin::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDow::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storePrefDepDateRangeEnd::operator\(\)](#) and [TRADEMGEN::DemandParserHelper::storePrefDepDateRangeStart::operator\(\)](#).

The documentation for this struct was generated from the following files:

- [trademgen/command/DemandParserHelper.hpp](#)
- [trademgen/command/DemandParserHelper.cpp](#)

24.42 TRADEMGEN::DemandParserHelper::storeDemandMean Struct Reference

#include <trademgen/command/DemandParserHelper.hpp> Inheritance diagram for TRADEMGEN::DemandParserHelper::storeDemandMean:



Public Member Functions

- [storeDemandMean](#) ([DemandStruct](#) &)
- void [operator\(\)](#) (double iReal) const

Public Attributes

- [DemandStruct](#) & [_demand](#)

24.42.1 Detailed Description

Store the demand mean value.

Definition at line 87 of file [DemandParserHelper.hpp](#).

24.42.2 Constructor & Destructor Documentation

24.42.2.1 TRADEMGEN::DemandParserHelper::storeDemandMean::storeDemandMean (DemandStruct & ioDemand)

Actor Constructor.

Definition at line 117 of file [DemandParserHelper.cpp](#).

24.42.3 Member Function Documentation

24.42.3.1 void TRADEMGEN::DemandParserHelper::storeDemandMean::operator() (double iReal) const

Actor Function (functor).

Definition at line 122 of file [DemandParserHelper.cpp](#).

References [TRADEMGEM::DemandParserHelper::ParserSemanticAction::_demand](#), and [TRADEMGEM::DemandStruct::_demandMean](#).

24.42.4 Member Data Documentation

24.42.4.1 DemandStruct& TRADEMGEN::DemandParserHelper::ParserSemanticAction::_demand [inherited]

Actor Context.

Definition at line 35 of file [DemandParserHelper.hpp](#).

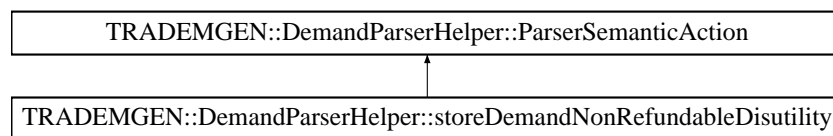
Referenced by [TRADEMGEM::DemandParserHelper::doEndDemand::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storeDTDProbMass::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storeDTD::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storeTimeValueProbMass::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storeTimeValue::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storeWTP::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storePrefDepTimeProbMass::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storePrefDepTime::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storeFFProbMass::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storeFFCode::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storeStayProbMass::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storeStayCode::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storeTripProbMass::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storeTripCode::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storeChannelProbMass::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storeChannelCode::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storePosProbMass::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storePosCode::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storeDemandNonRefundable::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storeDemandNonRefundableProb::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storeDemandChangeFeeDisutility::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storeDemandChangeFeeProb::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storeDemandStdDev::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storePrefCabin::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storeDestination::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storeOrigin::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storeDow::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storePrefDepDateRangeEnd::operator\(\)](#), and [TRADEMGEM::DemandParserHelper::storePrefDepDateRangeStart::operator\(\)](#).

The documentation for this struct was generated from the following files:

- [trademgen/command/DemandParserHelper.hpp](#)
- [trademgen/command/DemandParserHelper.cpp](#)

24.43 TRADEMGEN::DemandParserHelper::storeDemandNonRefundableDisutility Struct Reference

#include <trademgen/command/DemandParserHelper.hpp> Inheritance diagram for TRADEMGEN::DemandParserHelper::storeDemandNonRefundableDisutility::



Public Member Functions

- [storeDemandNonRefundableDisutility](#) ([DemandStruct](#) &)
- void [operator\(\)](#) (double *iReal*) const

Public Attributes

- [DemandStruct](#) & [_demand](#)

24.43.1 Detailed Description

Store the demand non refundable disutility.

Definition at line 127 of file [DemandParserHelper.hpp](#).

24.43.2 Constructor & Destructor Documentation

24.43.2.1 TRADEMGEN::DemandParserHelper::storeDemandNonRefundableDisutility::storeDemandNonRefundableDisutility([DemandStruct](#) & *ioDemand*)

Actor Constructor.

Definition at line 174 of file [DemandParserHelper.cpp](#).

24.43.3 Member Function Documentation

24.43.3.1 void TRADEMGEN::DemandParserHelper::storeDemandNonRefundableDisutility::operator() (double *iReal*) const

Actor Function (functor).

Definition at line 179 of file [DemandParserHelper.cpp](#).

24.44 TRADEMGEN::DemandParserHelper::storeDemandNonRefundableProb Struct Reference

164

References [TRADEMGEN::DemandParserHelper::ParserSemanticAction::_demand](#), and [TRADEMGEN::DemandStruct::_nonRefundableDisutility](#).

24.43.4 Member Data Documentation

24.43.4.1 DemandStruct& TRADEMGEN::DemandParserHelper::ParserSemanticAction::_demand [inherited]

Actor Context.

Definition at line 35 of file [DemandParserHelper.hpp](#).

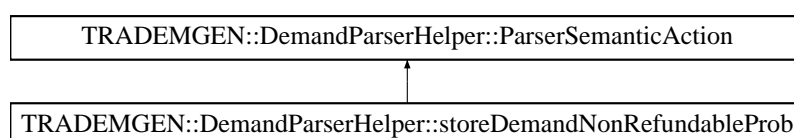
Referenced by [TRADEMGEN::DemandParserHelper::doEndDemand::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDTDProbMass::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDTD::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeTimeValueProbMass::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeTimeValue::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeWTP::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storePrefDepTimeProbMass::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storePrefDepTime::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeFFProbMass::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeFFCode::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeStayProbMass::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeStayCode::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeTripProbMass::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeTripCode::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeChannelProbMass::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeChannelCode::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storePosProbMass::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storePosCode::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDemandNonRefundableProb::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDemandChangeFeeDisutility::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDemandChangeFeeProb::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDemandStdDev::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDemandMean::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storePrefCabin::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDestination::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeOrigin::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDow::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storePrefDepDateRangeEnd::operator\(\)](#) and [TRADEMGEN::DemandParserHelper::storePrefDepDateRangeStart::operator\(\)](#).

The documentation for this struct was generated from the following files:

- [trademgen/command/DemandParserHelper.hpp](#)
- [trademgen/command/DemandParserHelper.cpp](#)

24.44 TRADEMGEN::DemandParserHelper::storeDemandNonRefundableProb Struct Reference

`#include <trademgen/command/DemandParserHelper.hpp>` Inheritance diagram for `TRADEMGEN::DemandParserHelper::storeDemandNonRefundableProb`:



Public Member Functions

- [storeDemandNonRefundableProb](#) ([DemandStruct](#) &)
- void [operator\(\)](#) (double *iReal*) const

Public Attributes

- [DemandStruct](#) & [_demand](#)

24.44.1 Detailed Description

Store the demand non refundable probability.

Definition at line 119 of file [DemandParserHelper.hpp](#).

24.44.2 Constructor & Destructor Documentation

24.44.2.1 TRADEMGEN::DemandParserHelper::storeDemandNonRefundableProb::storeDemandNonRefundableProb([DemandStruct](#) & *ioDemand*)

Actor Constructor.

Definition at line 162 of file [DemandParserHelper.cpp](#).

24.44.3 Member Function Documentation

24.44.3.1 void TRADEMGEN::DemandParserHelper::storeDemandNonRefundableProb::operator() (double *iReal*) const

Actor Function (functor).

Definition at line 167 of file [DemandParserHelper.cpp](#).

References [TRADEMGEM::DemandParserHelper::ParserSemanticAction::_demand](#), and [TRADEMGEM::DemandStruct::_nonRefundableProb](#).

24.44.4 Member Data Documentation

24.44.4.1 [DemandStruct& TRADEMGEM::DemandParserHelper::ParserSemanticAction::_demand](#) [*inherited*]

Actor Context.

Definition at line 35 of file [DemandParserHelper.hpp](#).

Referenced by [TRADEMGEM::DemandParserHelper::doEndDemand::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storeDTDProbMass::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storeDTD::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storeTimeValueProbMass::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storeTimeValue::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storeWTP::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storePrefDepTimeProbMass::operator\(\)](#),

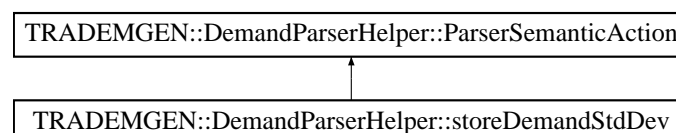
[TRADEMGEN::DemandParserHelper::storePrefDepTime::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeFFProbMass::operator\(\)](#),
[TRADEMGEN::DemandParserHelper::storeFFCode::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeStayProbMass::operator\(\)](#),
[TRADEMGEN::DemandParserHelper::storeStayCode::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeTripProbMass::operator\(\)](#),
[TRADEMGEN::DemandParserHelper::storeTripCode::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeChannelProbMass::operator\(\)](#),
[TRADEMGEN::DemandParserHelper::storeChannelCode::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storePosProbMass::operator\(\)](#),
[TRADEMGEN::DemandParserHelper::storePosCode::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDemandNonRefundable::operator\(\)](#),
[TRADEMGEN::DemandParserHelper::storeDemandChangeFeeDisutility::operator\(\)](#),
[TRADEMGEN::DemandParserHelper::storeDemandChangeFeeProb::operator\(\)](#),
[TRADEMGEN::DemandParserHelper::storeDemandStdDev::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDemandMean::operator\(\)](#),
[TRADEMGEN::DemandParserHelper::storePrefCabin::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDestination::operator\(\)](#),
[TRADEMGEN::DemandParserHelper::storeOrigin::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDow::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storePrefDepDateRangeEnd::operator\(\)](#)
and [TRADEMGEN::DemandParserHelper::storePrefDepDateRangeStart::operator\(\)](#).

The documentation for this struct was generated from the following files:

- [trademgen/command/DemandParserHelper.hpp](#)
- [trademgen/command/DemandParserHelper.cpp](#)

24.45 TRADEMGEN::DemandParserHelper::storeDemandStdDev Struct Reference

#include <[trademgen/command/DemandParserHelper.hpp](#)>Inheritance diagram for TRADEMGEN::DemandParserHelper::storeDemandStdDev::



Public Member Functions

- [storeDemandStdDev](#) ([DemandStruct](#) &)
- void [operator\(\)](#) (double iReal) const

Public Attributes

- [DemandStruct](#) & [_demand](#)

24.45.1 Detailed Description

Store the demand standard deviation value.

Definition at line 95 of file [DemandParserHelper.hpp](#).

24.45.2 Constructor & Destructor Documentation

24.45.2.1 TRADEMGEN::DemandParserHelper::storeDemandStdDev::storeDemandStdDev (DemandStruct & *ioDemand*)

Actor Constructor.

Definition at line 128 of file [DemandParserHelper.cpp](#).

24.45.3 Member Function Documentation

24.45.3.1 void TRADEMGEN::DemandParserHelper::storeDemandStdDev::operator() (double *iReal*) const

Actor Function (functor).

Definition at line 133 of file [DemandParserHelper.cpp](#).

References [TRADEMGEN::DemandParserHelper::ParserSemanticAction::_demand](#), and [TRADEMGEN::DemandStruct::_demandStdDev](#).

24.45.4 Member Data Documentation

24.45.4.1 DemandStruct& TRADEMGEN::DemandParserHelper::ParserSemanticAction::_demand [inherited]

Actor Context.

Definition at line 35 of file [DemandParserHelper.hpp](#).

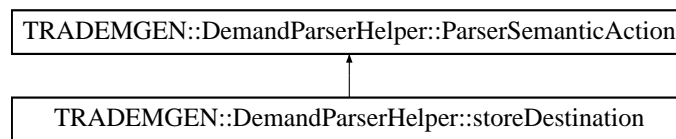
Referenced by [TRADEMGEN::DemandParserHelper::doEndDemand::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDTDProbMass::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDTD::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeTimeValueProbMass::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeTimeValue::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeWTP::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storePrefDepTimeProbMass::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storePrefDepTime::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeFFProbMass::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeFFCode::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeStayProbMass::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeStayCode::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeTripProbMass::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeTripCode::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeChannelProbMass::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeChannelCode::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storePosProbMass::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storePosCode::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDemandNonRefundable::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDemandNonRefundableProb::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDemandChangeFeeDisutility::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDemandChangeFeeProb::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDemandMean::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storePrefCabin::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDestination::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeOrigin::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDow::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storePrefDepDateRangeEnd::operator\(\)](#), and [TRADEMGEN::DemandParserHelper::storePrefDepDateRangeStart::operator\(\)](#).

The documentation for this struct was generated from the following files:

- [trademgen/command/DemandParserHelper.hpp](#)
- [trademgen/command/DemandParserHelper.cpp](#)

24.46 TRADEMGEN::DemandParserHelper::storeDestination Struct Reference

`#include <trademgen/command/DemandParserHelper.hpp>` Inheritance diagram for TRADEMGEN::DemandParserHelper::storeDestination::



Public Member Functions

- [storeDestination](#) ([DemandStruct](#) &)
- [void operator\(\)](#) ([iterator_t](#) iStr, [iterator_t](#) iStrEnd) const

Public Attributes

- [DemandStruct](#) & [_demand](#)

24.46.1 Detailed Description

Store the destination.

Definition at line 71 of file [DemandParserHelper.hpp](#).

24.46.2 Constructor & Destructor Documentation

24.46.2.1 TRADEMGEN::DemandParserHelper::storeDestination::storeDestination ([DemandStruct](#) & *ioDemand*)

Actor Constructor.

Definition at line 92 of file [DemandParserHelper.cpp](#).

24.46.3 Member Function Documentation

24.46.3.1 void TRADEMGEN::DemandParserHelper::storeDestination::operator() ([iterator_t](#) *iStr*, [iterator_t](#) *iStrEnd*) const

Actor Function (functor).

Definition at line 97 of file [DemandParserHelper.cpp](#).

References [TRADEMGEN::DemandParserHelper::ParserSemanticAction::_demand](#), and [TRADEMGEN::DemandStruct::_destination](#).

24.46.4 Member Data Documentation

24.46.4.1 DemandStruct& TRADEMGEN::DemandParserHelper::ParserSemanticAction::_-demand [inherited]

Actor Context.

Definition at line 35 of file [DemandParserHelper.hpp](#).

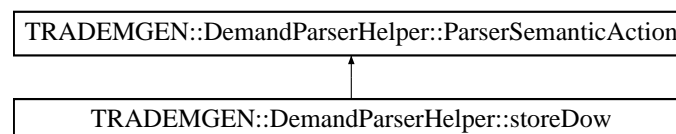
Referenced by [TRADEMGEN::DemandParserHelper::doEndDemand::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDTDProbMass::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDTD::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeTimeValueProbMass::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeTimeValue::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeWTP::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storePrefDepTimeProbMass::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storePrefDepTime::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeFFProbMass::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeFFCode::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeStayProbMass::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeStayCode::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeTripProbMass::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeTripCode::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeChannelProbMass::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeChannelCode::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storePosProbMass::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storePosCode::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDemandNonRefundableProb::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDemandChangeFeeDisutility::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDemandChangeFeeProb::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDemandStdDev::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDemandMean::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storePrefCabin::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeOrigin::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDow::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storePrefDepDateRangeEnd::operator\(\)](#), and [TRADEMGEN::DemandParserHelper::storePrefDepDateRangeStart::operator\(\)](#).

The documentation for this struct was generated from the following files:

- [trademgen/command/DemandParserHelper.hpp](#)
- [trademgen/command/DemandParserHelper.cpp](#)

24.47 TRADEMGEN::DemandParserHelper::storeDow Struct Reference

`#include <trademgen/command/DemandParserHelper.hpp>` Inheritance diagram for TRADEMGEN::DemandParserHelper::storeDow:



Public Member Functions

- [storeDow](#) ([DemandStruct](#) &)

- void [operator\(\)](#) ([iterator_t](#) iStr, [iterator_t](#) iStrEnd) const

Public Attributes

- [DemandStruct](#) & [_demand](#)

24.47.1 Detailed Description

Store the DOW (day of the Week).

Definition at line 55 of file [DemandParserHelper.hpp](#).

24.47.2 Constructor & Destructor Documentation

24.47.2.1 TRADEMGEN::DemandParserHelper::storeDow::storeDow (DemandStruct & ioDemand)

Actor Constructor.

Definition at line 70 of file [DemandParserHelper.cpp](#).

24.47.3 Member Function Documentation

24.47.3.1 void TRADEMGEN::DemandParserHelper::storeDow::operator() (iterator_t iStr, iterator_t iStrEnd) const

Actor Function (functor).

Definition at line 75 of file [DemandParserHelper.cpp](#).

References [TRADEMGEM::DemandParserHelper::ParserSemanticAction::_demand](#), and [TRADEMGEM::DemandStruct::_dow](#).

24.47.4 Member Data Documentation

24.47.4.1 DemandStruct& TRADEMGEN::DemandParserHelper::ParserSemanticAction::_demand [inherited]

Actor Context.

Definition at line 35 of file [DemandParserHelper.hpp](#).

Referenced by [TRADEMGEM::DemandParserHelper::doEndDemand::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storeDTDProbMass::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storeDTD::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storeTimeValueProbMass::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storeTimeValue::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storeWTP::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storePrefDepTimeProbMass::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storePrefDepTime::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storeFFProbMass::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storeFFCode::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storeStayProbMass::operator\(\)](#)

```

TRADEMGEN::DemandParserHelper::storeStayCode::operator(),
GEN::DemandParserHelper::storeTripProbMass::operator(),
TRADEMGEN::DemandParserHelper::storeTripCode::operator(), TRADEMGEN::DemandParserHelper::storeChannelProbMass::operator(),
TRADEMGEN::DemandParserHelper::storeChannelCode::operator(),
GEN::DemandParserHelper::storePosProbMass::operator(),
GEN::DemandParserHelper::storePosCode::operator(), TRADEMGEN::DemandParserHelper::storeDemandNonRefundableProb::operator(),
TRADEMGEN::DemandParserHelper::storeDemandNonRefundableProb::operator(),
TRADEMGEN::DemandParserHelper::storeDemandChangeFeeDisutility::operator(),
TRADEMGEN::DemandParserHelper::storeDemandChangeFeeProb::operator(),
TRADEMGEN::DemandParserHelper::storeDemandStdDev::operator(),
GEN::DemandParserHelper::storeDemandMean::operator(),
GEN::DemandParserHelper::storePrefCabin::operator(), TRADEMGEN::DemandParserHelper::storeDestination::operator(),
TRADEMGEN::DemandParserHelper::storeOrigin::operator(), operator(),
GEN::DemandParserHelper::storePrefDepDateRangeEnd::operator(), and
GEN::DemandParserHelper::storePrefDepDateRangeStart::operator().

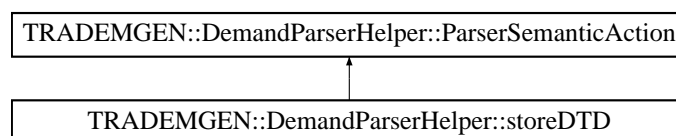
```

The documentation for this struct was generated from the following files:

- trademgen/command/[DemandParserHelper.hpp](#)
- trademgen/command/[DemandParserHelper.cpp](#)

24.48 TRADEMGEN::DemandParserHelper::storeDTD Struct Reference

`#include <trademgen/command/DemandParserHelper.hpp>` Inheritance diagram for TRADEMGEN::DemandParserHelper::storeDTD::



Public Member Functions

- [storeDTD](#) ([DemandStruct](#) &)
- void [operator\(\)](#) (unsigned int iInteger) const

Public Attributes

- [DemandStruct](#) & [_demand](#)

24.48.1 Detailed Description

Store the parameters for the arrival pattern (as expressed in DTD) continuous probability distribution.

Definition at line 258 of file [DemandParserHelper.hpp](#).

24.48.2 Constructor & Destructor Documentation

24.48.2.1 TRADEMGEN::DemandParserHelper::storeDTD::storeDTD (DemandStruct & ioDemand)

Actor Constructor.

Definition at line 428 of file [DemandParserHelper.cpp](#).

24.48.3 Member Function Documentation

24.48.3.1 void TRADEMGEN::DemandParserHelper::storeDTD::operator() (unsigned int iInteger) const

Actor Function (functor).

Definition at line 433 of file [DemandParserHelper.cpp](#).

References [TRADEMGEM::DemandParserHelper::ParserSemanticAction::_demand](#), and [TRADEMGEM::DemandStruct::_itDTD](#).

24.48.4 Member Data Documentation

24.48.4.1 DemandStruct& TRADEMGEN::DemandParserHelper::ParserSemanticAction::_demand [inherited]

Actor Context.

Definition at line 35 of file [DemandParserHelper.hpp](#).

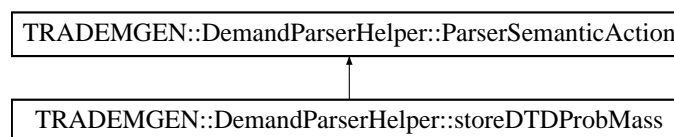
Referenced by [TRADEMGEM::DemandParserHelper::doEndDemand::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storeDTDProbMass::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storeTimeValueProbMass::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storeTimeValue::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storeWTP::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storePrefDepTimeProbMass::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storePrefDepTime::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storeFFProbMass::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storeFFCode::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storeStayProbMass::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storeStayCode::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storeTripProbMass::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storeTripCode::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storeChannelProbMass::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storeChannelCode::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storePosProbMass::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storePosCode::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storeDemandNonRefundable::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storeDemandNonRefundableProb::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storeDemandChangeFeeDisutility::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storeDemandChangeFeeProb::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storeDemandStdDev::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storeDemandMean::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storePrefCabin::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storeDestination::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storeOrigin::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storeDow::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storePrefDepDateRangeEnd::operator\(\)](#) and [TRADEMGEM::DemandParserHelper::storePrefDepDateRangeStart::operator\(\)](#).

The documentation for this struct was generated from the following files:

- [trademgen/command/DemandParserHelper.hpp](#)
- [trademgen/command/DemandParserHelper.cpp](#)

24.49 TRADEMGEN::DemandParserHelper::storeDTDProbMass Struct Reference

`#include <trademgen/command/DemandParserHelper.hpp>` Inheritance diagram for TRADEMGEN::DemandParserHelper::storeDTDProbMass::



Public Member Functions

- [storeDTDProbMass](#) ([DemandStruct](#) &)
- [void operator\(\)](#) (double *iReal*) const

Public Attributes

- [DemandStruct](#) & [_demand](#)

24.49.1 Detailed Description

Store the parameters for the arrival pattern (as expressed in DTD) continuous probability distribution.

Definition at line 267 of file [DemandParserHelper.hpp](#).

24.49.2 Constructor & Destructor Documentation

24.49.2.1 TRADEMGEN::DemandParserHelper::storeDTDProbMass::storeDTDProbMass ([DemandStruct](#) & *ioDemand*)

Actor Constructor.

Definition at line 440 of file [DemandParserHelper.cpp](#).

24.49.3 Member Function Documentation

24.49.3.1 void TRADEMGEN::DemandParserHelper::storeDTDProbMass::operator() (double *iReal*) const

Actor Function (functor).

Definition at line 445 of file [DemandParserHelper.cpp](#).

References [TRADEMGEN::DemandParserHelper::ParserSemanticAction::_demand](#), [TRADEMGEN::DemandStruct::_dtdProbDist](#), and [TRADEMGEN::DemandStruct::_itDTD](#).

24.49.4 Member Data Documentation

24.49.4.1 DemandStruct& TRADEMGEN::DemandParserHelper::ParserSemanticAction::_-demand [inherited]

Actor Context.

Definition at line 35 of file [DemandParserHelper.hpp](#).

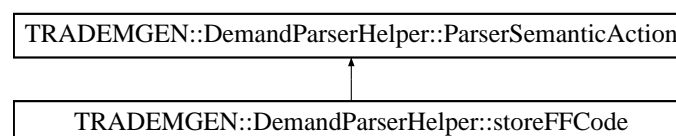
Referenced by [TRADEMGEN::DemandParserHelper::doEndDemand::operator\(\)\(\)](#), [operator\(\)\(\)](#), [TRADEMGEN::DemandParserHelper::storeDTD::operator\(\)\(\)](#), [TRADEMGEN::DemandParserHelper::storeTimeValueProbMass::operator\(\)\(\)](#), [TRADEMGEN::DemandParserHelper::storeTimeValue::operator\(\)\(\)](#), [TRADEMGEN::DemandParserHelper::storeWTP::operator\(\)\(\)](#), [TRADEMGEN::DemandParserHelper::storePrefDepTimeProbMass::operator\(\)\(\)](#), [TRADEMGEN::DemandParserHelper::storePrefDepTime::operator\(\)\(\)](#), [TRADEMGEN::DemandParserHelper::storeFFProbMass::operator\(\)\(\)](#), [TRADEMGEN::DemandParserHelper::storeFFCode::operator\(\)\(\)](#), [TRADEMGEN::DemandParserHelper::storeStayProbMass::operator\(\)\(\)](#), [TRADEMGEN::DemandParserHelper::storeStayCode::operator\(\)\(\)](#), [TRADEMGEN::DemandParserHelper::storeTripProbMass::operator\(\)\(\)](#), [TRADEMGEN::DemandParserHelper::storeTripCode::operator\(\)\(\)](#), [TRADEMGEN::DemandParserHelper::storeChannelProbMass::operator\(\)\(\)](#), [TRADEMGEN::DemandParserHelper::storeChannelCode::operator\(\)\(\)](#), [TRADEMGEN::DemandParserHelper::storePosProbMass::operator\(\)\(\)](#), [TRADEMGEN::DemandParserHelper::storePosCode::operator\(\)\(\)](#), [TRADEMGEN::DemandParserHelper::storeDemandNonRefundableProb::operator\(\)\(\)](#), [TRADEMGEN::DemandParserHelper::storeDemandChangeFeeDisutility::operator\(\)\(\)](#), [TRADEMGEN::DemandParserHelper::storeDemandChangeFeeProb::operator\(\)\(\)](#), [TRADEMGEN::DemandParserHelper::storeDemandStdDev::operator\(\)\(\)](#), [TRADEMGEN::DemandParserHelper::storeDemandMean::operator\(\)\(\)](#), [TRADEMGEN::DemandParserHelper::storePrefCabin::operator\(\)\(\)](#), [TRADEMGEN::DemandParserHelper::storeDestination::operator\(\)\(\)](#), [TRADEMGEN::DemandParserHelper::storeOrigin::operator\(\)\(\)](#), [TRADEMGEN::DemandParserHelper::storeDow::operator\(\)\(\)](#), [TRADEMGEN::DemandParserHelper::storePrefDepDateRangeEnd::operator\(\)\(\)](#) and [TRADEMGEN::DemandParserHelper::storePrefDepDateRangeStart::operator\(\)\(\)](#).

The documentation for this struct was generated from the following files:

- [trademgen/command/DemandParserHelper.hpp](#)
- [trademgen/command/DemandParserHelper.cpp](#)

24.50 TRADEMGEN::DemandParserHelper::storeFFCode Struct Reference

`#include <trademgen/command/DemandParserHelper.hpp>` Inheritance diagram for `TRADEMGEN::DemandParserHelper::storeFFCode::`



Public Member Functions

- [storeFFCode](#) ([DemandStruct](#) &)
- [void operator\(\)](#) ([iterator_t](#) iStr, [iterator_t](#) iStrEnd) const

Public Attributes

- [DemandStruct](#) & `_demand`

24.50.1 Detailed Description

Store the frequent flyer code.

Definition at line 199 of file [DemandParserHelper.hpp](#).

24.50.2 Constructor & Destructor Documentation**24.50.2.1 TRADEMGEN::DemandParserHelper::storeFFCode::storeFFCode (DemandStruct & ioDemand)**

Actor Constructor.

Definition at line 323 of file [DemandParserHelper.cpp](#).

24.50.3 Member Function Documentation**24.50.3.1 void TRADEMGEN::DemandParserHelper::storeFFCode::operator() (iterator_t iStr, iterator_t iStrEnd) const**

Actor Function (functor).

Definition at line 328 of file [DemandParserHelper.cpp](#).

References [TRADEMGEM::DemandParserHelper::ParserSemanticAction::_demand](#), and [TRADEMGEM::DemandStruct::_itFFCode](#).

24.50.4 Member Data Documentation**24.50.4.1 DemandStruct& TRADEMGEN::DemandParserHelper::ParserSemanticAction::_demand [inherited]**

Actor Context.

Definition at line 35 of file [DemandParserHelper.hpp](#).

Referenced by [TRADEMGEM::DemandParserHelper::doEndDemand::operator\(\)\(\)](#), [TRADEMGEM::DemandParserHelper::storeDTDProbMass::operator\(\)\(\)](#), [TRADEMGEM::DemandParserHelper::storeDTD::operator\(\)\(\)](#), [TRADEMGEM::DemandParserHelper::storeTimeValueProbMass::operator\(\)\(\)](#), [TRADEMGEM::DemandParserHelper::storeTimeValue::operator\(\)\(\)](#), [TRADEMGEM::DemandParserHelper::storeWTP::operator\(\)\(\)](#), [TRADEMGEM::DemandParserHelper::storePrefDepTimeProbMass::operator\(\)\(\)](#), [TRADEMGEM::DemandParserHelper::storePrefDepTime::operator\(\)\(\)](#), [TRADEMGEM::DemandParserHelper::storeFFProbMass::operator\(\)\(\)](#), [TRADEMGEM::DemandParserHelper::storeStayProbMass::operator\(\)\(\)](#), [TRADEMGEM::DemandParserHelper::storeStayCode::operator\(\)\(\)](#), [TRADEMGEM::DemandParserHelper::storeTripProbMass::operator\(\)\(\)](#), [TRADEMGEM::DemandParserHelper::storeTripCode::operator\(\)\(\)](#), [TRADEMGEM::DemandParserHelper::storeChannelProbMass::operator\(\)\(\)](#), [TRADEMGEM::DemandParserHelper::storeChannelCode::operator\(\)\(\)](#), and [TRADEMGEM::DemandParserHelper::storePosProbMass::operator\(\)\(\)](#).

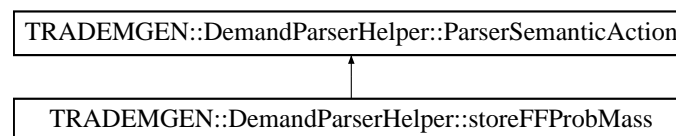
[GEN::DemandParserHelper::storePosCode::operator\(\)\(\)](#), [TRADEMG](#)[EN::DemandParserHelper::storeDemandNonRefundable](#)
[TRADEMG](#)[EN::DemandParserHelper::storeDemandNonRefundableProb::operator\(\)\(\)](#),
[TRADEMG](#)[EN::DemandParserHelper::storeDemandChangeFeeDisutility::operator\(\)\(\)](#),
[TRADEMG](#)[EN::DemandParserHelper::storeDemandChangeFeeProb::operator\(\)\(\)](#),
[TRADEMG](#)[EN::DemandParserHelper::storeDemandStdDev::operator\(\)\(\)](#), [TRADEMG](#)[EN::DemandParserHelper::storeDemandMean::operator\(\)\(\)](#), [TRADEMG](#)[EN::DemandParserHelper::storePrefCabin::operator\(\)\(\)](#), [TRADEMG](#)[EN::DemandParserHelper::storeDestination::operator\(\)\(\)](#),
[TRADEMG](#)[EN::DemandParserHelper::storeOrigin::operator\(\)\(\)](#), [TRADEMG](#)[EN::DemandParserHelper::storeDow::operator\(\)\(\)](#), [TRADEMG](#)[EN::DemandParserHelper::storePrefDepDateRangeEnd::operator\(\)\(\)](#)
 and [TRADEMG](#)[EN::DemandParserHelper::storePrefDepDateRangeStart::operator\(\)\(\)](#).

The documentation for this struct was generated from the following files:

- [trademgen/command/DemandParserHelper.hpp](#)
- [trademgen/command/DemandParserHelper.cpp](#)

24.51 TRADEMGEN::DemandParserHelper::storeFFProbMass Struct Reference

`#include <trademgen/command/DemandParserHelper.hpp>` Inheritance diagram for TRADEMGEN::DemandParserHelper::storeFFProbMass::



Public Member Functions

- [storeFFProbMass](#) ([DemandStruct](#) &)
- void [operator\(\)](#) (double iReal) const

Public Attributes

- [DemandStruct](#) & [_demand](#)

24.51.1 Detailed Description

Store the frequent flyer probability mass.

Definition at line 207 of file [DemandParserHelper.hpp](#).

24.51.2 Constructor & Destructor Documentation

24.51.2.1 TRADEMGEN::DemandParserHelper::storeFFProbMass::storeFFProbMass (DemandStruct & ioDemand)

Actor Constructor.

Definition at line 334 of file [DemandParserHelper.cpp](#).

24.51.3 Member Function Documentation

24.51.3.1 void TRADEMGEN::DemandParserHelper::storeFFProbMass::operator() (double *iReal*) const

Actor Function (functor).

Definition at line 339 of file [DemandParserHelper.cpp](#).

References [TRADEMGEN::DemandParserHelper::ParserSemanticAction::_demand](#), [TRADEMGEN::DemandStruct::_ffProbDist](#), and [TRADEMGEN::DemandStruct::_itFFCode](#).

24.51.4 Member Data Documentation

24.51.4.1 DemandStruct& TRADEMGEN::DemandParserHelper::ParserSemanticAction::_demand [inherited]

Actor Context.

Definition at line 35 of file [DemandParserHelper.hpp](#).

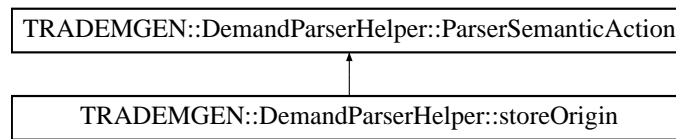
Referenced by [TRADEMGEN::DemandParserHelper::doEndDemand::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDTDProbMass::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDTD::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeTimeValueProbMass::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeTimeValue::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeWTP::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storePrefDepTimeProbMass::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storePrefDepTime::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeFFCode::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeStayProbMass::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeStayCode::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeTripProbMass::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeTripCode::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeChannelProbMass::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeChannelCode::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storePosProbMass::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storePosCode::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDemandNonRefundable::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDemandNonRefundableProb::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDemandChangeFeeDisutility::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDemandChangeFeeProb::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDemandStdDev::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDemandMean::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storePrefCabin::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDestination::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeOrigin::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDow::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storePrefDepDateRangeEnd::operator\(\)](#) and [TRADEMGEN::DemandParserHelper::storePrefDepDateRangeStart::operator\(\)](#).

The documentation for this struct was generated from the following files:

- [trademgen/command/DemandParserHelper.hpp](#)
- [trademgen/command/DemandParserHelper.cpp](#)

24.52 TRADEMGEN::DemandParserHelper::storeOrigin Struct Reference

`#include <trademgen/command/DemandParserHelper.hpp>` Inheritance diagram for `TRADEMGEN::DemandParserHelper::storeOrigin::`



Public Member Functions

- [storeOrigin](#) ([DemandStruct](#) &)
- void [operator\(\)](#) ([iterator_t](#) iStr, [iterator_t](#) iStrEnd) const

Public Attributes

- [DemandStruct](#) & [_demand](#)

24.52.1 Detailed Description

Store the origin.

Definition at line 63 of file [DemandParserHelper.hpp](#).

24.52.2 Constructor & Destructor Documentation

24.52.2.1 TRADEMGEN::DemandParserHelper::storeOrigin::storeOrigin ([DemandStruct](#) & *ioDemand*)

Actor Constructor.

Definition at line 81 of file [DemandParserHelper.cpp](#).

24.52.3 Member Function Documentation

24.52.3.1 void TRADEMGEN::DemandParserHelper::storeOrigin::operator() ([iterator_t](#) iStr, [iterator_t](#) iStrEnd) const

Actor Function (functor).

Definition at line 86 of file [DemandParserHelper.cpp](#).

References [TRADEMGEM::DemandParserHelper::ParserSemanticAction::_demand](#), and [TRADEMGEM::DemandStruct::_origin](#).

24.52.4 Member Data Documentation

24.52.4.1 [DemandStruct](#)& TRADEMGEN::DemandParserHelper::ParserSemanticAction::_demand [inherited]

Actor Context.

Definition at line 35 of file [DemandParserHelper.hpp](#).

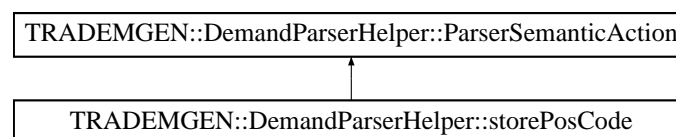
Referenced by TRADEMGEN::DemandParserHelper::doEndDemand::operator(),
 TRADEMGEN::DemandParserHelper::storeDTDProbMass::operator(),
 TRADEMGEN::DemandParserHelper::storeDTD::operator(), TRADEM-
 GEN::DemandParserHelper::storeTimeValueProbMass::operator(), TRADEM-
 GEN::DemandParserHelper::storeTimeValue::operator(), TRADEMGEN::DemandParserHelper::storeWTP::operator(),
 TRADEMGEN::DemandParserHelper::storePrefDepTimeProbMass::operator(),
 TRADEMGEN::DemandParserHelper::storePrefDepTime::operator(), TRADEM-
 GEN::DemandParserHelper::storeFFProbMass::operator(), TRADEM-
 GEN::DemandParserHelper::storeFFCode::operator(), TRADEMGEN::DemandParserHelper::storeStayProbMass::operator(),
 TRADEMGEN::DemandParserHelper::storeStayCode::operator(), TRADEM-
 GEN::DemandParserHelper::storeTripProbMass::operator(), TRADEM-
 GEN::DemandParserHelper::storeTripCode::operator(), TRADEMGEN::DemandParserHelper::storeChannelProbMass::oper
 TRADEMGEN::DemandParserHelper::storeChannelCode::operator(), TRADEM-
 GEN::DemandParserHelper::storePosProbMass::operator(), TRADEM-
 GEN::DemandParserHelper::storePosCode::operator(), TRADEMGEN::DemandParserHelper::storeDemandNonRefundable
 TRADEMGEN::DemandParserHelper::storeDemandNonRefundableProb::operator(),
 TRADEMGEN::DemandParserHelper::storeDemandChangeFeeDisutility::operator(),
 TRADEMGEN::DemandParserHelper::storeDemandChangeFeeProb::operator(),
 TRADEMGEN::DemandParserHelper::storeDemandStdDev::operator(), TRADEM-
 GEN::DemandParserHelper::storeDemandMean::operator(), TRADEM-
 GEN::DemandParserHelper::storePrefCabin::operator(), TRADEMGEN::DemandParserHelper::storeDestination::operator()
 operator(), TRADEMGEN::DemandParserHelper::storeDow::operator(), TRADEM-
 GEN::DemandParserHelper::storePrefDepDateRangeEnd::operator(), and TRADEM-
 GEN::DemandParserHelper::storePrefDepDateRangeStart::operator().

The documentation for this struct was generated from the following files:

- trademgen/command/DemandParserHelper.hpp
- trademgen/command/DemandParserHelper.cpp

24.53 TRADEMGEN::DemandParserHelper::storePosCode Struct Reference

#include <trademgen/command/DemandParserHelper.hpp> Inheritance diagram for TRADEMGEN::DemandParserHelper::storePosCode::



Public Member Functions

- storePosCode (DemandStruct &)
- void operator() (iterator_t iStr, iterator_t iStrEnd) const

Public Attributes

- DemandStruct & _demand

24.53.1 Detailed Description

Store the pos type code.

Definition at line 135 of file [DemandParserHelper.hpp](#).

24.53.2 Constructor & Destructor Documentation

24.53.2.1 TRADEMGEN::DemandParserHelper::storePosCode::storePosCode (DemandStruct & ioDemand)

Actor Constructor.

Definition at line 185 of file [DemandParserHelper.cpp](#).

24.53.3 Member Function Documentation

24.53.3.1 void TRADEMGEN::DemandParserHelper::storePosCode::operator() (iterator_t iStr, iterator_t iStrEnd) const

Actor Function (functor).

Definition at line 190 of file [DemandParserHelper.cpp](#).

References [TRADEMGEM::DemandParserHelper::ParserSemanticAction::_demand](#), and [TRADEMGEM::DemandStruct::_itPosCode](#).

24.53.4 Member Data Documentation

24.53.4.1 DemandStruct& TRADEMGEN::DemandParserHelper::ParserSemanticAction::_demand [inherited]

Actor Context.

Definition at line 35 of file [DemandParserHelper.hpp](#).

Referenced by [TRADEMGEM::DemandParserHelper::doEndDemand::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storeDTDProbMass::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storeDTD::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storeTimeValueProbMass::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storeTimeValue::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storeWTP::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storePrefDepTimeProbMass::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storePrefDepTime::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storeFFProbMass::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storeFFCode::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storeStayProbMass::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storeStayCode::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storeTripProbMass::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storeTripCode::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storeChannelProbMass::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storeChannelCode::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storePosProbMass::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storeDemandNonRefundableDisutility::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storeDemandNonRefundableProb::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storeDemandChangeFeeDisutility::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storeDemandChangeFeeProb::operator\(\)](#),

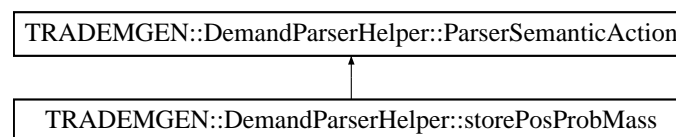
TRADEMGEN::DemandParserHelper::storeDemandStdDev::operator(), TRADEM-
 GEN::DemandParserHelper::storeDemandMean::operator(), TRADEM-
 GEN::DemandParserHelper::storePrefCabin::operator(), TRADEMGEN::DemandParserHelper::storeDestination::operator()
 TRADEMGEN::DemandParserHelper::storeOrigin::operator(), TRADEM-
 GEN::DemandParserHelper::storeDow::operator(), TRADEMGEN::DemandParserHelper::storePrefDepDateRangeEnd::ope
 and TRADEMGEN::DemandParserHelper::storePrefDepDateRangeStart::operator()).

The documentation for this struct was generated from the following files:

- trademgen/command/DemandParserHelper.hpp
- trademgen/command/DemandParserHelper.cpp

24.54 TRADEMGEN::DemandParserHelper::storePosProbMass Struct Reference

#include <trademgen/command/DemandParserHelper.hpp> Inheritance diagram for TRADEMGEN::DemandParserHelper::storePosProbMass:



Public Member Functions

- storePosProbMass (DemandStruct &)
- void operator() (double iReal) const

Public Attributes

- DemandStruct & _demand

24.54.1 Detailed Description

Store the pos type probability mass.

Definition at line 143 of file DemandParserHelper.hpp.

24.54.2 Constructor & Destructor Documentation

24.54.2.1 TRADEMGEN::DemandParserHelper::storePosProbMass::storePosProbMass (DemandStruct & ioDemand)

Actor Constructor.

Definition at line 197 of file DemandParserHelper.cpp.

24.54.3 Member Function Documentation

24.54.3.1 void TRADEMGEN::DemandParserHelper::storePosProbMass::operator() (double *iReal*) const

Actor Function (functor).

Definition at line 202 of file [DemandParserHelper.cpp](#).

References [TRADEMGEN::DemandParserHelper::ParserSemanticAction::_demand](#), [TRADEMGEN::DemandStruct::_itPosCode](#), and [TRADEMGEN::DemandStruct::_posProbDist](#).

24.54.4 Member Data Documentation

24.54.4.1 DemandStruct& TRADEMGEN::DemandParserHelper::ParserSemanticAction::_demand [inherited]

Actor Context.

Definition at line 35 of file [DemandParserHelper.hpp](#).

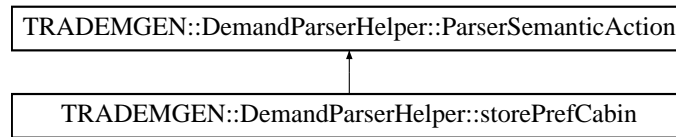
Referenced by [TRADEMGEN::DemandParserHelper::doEndDemand::operator\(\)\(\)](#), [TRADEMGEN::DemandParserHelper::storeDTDProbMass::operator\(\)\(\)](#), [TRADEMGEN::DemandParserHelper::storeDTD::operator\(\)\(\)](#), [TRADEMGEN::DemandParserHelper::storeTimeValueProbMass::operator\(\)\(\)](#), [TRADEMGEN::DemandParserHelper::storeTimeValue::operator\(\)\(\)](#), [TRADEMGEN::DemandParserHelper::storeWTP::operator\(\)\(\)](#), [TRADEMGEN::DemandParserHelper::storePrefDepTimeProbMass::operator\(\)\(\)](#), [TRADEMGEN::DemandParserHelper::storePrefDepTime::operator\(\)\(\)](#), [TRADEMGEN::DemandParserHelper::storeFFProbMass::operator\(\)\(\)](#), [TRADEMGEN::DemandParserHelper::storeFFCode::operator\(\)\(\)](#), [TRADEMGEN::DemandParserHelper::storeStayProbMass::operator\(\)\(\)](#), [TRADEMGEN::DemandParserHelper::storeStayCode::operator\(\)\(\)](#), [TRADEMGEN::DemandParserHelper::storeTripProbMass::operator\(\)\(\)](#), [TRADEMGEN::DemandParserHelper::storeTripCode::operator\(\)\(\)](#), [TRADEMGEN::DemandParserHelper::storeChannelProbMass::operator\(\)\(\)](#), [TRADEMGEN::DemandParserHelper::storeChannelCode::operator\(\)\(\)](#), [TRADEMGEN::DemandParserHelper::storePosCode::operator\(\)\(\)](#), [TRADEMGEN::DemandParserHelper::storeDemandNonRefundableDisutility::operator\(\)\(\)](#), [TRADEMGEN::DemandParserHelper::storeDemandNonRefundableProb::operator\(\)\(\)](#), [TRADEMGEN::DemandParserHelper::storeDemandChangeFeeDisutility::operator\(\)\(\)](#), [TRADEMGEN::DemandParserHelper::storeDemandChangeFeeProb::operator\(\)\(\)](#), [TRADEMGEN::DemandParserHelper::storeDemandStdDev::operator\(\)\(\)](#), [TRADEMGEN::DemandParserHelper::storeDemandMean::operator\(\)\(\)](#), [TRADEMGEN::DemandParserHelper::storePrefCabin::operator\(\)\(\)](#), [TRADEMGEN::DemandParserHelper::storeDestination::operator\(\)\(\)](#), [TRADEMGEN::DemandParserHelper::storeOrigin::operator\(\)\(\)](#), [TRADEMGEN::DemandParserHelper::storeDow::operator\(\)\(\)](#), [TRADEMGEN::DemandParserHelper::storePrefDepDateRangeEnd::operator\(\)\(\)](#) and [TRADEMGEN::DemandParserHelper::storePrefDepDateRangeStart::operator\(\)\(\)](#).

The documentation for this struct was generated from the following files:

- [trademgen/command/DemandParserHelper.hpp](#)
- [trademgen/command/DemandParserHelper.cpp](#)

24.55 TRADEMGEN::DemandParserHelper::storePrefCabin Struct Reference

`#include <trademgen/command/DemandParserHelper.hpp>` Inheritance diagram for `TRADEMGEN::DemandParserHelper::storePrefCabin::`



Public Member Functions

- [storePrefCabin](#) ([DemandStruct](#) &)
- void [operator\(\)](#) ([iterator_t](#) iStr, [iterator_t](#) iStrEnd) const

Public Attributes

- [DemandStruct](#) & [_demand](#)

24.55.1 Detailed Description

Store the preferred cabin.

Definition at line 79 of file [DemandParserHelper.hpp](#).

24.55.2 Constructor & Destructor Documentation

24.55.2.1 TRADEMGEM::DemandParserHelper::storePrefCabin::storePrefCabin ([DemandStruct](#) & *ioDemand*)

Actor Constructor.

Definition at line 104 of file [DemandParserHelper.cpp](#).

24.55.3 Member Function Documentation

24.55.3.1 void TRADEMGEM::DemandParserHelper::storePrefCabin::operator() ([iterator_t](#) iStr, [iterator_t](#) iStrEnd) const

Actor Function (functor).

Definition at line 109 of file [DemandParserHelper.cpp](#).

References [TRADEMGEM::DemandParserHelper::ParserSemanticAction::_demand](#), and [TRADEMGEM::DemandStruct::_prefCabin](#).

24.55.4 Member Data Documentation

24.55.4.1 [DemandStruct](#)& TRADEMGEM::DemandParserHelper::ParserSemanticAction::_demand [inherited]

Actor Context.

Definition at line 35 of file [DemandParserHelper.hpp](#).

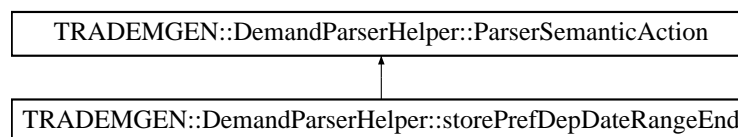
Referenced by TRADEMGEN::DemandParserHelper::doEndDemand::operator(),
 TRADEMGEN::DemandParserHelper::storeDTDProbMass::operator(),
 TRADEMGEN::DemandParserHelper::storeDTD::operator(), TRADEM-
 GEN::DemandParserHelper::storeTimeValueProbMass::operator(), TRADEM-
 GEN::DemandParserHelper::storeTimeValue::operator(), TRADEMGEN::DemandParserHelper::storeWTP::operator(),
 TRADEMGEN::DemandParserHelper::storePrefDepTimeProbMass::operator(),
 TRADEMGEN::DemandParserHelper::storePrefDepTime::operator(), TRADEM-
 GEN::DemandParserHelper::storeFFProbMass::operator(), TRADEM-
 GEN::DemandParserHelper::storeFFCode::operator(), TRADEMGEN::DemandParserHelper::storeStayProbMass::operator(),
 TRADEMGEN::DemandParserHelper::storeStayCode::operator(), TRADEM-
 GEN::DemandParserHelper::storeTripProbMass::operator(), TRADEM-
 GEN::DemandParserHelper::storeTripCode::operator(), TRADEMGEN::DemandParserHelper::storeChannelProbMass::oper
 TRADEMGEN::DemandParserHelper::storeChannelCode::operator(), TRADEM-
 GEN::DemandParserHelper::storePosProbMass::operator(), TRADEM-
 GEN::DemandParserHelper::storePosCode::operator(), TRADEMGEN::DemandParserHelper::storeDemandNonRefundable
 TRADEMGEN::DemandParserHelper::storeDemandNonRefundableProb::operator(),
 TRADEMGEN::DemandParserHelper::storeDemandChangeFeeDisutility::operator(),
 TRADEMGEN::DemandParserHelper::storeDemandChangeFeeProb::operator(),
 TRADEMGEN::DemandParserHelper::storeDemandStdDev::operator(),
 TRADEMGEN::DemandParserHelper::storeDemandMean::operator(), oper-
 ator(), TRADEMGEN::DemandParserHelper::storeDestination::operator(),
 TRADEMGEN::DemandParserHelper::storeOrigin::operator(), TRADEM-
 GEN::DemandParserHelper::storeDow::operator(), TRADEMGEN::DemandParserHelper::storePrefDepDateRangeEnd::oper
 and TRADEMGEN::DemandParserHelper::storePrefDepDateRangeStart::operator().

The documentation for this struct was generated from the following files:

- trademgen/command/DemandParserHelper.hpp
- trademgen/command/DemandParserHelper.cpp

24.56 TRADEMGEN::DemandParserHelper::storePrefDepDateRangeEnd Struct Reference

#include <trademgen/command/DemandParserHelper.hpp> Inheritance diagram for TRADEMGEN::DemandParserHelper::storePrefDepDateRangeEnd::



Public Member Functions

- storePrefDepDateRangeEnd (DemandStruct &)
- void operator() (iterator_t iStr, iterator_t iStrEnd) const

Public Attributes

- DemandStruct & _demand

24.56.1 Detailed Description

Store the end of the date range.

Definition at line 47 of file [DemandParserHelper.hpp](#).

24.56.2 Constructor & Destructor Documentation

24.56.2.1 TRADEMGEN::DemandParserHelper::storePrefDepDateRangeEnd::storePrefDepDateRangeEnd (DemandStruct & ioDemand)

Actor Constructor.

Definition at line 47 of file [DemandParserHelper.cpp](#).

24.56.3 Member Function Documentation

24.56.3.1 void TRADEMGEN::DemandParserHelper::storePrefDepDateRangeEnd::operator() (iterator_t iStr, iterator_t iStrEnd) const

Actor Function (functor).

Definition at line 52 of file [DemandParserHelper.cpp](#).

References [TRADEMGEM::DemandStruct::_dateRange](#), [TRADEMGEM::DemandParserHelper::ParserSemanticAction::_demand](#), [TRADEMGEM::DemandStruct::_itSeconds](#), [TRADEMGEM::DemandStruct::_prefDepDateEnd](#), [TRADEMGEM::DemandStruct::_prefDepDateStart](#), and [TRADEMGEM::DemandStruct::getDate\(\)](#).

24.56.4 Member Data Documentation

24.56.4.1 DemandStruct& TRADEMGEN::DemandParserHelper::ParserSemanticAction::_demand [inherited]

Actor Context.

Definition at line 35 of file [DemandParserHelper.hpp](#).

Referenced by [TRADEMGEM::DemandParserHelper::doEndDemand::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storeDTDProbMass::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storeDTD::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storeTimeValueProbMass::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storeTimeValue::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storeWTP::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storePrefDepTimeProbMass::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storePrefDepTime::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storeFFProbMass::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storeFFCode::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storeStayProbMass::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storeStayCode::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storeTripProbMass::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storeTripCode::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storeChannelProbMass::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storeChannelCode::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storePosProbMass::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storePosCode::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storeDemandNonRefundable::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storeDemandNonRefundableProb::operator\(\)](#), and [TRADEMGEM::DemandParserHelper::storeDemandChangeFeeDisutility::operator\(\)](#).

```

TRADEMGEN::DemandParserHelper::storeDemandChangeFeeProb::operator(),
TRADEMGEN::DemandParserHelper::storeDemandStdDev::operator(),
TRADEMGEN::DemandParserHelper::storeDemandMean::operator(),
TRADEMGEN::DemandParserHelper::storePrefCabin::operator(), TRADEMGEN::DemandParserHelper::storeDestination::operator(),
TRADEMGEN::DemandParserHelper::storeOrigin::operator(),
TRADEMGEN::DemandParserHelper::storeDow::operator(), operator(), and
TRADEMGEN::DemandParserHelper::storePrefDepDateRangeStart::operator().

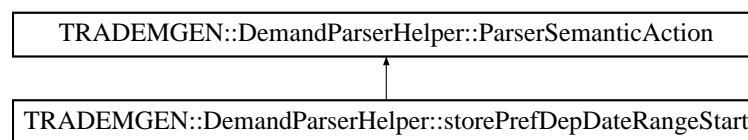
```

The documentation for this struct was generated from the following files:

- [trademgen/command/DemandParserHelper.hpp](#)
- [trademgen/command/DemandParserHelper.cpp](#)

24.57 TRADEMGEN::DemandParserHelper::storePrefDepDateRangeStart Struct Reference

`#include <trademgen/command/DemandParserHelper.hpp>` Inheritance diagram for TRADEMGEN::DemandParserHelper::storePrefDepDateRangeStart::



Public Member Functions

- [storePrefDepDateRangeStart](#) ([DemandStruct](#) &)
- [operator\(\)](#) ([iterator_t](#) iStr, [iterator_t](#) iStrEnd) const

Public Attributes

- [DemandStruct](#) & [_demand](#)

24.57.1 Detailed Description

Store the start of the date range.

Definition at line 39 of file [DemandParserHelper.hpp](#).

24.57.2 Constructor & Destructor Documentation

24.57.2.1 TRADEMGEN::DemandParserHelper::storePrefDepDateRangeStart::storePrefDepDateRangeStart ([DemandStruct](#) & *ioDemand*)

Actor Constructor.

Definition at line 32 of file [DemandParserHelper.cpp](#).

24.57.3 Member Function Documentation

24.57.3.1 void TRADEMGEN::DemandParserHelper::storePrefDepDateRangeStart::operator() (iterator_t iStr, iterator_t iStrEnd) const

Actor Function (functor).

Definition at line 37 of file [DemandParserHelper.cpp](#).

References [TRADEMGEN::DemandParserHelper::ParserSemanticAction::_demand](#), [TRADEMGEN::DemandStruct::_itSeconds](#), [TRADEMGEN::DemandStruct::_prefDepDateStart](#), and [TRADEMGEN::DemandStruct::getDate\(\)](#).

24.57.4 Member Data Documentation

24.57.4.1 DemandStruct& TRADEMGEN::DemandParserHelper::ParserSemanticAction::_demand [inherited]

Actor Context.

Definition at line 35 of file [DemandParserHelper.hpp](#).

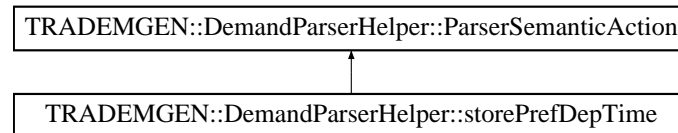
Referenced by [TRADEMGEN::DemandParserHelper::doEndDemand::operator\(\)\(\)](#), [TRADEMGEN::DemandParserHelper::storeDTDProbMass::operator\(\)\(\)](#), [TRADEMGEN::DemandParserHelper::storeDTD::operator\(\)\(\)](#), [TRADEMGEN::DemandParserHelper::storeTimeValueProbMass::operator\(\)\(\)](#), [TRADEMGEN::DemandParserHelper::storeTimeValue::operator\(\)\(\)](#), [TRADEMGEN::DemandParserHelper::storeWTP::operator\(\)\(\)](#), [TRADEMGEN::DemandParserHelper::storePrefDepTimeProbMass::operator\(\)\(\)](#), [TRADEMGEN::DemandParserHelper::storePrefDepTime::operator\(\)\(\)](#), [TRADEMGEN::DemandParserHelper::storeFFProbMass::operator\(\)\(\)](#), [TRADEMGEN::DemandParserHelper::storeFFCode::operator\(\)\(\)](#), [TRADEMGEN::DemandParserHelper::storeStayProbMass::operator\(\)\(\)](#), [TRADEMGEN::DemandParserHelper::storeStayCode::operator\(\)\(\)](#), [TRADEMGEN::DemandParserHelper::storeTripProbMass::operator\(\)\(\)](#), [TRADEMGEN::DemandParserHelper::storeTripCode::operator\(\)\(\)](#), [TRADEMGEN::DemandParserHelper::storeChannelProbMass::operator\(\)\(\)](#), [TRADEMGEN::DemandParserHelper::storeChannelCode::operator\(\)\(\)](#), [TRADEMGEN::DemandParserHelper::storePosProbMass::operator\(\)\(\)](#), [TRADEMGEN::DemandParserHelper::storePosCode::operator\(\)\(\)](#), [TRADEMGEN::DemandParserHelper::storeDemandNonRefundable::operator\(\)\(\)](#), [TRADEMGEN::DemandParserHelper::storeDemandNonRefundableProb::operator\(\)\(\)](#), [TRADEMGEN::DemandParserHelper::storeDemandChangeFeeDisutility::operator\(\)\(\)](#), [TRADEMGEN::DemandParserHelper::storeDemandChangeFeeProb::operator\(\)\(\)](#), [TRADEMGEN::DemandParserHelper::storeDemandStdDev::operator\(\)\(\)](#), [TRADEMGEN::DemandParserHelper::storeDemandMean::operator\(\)\(\)](#), [TRADEMGEN::DemandParserHelper::storePrefCabin::operator\(\)\(\)](#), [TRADEMGEN::DemandParserHelper::storeDestination::operator\(\)\(\)](#), [TRADEMGEN::DemandParserHelper::storeOrigin::operator\(\)\(\)](#), [TRADEMGEN::DemandParserHelper::storeDow::operator\(\)\(\)](#), [TRADEMGEN::DemandParserHelper::storePrefDepDateRangeEnd::operator\(\)\(\)](#), and [operator\(\)\(\)](#).

The documentation for this struct was generated from the following files:

- [trademgen/command/DemandParserHelper.hpp](#)
- [trademgen/command/DemandParserHelper.cpp](#)

24.58 TRADEMGEN::DemandParserHelper::storePrefDepTime Struct Reference

#include <trademgen/command/DemandParserHelper.hpp> Inheritance diagram for TRADEMGEN::DemandParserHelper::storePrefDepTime::



Public Member Functions

- [storePrefDepTime](#) ([DemandStruct](#) &)
- [void operator\(\)](#) ([iterator_t](#) iStr, [iterator_t](#) iStrEnd) const

Public Attributes

- [DemandStruct](#) & [_demand](#)

24.58.1 Detailed Description

Store the parameters for the preferred departure time continuous probability distribution.

Definition at line 216 of file [DemandParserHelper.hpp](#).

24.58.2 Constructor & Destructor Documentation

24.58.2.1 TRADEMGEN::DemandParserHelper::storePrefDepTime::storePrefDepTime ([DemandStruct](#) & *ioDemand*)

Actor Constructor.

Definition at line 357 of file [DemandParserHelper.cpp](#).

24.58.3 Member Function Documentation

24.58.3.1 void TRADEMGEN::DemandParserHelper::storePrefDepTime::operator() ([iterator_t](#) *iStr*, [iterator_t](#) *iStrEnd*) const

Actor Function (functor).

Definition at line 362 of file [DemandParserHelper.cpp](#).

References [TRADEMG...DemandParserHelper::ParserSemanticAction::_demand](#), [TRADEMG...DemandStruct::_itMinutes](#), [TRADEMG...DemandStruct::_itPrefDepTime](#), [TRADEMG...DemandStruct::_itSeconds](#), and [TRADEMG...DemandStruct::getTime\(\)](#).

24.58.4 Member Data Documentation

24.58.4.1 DemandStruct& TRADEMGEN::DemandParserHelper::ParserSemanticAction::_-demand [inherited]

Actor Context.

Definition at line 35 of file [DemandParserHelper.hpp](#).

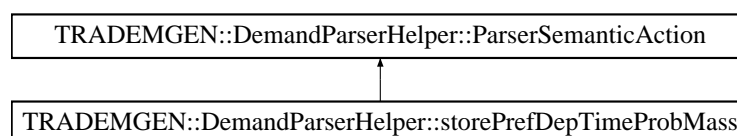
Referenced by [TRADEMGEN::DemandParserHelper::doEndDemand::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDTDProbMass::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDTD::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeTimeValueProbMass::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeTimeValue::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeWTP::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storePrefDepTimeProbMass::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeFFProbMass::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeFFCode::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeStayProbMass::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeStayCode::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeTripProbMass::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeTripCode::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeChannelProbMass::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeChannelCode::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storePosProbMass::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storePosCode::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDemandNonRefundableProb::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDemandChangeFeeDisutility::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDemandChangeFeeProb::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDemandStdDev::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDemandMean::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storePrefCabin::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDestination::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeOrigin::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDow::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storePrefDepDateRangeEnd::operator\(\)](#) and [TRADEMGEN::DemandParserHelper::storePrefDepDateRangeStart::operator\(\)](#).

The documentation for this struct was generated from the following files:

- [trademgen/command/DemandParserHelper.hpp](#)
- [trademgen/command/DemandParserHelper.cpp](#)

24.59 TRADEMGEN::DemandParserHelper::storePrefDepTimeProbMass Struct Reference

`#include <trademgen/command/DemandParserHelper.hpp>` Inheritance diagram for TRADEMGEN::DemandParserHelper::storePrefDepTimeProbMass::



Public Member Functions

- [storePrefDepTimeProbMass](#) ([DemandStruct](#) &)
- void [operator\(\)](#) (double *iReal*) const

Public Attributes

- [DemandStruct](#) & [_demand](#)

24.59.1 Detailed Description

Store the parameters for the preferred departure time continuous probability distribution.

Definition at line 225 of file [DemandParserHelper.hpp](#).

24.59.2 Constructor & Destructor Documentation**24.59.2.1 TRADEMGEN::DemandParserHelper::storePrefDepTimeProbMass::storePrefDepTimeProbMass (DemandStruct & *ioDemand*)**

Actor Constructor.

Definition at line 377 of file [DemandParserHelper.cpp](#).

24.59.3 Member Function Documentation**24.59.3.1 void TRADEMGEN::DemandParserHelper::storePrefDepTimeProbMass::operator() (double *iReal*) const**

Actor Function (functor).

Definition at line 382 of file [DemandParserHelper.cpp](#).

References [TRADEMGENT::DemandParserHelper::ParserSemanticAction::_demand](#), [TRADEMGENT::DemandStruct::_itPrefDepTime](#), and [TRADEMGENT::DemandStruct::_prefDepTimeProbDist](#).

24.59.4 Member Data Documentation**24.59.4.1 DemandStruct& TRADEMGEN::DemandParserHelper::ParserSemanticAction::_demand [inherited]**

Actor Context.

Definition at line 35 of file [DemandParserHelper.hpp](#).

Referenced by [TRADEMGENT::DemandParserHelper::doEndDemand::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeDTDProbMass::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeDTD::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeTimeValueProbMass::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeTimeValue::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeWTP::operator\(\)](#), [operator\(\)](#), and [TRADEMGENT::DemandParserHelper::storePrefDepTime::operator\(\)](#).

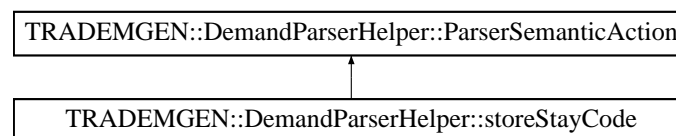
TRADEMGEN::DemandParserHelper::storeFFProbMass::operator(), TRADEM-
 GEN::DemandParserHelper::storeFFCode::operator(), TRADEMGEN::DemandParserHelper::storeStayProbMass::operator()
 TRADEMGEN::DemandParserHelper::storeStayCode::operator(), TRADEM-
 GEN::DemandParserHelper::storeTripProbMass::operator(), TRADEM-
 GEN::DemandParserHelper::storeTripCode::operator(), TRADEMGEN::DemandParserHelper::storeChannelProbMass::oper
 TRADEMGEN::DemandParserHelper::storeChannelCode::operator(), TRADEM-
 GEN::DemandParserHelper::storePosProbMass::operator(), TRADEM-
 GEN::DemandParserHelper::storePosCode::operator(), TRADEMGEN::DemandParserHelper::storeDemandNonRefundable
 TRADEMGEN::DemandParserHelper::storeDemandNonRefundableProb::operator(),
 TRADEMGEN::DemandParserHelper::storeDemandChangeFeeDisutility::operator(),
 TRADEMGEN::DemandParserHelper::storeDemandChangeFeeProb::operator(),
 TRADEMGEN::DemandParserHelper::storeDemandStdDev::operator(), TRADEM-
 GEN::DemandParserHelper::storeDemandMean::operator(), TRADEM-
 GEN::DemandParserHelper::storePrefCabin::operator(), TRADEMGEN::DemandParserHelper::storeDestination::operator()
 TRADEMGEN::DemandParserHelper::storeOrigin::operator(), TRADEM-
 GEN::DemandParserHelper::storeDow::operator(), TRADEMGEN::DemandParserHelper::storePrefDepDateRangeEnd::oper
 and TRADEMGEN::DemandParserHelper::storePrefDepDateRangeStart::operator().

The documentation for this struct was generated from the following files:

- trademgen/command/[DemandParserHelper.hpp](#)
- trademgen/command/[DemandParserHelper.cpp](#)

24.60 TRADEMGEN::DemandParserHelper::storeStayCode Struct Reference

#include <trademgen/command/DemandParserHelper.hpp> Inheritance diagram for TRADEMGEN::DemandParserHelper::storeStayCode::



Public Member Functions

- [storeStayCode](#) ([DemandStruct](#) &)
- void [operator\(\)](#) (unsigned int iInteger) const

Public Attributes

- [DemandStruct](#) & [_demand](#)

24.60.1 Detailed Description

Store the stay type code.

Definition at line 183 of file [DemandParserHelper.hpp](#).

24.60.2 Constructor & Destructor Documentation

24.60.2.1 TRADEMGEN::DemandParserHelper::storeStayCode::storeStayCode (DemandStruct & ioDemand)

Actor Constructor.

Definition at line 289 of file [DemandParserHelper.cpp](#).

24.60.3 Member Function Documentation

24.60.3.1 void TRADEMGEN::DemandParserHelper::storeStayCode::operator() (unsigned int iInteger) const

Actor Function (functor).

Definition at line 294 of file [DemandParserHelper.cpp](#).

References [TRADEMGEM::DemandParserHelper::ParserSemanticAction::_demand](#), and [TRADEMGEM::DemandStruct::_itStayDuration](#).

24.60.4 Member Data Documentation

24.60.4.1 DemandStruct& TRADEMGEN::DemandParserHelper::ParserSemanticAction::_demand [inherited]

Actor Context.

Definition at line 35 of file [DemandParserHelper.hpp](#).

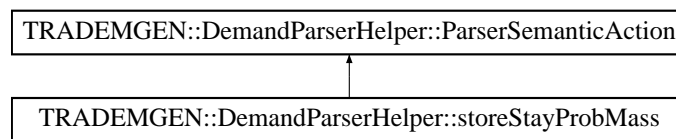
Referenced by [TRADEMGEM::DemandParserHelper::doEndDemand::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storeDTDProbMass::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storeDTD::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storeTimeValueProbMass::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storeTimeValue::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storeWTP::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storePrefDepTimeProbMass::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storePrefDepTime::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storeFFProbMass::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storeFFCode::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storeStayProbMass::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storeTripProbMass::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storeTripCode::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storeChannelProbMass::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storeChannelCode::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storePosProbMass::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storePosCode::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storeDemandNonRefundable::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storeDemandNonRefundableProb::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storeDemandChangeFeeDisutility::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storeDemandChangeFeeProb::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storeDemandStdDev::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storeDemandMean::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storePrefCabin::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storeDestination::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storeOrigin::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storeDow::operator\(\)](#), [TRADEMGEM::DemandParserHelper::storePrefDepDateRangeEnd::operator\(\)](#) and [TRADEMGEM::DemandParserHelper::storePrefDepDateRangeStart::operator\(\)](#).

The documentation for this struct was generated from the following files:

- [trademgen/command/DemandParserHelper.hpp](#)
- [trademgen/command/DemandParserHelper.cpp](#)

24.61 TRADEMGEN::DemandParserHelper::storeStayProbMass Struct Reference

#include <trademgen/command/DemandParserHelper.hpp> Inheritance diagram for TRADEMGEN::DemandParserHelper::storeStayProbMass::



Public Member Functions

- [storeStayProbMass](#) ([DemandStruct](#) &)
- void [operator\(\)](#) (double *iReal*) const

Public Attributes

- [DemandStruct](#) & [_demand](#)

24.61.1 Detailed Description

Store the stay type probability mass.

Definition at line 191 of file [DemandParserHelper.hpp](#).

24.61.2 Constructor & Destructor Documentation

24.61.2.1 TRADEMGEN::DemandParserHelper::storeStayProbMass::storeStayProbMass ([DemandStruct](#) & *ioDemand*)

Actor Constructor.

Definition at line 301 of file [DemandParserHelper.cpp](#).

24.61.3 Member Function Documentation

24.61.3.1 void TRADEMGEN::DemandParserHelper::storeStayProbMass::operator() (double *iReal*) const

Actor Function (functor).

Definition at line 306 of file [DemandParserHelper.cpp](#).

References [TRADEMGEN::DemandParserHelper::ParserSemanticAction::_demand](#), [TRADEMGEN::DemandStruct::_itStayDuration](#), and [TRADEMGEN::DemandStruct::_stayProbDist](#).

24.61.4 Member Data Documentation

24.61.4.1 DemandStruct& TRADEMGEN::DemandParserHelper::ParserSemanticAction::_demand [inherited]

Actor Context.

Definition at line 35 of file [DemandParserHelper.hpp](#).

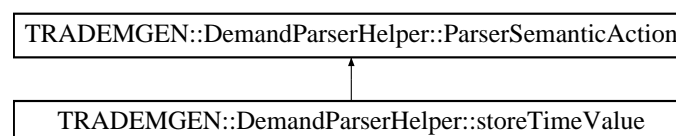
Referenced by [TRADEMGEN::DemandParserHelper::doEndDemand::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDTDProbMass::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDTD::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeTimeValueProbMass::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeTimeValue::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeWTP::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storePrefDepTimeProbMass::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storePrefDepTime::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeFFProbMass::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeFFCode::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeStayCode::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeTripProbMass::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeTripCode::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeChannelProbMass::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeChannelCode::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storePosProbMass::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storePosCode::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDemandNonRefundable::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDemandNonRefundableProb::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDemandChangeFeeDisutility::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDemandChangeFeeProb::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDemandStdDev::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDemandMean::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storePrefCabin::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDestination::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeOrigin::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDow::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storePrefDepDateRangeEnd::operator\(\)](#), and [TRADEMGEN::DemandParserHelper::storePrefDepDateRangeStart::operator\(\)](#).

The documentation for this struct was generated from the following files:

- [trademgen/command/DemandParserHelper.hpp](#)
- [trademgen/command/DemandParserHelper.cpp](#)

24.62 TRADEMGEN::DemandParserHelper::storeTimeValue Struct Reference

`#include <trademgen/command/DemandParserHelper.hpp>` Inheritance diagram for TRADEMGEN::DemandParserHelper::storeTimeValue:



Public Member Functions

- [storeTimeValue](#) ([DemandStruct](#) &)
- void [operator\(\)](#) (double *iReal*) const

Public Attributes

- [DemandStruct](#) & [_demand](#)

24.62.1 Detailed Description

Store the time value.

Definition at line 241 of file [DemandParserHelper.hpp](#).

24.62.2 Constructor & Destructor Documentation**24.62.2.1 TRADEMGEN::DemandParserHelper::storeTimeValue::storeTimeValue (DemandStruct & *ioDemand*)**

Actor Constructor.

Definition at line 404 of file [DemandParserHelper.cpp](#).

24.62.3 Member Function Documentation**24.62.3.1 void TRADEMGEN::DemandParserHelper::storeTimeValue::operator() (double *iReal*) const**

Actor Function (functor).

Definition at line 409 of file [DemandParserHelper.cpp](#).

References [TRADEMGENT::DemandParserHelper::ParserSemanticAction::_demand](#), and [TRADEMGENT::DemandStruct::_itTimeValue](#).

24.62.4 Member Data Documentation**24.62.4.1 DemandStruct& TRADEMGEN::DemandParserHelper::ParserSemanticAction::_demand [inherited]**

Actor Context.

Definition at line 35 of file [DemandParserHelper.hpp](#).

Referenced by [TRADEMGENT::DemandParserHelper::doEndDemand::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeDTDProbMass::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeDTD::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeTimeValueProbMass::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeWTP::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storePrefDepTimeProbMass::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storePrefDepTime::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeFFProbMass::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeFFCode::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeStayProbMass::operator\(\)](#)

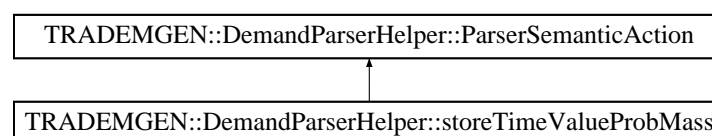
[TRADEMGEN::DemandParserHelper::storeStayCode::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeTripProbMass::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeTripCode::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeChannelProbMass::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeChannelCode::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storePosProbMass::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storePosCode::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDemandNonRefundableProb::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDemandChangeFeeDisutility::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDemandChangeFeeProb::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDemandStdDev::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDemandMean::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storePrefCabin::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDestination::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeOrigin::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDow::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storePrefDepDateRangeEnd::operator\(\)](#) and [TRADEMGEN::DemandParserHelper::storePrefDepDateRangeStart::operator\(\)](#).

The documentation for this struct was generated from the following files:

- [trademgen/command/DemandParserHelper.hpp](#)
- [trademgen/command/DemandParserHelper.cpp](#)

24.63 TRADEMGEN::DemandParserHelper::storeTimeValueProbMass Struct Reference

#include <[trademgen/command/DemandParserHelper.hpp](#)> Inheritance diagram for TRADEMGEN::DemandParserHelper::storeTimeValueProbMass::



Public Member Functions

- [storeTimeValueProbMass](#) ([DemandStruct](#) &)
- void [operator\(\)](#) (double iReal) const

Public Attributes

- [DemandStruct](#) & [_demand](#)

24.63.1 Detailed Description

Store the time value probability mass.

Definition at line 249 of file [DemandParserHelper.hpp](#).

24.63.2 Constructor & Destructor Documentation

24.63.2.1 TRADEMGEN::DemandParserHelper::storeTimeValueProbMass::storeTimeValueProbMass (DemandStruct & *ioDemand*)

Actor Constructor.

Definition at line 415 of file [DemandParserHelper.cpp](#).

24.63.3 Member Function Documentation

24.63.3.1 void TRADEMGEN::DemandParserHelper::storeTimeValueProbMass::operator() (double *iReal*) const

Actor Function (functor).

Definition at line 420 of file [DemandParserHelper.cpp](#).

References [TRADEMGEN::DemandParserHelper::ParserSemanticAction::_demand](#), [TRADEMGEN::DemandStruct::_itTimeValue](#), and [TRADEMGEN::DemandStruct::_timeValueProbDist](#).

24.63.4 Member Data Documentation

24.63.4.1 DemandStruct& TRADEMGEN::DemandParserHelper::ParserSemanticAction::_demand [inherited]

Actor Context.

Definition at line 35 of file [DemandParserHelper.hpp](#).

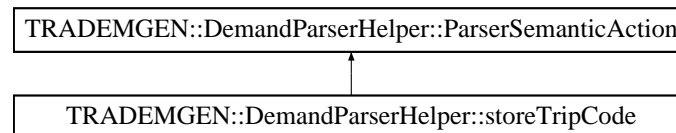
Referenced by [TRADEMGEN::DemandParserHelper::doEndDemand::operator\(\)\(\)](#), [TRADEMGEN::DemandParserHelper::storeDTDProbMass::operator\(\)\(\)](#), [TRADEMGEN::DemandParserHelper::storeDTD::operator\(\)\(\)](#), [TRADEMGEN::DemandParserHelper::storeTimeValue::operator\(\)\(\)](#), [TRADEMGEN::DemandParserHelper::storeWTP::operator\(\)\(\)](#), [TRADEMGEN::DemandParserHelper::storePrefDepTimeProbMass::operator\(\)\(\)](#), [TRADEMGEN::DemandParserHelper::storePrefDepTime::operator\(\)\(\)](#), [TRADEMGEN::DemandParserHelper::storeFFProbMass::operator\(\)\(\)](#), [TRADEMGEN::DemandParserHelper::storeFFCode::operator\(\)\(\)](#), [TRADEMGEN::DemandParserHelper::storeStayProbMass::operator\(\)\(\)](#), [TRADEMGEN::DemandParserHelper::storeStayCode::operator\(\)\(\)](#), [TRADEMGEN::DemandParserHelper::storeTripProbMass::operator\(\)\(\)](#), [TRADEMGEN::DemandParserHelper::storeTripCode::operator\(\)\(\)](#), [TRADEMGEN::DemandParserHelper::storeChannelProbMass::operator\(\)\(\)](#), [TRADEMGEN::DemandParserHelper::storeChannelCode::operator\(\)\(\)](#), [TRADEMGEN::DemandParserHelper::storePosProbMass::operator\(\)\(\)](#), [TRADEMGEN::DemandParserHelper::storePosCode::operator\(\)\(\)](#), [TRADEMGEN::DemandParserHelper::storeDemandNonRefundable::operator\(\)\(\)](#), [TRADEMGEN::DemandParserHelper::storeDemandNonRefundableProb::operator\(\)\(\)](#), [TRADEMGEN::DemandParserHelper::storeDemandChangeFeeDisutility::operator\(\)\(\)](#), [TRADEMGEN::DemandParserHelper::storeDemandChangeFeeProb::operator\(\)\(\)](#), [TRADEMGEN::DemandParserHelper::storeDemandStdDev::operator\(\)\(\)](#), [TRADEMGEN::DemandParserHelper::storeDemandMean::operator\(\)\(\)](#), [TRADEMGEN::DemandParserHelper::storePrefCabin::operator\(\)\(\)](#), [TRADEMGEN::DemandParserHelper::storeDestination::operator\(\)\(\)](#), [TRADEMGEN::DemandParserHelper::storeOrigin::operator\(\)\(\)](#), [TRADEMGEN::DemandParserHelper::storeDow::operator\(\)\(\)](#), [TRADEMGEN::DemandParserHelper::storePrefDepDateRangeEnd::operator\(\)\(\)](#), and [TRADEMGEN::DemandParserHelper::storePrefDepDateRangeStart::operator\(\)\(\)](#).

The documentation for this struct was generated from the following files:

- [trademgen/command/DemandParserHelper.hpp](#)
- [trademgen/command/DemandParserHelper.cpp](#)

24.64 TRADEMGEN::DemandParserHelper::storeTripCode Struct Reference

`#include <trademgen/command/DemandParserHelper.hpp>` Inheritance diagram for TRADEMGEN::DemandParserHelper::storeTripCode::



Public Member Functions

- [storeTripCode](#) ([DemandStruct](#) &)
- [void operator\(\)](#) ([iterator_t](#) iStr, [iterator_t](#) iStrEnd) const

Public Attributes

- [DemandStruct](#) & [_demand](#)

24.64.1 Detailed Description

Store the trip type code.

Definition at line 167 of file [DemandParserHelper.hpp](#).

24.64.2 Constructor & Destructor Documentation

24.64.2.1 TRADEMGEN::DemandParserHelper::storeTripCode::storeTripCode ([DemandStruct](#) & *ioDemand*)

Actor Constructor.

Definition at line 254 of file [DemandParserHelper.cpp](#).

24.64.3 Member Function Documentation

24.64.3.1 void TRADEMGEN::DemandParserHelper::storeTripCode::operator() ([iterator_t](#) iStr, [iterator_t](#) iStrEnd) const

Actor Function (functor).

Definition at line 259 of file [DemandParserHelper.cpp](#).

References [TRADEMGEN::DemandParserHelper::ParserSemanticAction::_demand](#), and [TRADEMGEN::DemandStruct::_itTripCode](#).

24.64.4 Member Data Documentation

24.64.4.1 DemandStruct& TRADEMGEN::DemandParserHelper::ParserSemanticAction::_-demand [inherited]

Actor Context.

Definition at line 35 of file [DemandParserHelper.hpp](#).

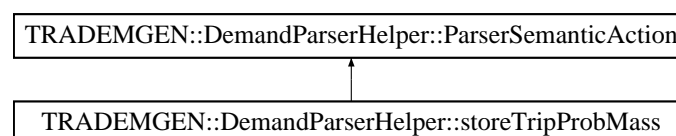
Referenced by [TRADEMGEN::DemandParserHelper::doEndDemand::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDTDProbMass::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDTD::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeTimeValueProbMass::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeTimeValue::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeWTP::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storePrefDepTimeProbMass::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storePrefDepTime::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeFFProbMass::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeFFCode::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeStayProbMass::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeStayCode::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeTripProbMass::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeChannelProbMass::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeChannelCode::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storePosProbMass::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storePosCode::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDemandNonRefundableProb::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDemandChangeFeeDisutility::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDemandChangeFeeProb::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDemandStdDev::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDemandMean::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storePrefCabin::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDestination::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeOrigin::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDow::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storePrefDepDateRangeEnd::operator\(\)](#) and [TRADEMGEN::DemandParserHelper::storePrefDepDateRangeStart::operator\(\)](#).

The documentation for this struct was generated from the following files:

- [trademgen/command/DemandParserHelper.hpp](#)
- [trademgen/command/DemandParserHelper.cpp](#)

24.65 TRADEMGEN::DemandParserHelper::storeTripProbMass Struct Reference

`#include <trademgen/command/DemandParserHelper.hpp>` Inheritance diagram for TRADEMGEN::DemandParserHelper::storeTripProbMass:



Public Member Functions

- [storeTripProbMass](#) ([DemandStruct](#) &)
- void [operator\(\)](#) (double *iReal*) const

Public Attributes

- [DemandStruct](#) & [_demand](#)

24.65.1 Detailed Description

Store the trip type probability mass.

Definition at line 175 of file [DemandParserHelper.hpp](#).

24.65.2 Constructor & Destructor Documentation**24.65.2.1 TRADEMGEN::DemandParserHelper::storeTripProbMass::storeTripProbMass (DemandStruct & *ioDemand*)**

Actor Constructor.

Definition at line 266 of file [DemandParserHelper.cpp](#).

24.65.3 Member Function Documentation**24.65.3.1 void TRADEMGEN::DemandParserHelper::storeTripProbMass::operator() (double *iReal*) const**

Actor Function (functor).

Definition at line 271 of file [DemandParserHelper.cpp](#).

References [TRADEMGEN::DemandParserHelper::ParserSemanticAction::_demand](#), [TRADEMGEN::DemandStruct::_itTripCode](#), and [TRADEMGEN::DemandStruct::_tripProbDist](#).

24.65.4 Member Data Documentation**24.65.4.1 DemandStruct& TRADEMGEN::DemandParserHelper::ParserSemanticAction::_demand [inherited]**

Actor Context.

Definition at line 35 of file [DemandParserHelper.hpp](#).

Referenced by [TRADEMGEN::DemandParserHelper::doEndDemand::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDTDProbMass::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDTD::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeTimeValueProbMass::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeTimeValue::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeWTP::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storePrefDepTimeProbMass::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storePrefDepTime::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeFFProbMass::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeFFCode::operator\(\)](#), and [TRADEMGEN::DemandParserHelper::storeStayProbMass::operator\(\)](#).


```

TRADEMGEN::DemandParserHelper::storeStayCode::operator(),           operator(),
TRADEMGEN::DemandParserHelper::storeTripCode::operator(),          TRADEM-
GEN::DemandParserHelper::storeChannelProbMass::operator(),          TRADEM-
GEN::DemandParserHelper::storeChannelCode::operator(),              TRADEM-
GEN::DemandParserHelper::storePosProbMass::operator(),              TRADEM-
GEN::DemandParserHelper::storePosCode::operator(), TRADEMGEN::DemandParserHelper::storeDemandNonRefundable()
TRADEMGEN::DemandParserHelper::storeDemandNonRefundableProb::operator(),
TRADEMGEN::DemandParserHelper::storeDemandChangeFeeDisutility::operator(),
TRADEMGEN::DemandParserHelper::storeDemandChangeFeeProb::operator(),
TRADEMGEN::DemandParserHelper::storeDemandStdDev::operator(),      TRADEM-
GEN::DemandParserHelper::storeDemandMean::operator(),              TRADEM-
GEN::DemandParserHelper::storePrefCabin::operator(), TRADEMGEN::DemandParserHelper::storeDestination::operator(),
TRADEMGEN::DemandParserHelper::storeOrigin::operator(),            TRADEM-
GEN::DemandParserHelper::storeDow::operator(), TRADEMGEN::DemandParserHelper::storePrefDepDateRangeEnd::operator()
and TRADEMGEN::DemandParserHelper::storePrefDepDateRangeStart::operator().

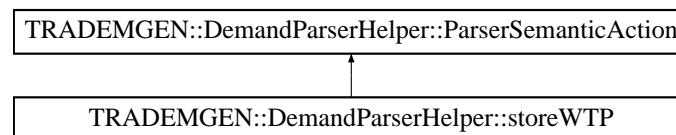
```

The documentation for this struct was generated from the following files:

- trademgen/command/[DemandParserHelper.hpp](#)
- trademgen/command/[DemandParserHelper.cpp](#)

24.66 TRADEMGEN::DemandParserHelper::storeWTP Struct Reference

`#include <trademgen/command/DemandParserHelper.hpp>` Inheritance diagram for TRADEMGEN::DemandParserHelper::storeWTP:



Public Member Functions

- [storeWTP](#) ([DemandStruct](#) &)
- void [operator\(\)](#) (double iReal) const

Public Attributes

- [DemandStruct](#) & [_demand](#)

24.66.1 Detailed Description

Store the parameters for the min Willingness-To-Pay (WTP).

Definition at line 233 of file [DemandParserHelper.hpp](#).

24.66.2 Constructor & Destructor Documentation

24.66.2.1 TRADEMGEN::DemandParserHelper::storeWTP::storeWTP (DemandStruct & ioDemand)

Actor Constructor.

Definition at line 393 of file [DemandParserHelper.cpp](#).

24.66.3 Member Function Documentation

24.66.3.1 void TRADEMGEN::DemandParserHelper::storeWTP::operator() (double iReal) const

Actor Function (functor).

Definition at line 398 of file [DemandParserHelper.cpp](#).

References [TRADEMGEM::DemandParserHelper::ParserSemanticAction::_demand](#), and [TRADEMGEM::DemandStruct::_minWTP](#).

24.66.4 Member Data Documentation

24.66.4.1 DemandStruct& TRADEMGEN::DemandParserHelper::ParserSemanticAction::_demand [inherited]

Actor Context.

Definition at line 35 of file [DemandParserHelper.hpp](#).

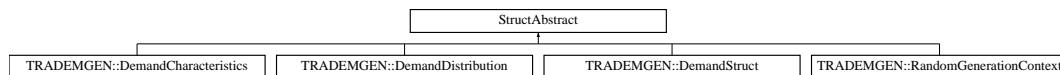
Referenced by [TRADEMGEM::DemandParserHelper::doEndDemand::operator\(\)\(\)](#), [TRADEMGEM::DemandParserHelper::storeDTDProbMass::operator\(\)\(\)](#), [TRADEMGEM::DemandParserHelper::storeDTD::operator\(\)\(\)](#), [TRADEMGEM::DemandParserHelper::storeTimeValueProbMass::operator\(\)\(\)](#), [TRADEMGEM::DemandParserHelper::storeTimeValue::operator\(\)\(\)](#), [TRADEMGEM::DemandParserHelper::storePrefDepTimeProbMass::operator\(\)\(\)](#), [TRADEMGEM::DemandParserHelper::storePrefDepTime::operator\(\)\(\)](#), [TRADEMGEM::DemandParserHelper::storeFFProbMass::operator\(\)\(\)](#), [TRADEMGEM::DemandParserHelper::storeFFCode::operator\(\)\(\)](#), [TRADEMGEM::DemandParserHelper::storeStayProbMass::operator\(\)\(\)](#), [TRADEMGEM::DemandParserHelper::storeStayCode::operator\(\)\(\)](#), [TRADEMGEM::DemandParserHelper::storeTripProbMass::operator\(\)\(\)](#), [TRADEMGEM::DemandParserHelper::storeTripCode::operator\(\)\(\)](#), [TRADEMGEM::DemandParserHelper::storeChannelProbMass::operator\(\)\(\)](#), [TRADEMGEM::DemandParserHelper::storeChannelCode::operator\(\)\(\)](#), [TRADEMGEM::DemandParserHelper::storePosProbMass::operator\(\)\(\)](#), [TRADEMGEM::DemandParserHelper::storePosCode::operator\(\)\(\)](#), [TRADEMGEM::DemandParserHelper::storeDemandNonRefundable::operator\(\)\(\)](#), [TRADEMGEM::DemandParserHelper::storeDemandNonRefundableProb::operator\(\)\(\)](#), [TRADEMGEM::DemandParserHelper::storeDemandChangeFeeDisutility::operator\(\)\(\)](#), [TRADEMGEM::DemandParserHelper::storeDemandChangeFeeProb::operator\(\)\(\)](#), [TRADEMGEM::DemandParserHelper::storeDemandStdDev::operator\(\)\(\)](#), [TRADEMGEM::DemandParserHelper::storeDemandMean::operator\(\)\(\)](#), [TRADEMGEM::DemandParserHelper::storePrefCabin::operator\(\)\(\)](#), [TRADEMGEM::DemandParserHelper::storeDestination::operator\(\)\(\)](#), [TRADEMGEM::DemandParserHelper::storeOrigin::operator\(\)\(\)](#), [TRADEMGEM::DemandParserHelper::storeDow::operator\(\)\(\)](#), [TRADEMGEM::DemandParserHelper::storePrefDepDateRangeEnd::operator\(\)\(\)](#), and [TRADEMGEM::DemandParserHelper::storePrefDepDateRangeStart::operator\(\)\(\)](#).

The documentation for this struct was generated from the following files:

- trademgen/command/[DemandParserHelper.hpp](#)
- trademgen/command/[DemandParserHelper.cpp](#)

24.67 StructAbstract Class Reference

Inheritance diagram for StructAbstract::

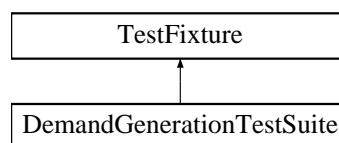


The documentation for this class was generated from the following files:

- trademgen/basic/[RandomGenerationContext.hpp](#)
- trademgen/basic/[DemandCharacteristics.hpp](#)
- trademgen/bom/[DemandStruct.hpp](#)
- trademgen/basic/[DemandDistribution.hpp](#)

24.68 TestFixture Class Reference

Inheritance diagram for TestFixture::

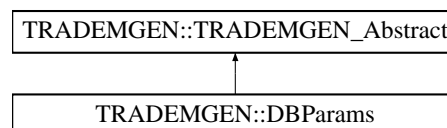


The documentation for this class was generated from the following file:

- test/trademgen/[DemandGenerationTestSuite.hpp](#)

24.69 TRADEMGEN::TRADEMGEN_Abstract Struct Reference

#include <trademgen/TRADEMGEN_Abstract.hpp>Inheritance diagram for TRADEMGEN::TRADEMGEN_Abstract::



Public Member Functions

- virtual void [toStream](#) (std::ostream &ioOut) const =0
- virtual void [fromStream](#) (std::istream &ioIn)=0
- virtual std::string [toString](#) () const =0

Protected Member Functions

- [TRADEMGEN_Abstract \(\)](#)
- [TRADEMGEN_Abstract \(const TRADEMGEN_Abstract &\)](#)
- virtual [~TRADEMGEN_Abstract \(\)](#)

24.69.1 Detailed Description

Base class for the [TRADEMGEN](#) interface structures.

Definition at line 16 of file [TRADEMGEN_Abstract.hpp](#).

24.69.2 Constructor & Destructor Documentation

24.69.2.1 TRADEMGEN::TRADEMGEN_Abstract::TRADEMGEN_Abstract () [inline, protected]

Protected Default Constructor to ensure this class is abstract.

Definition at line 33 of file [TRADEMGEN_Abstract.hpp](#).

24.69.2.2 TRADEMGEN::TRADEMGEN_Abstract::TRADEMGEN_Abstract (const TRADEMGEN_Abstract &) [inline, protected]

Definition at line 34 of file [TRADEMGEN_Abstract.hpp](#).

24.69.2.3 virtual TRADEMGEN::TRADEMGEN_Abstract::~~TRADEMGEN_Abstract () [inline, protected, virtual]

Destructor.

Definition at line 37 of file [TRADEMGEN_Abstract.hpp](#).

24.69.3 Member Function Documentation

24.69.3.1 virtual void TRADEMGEN::TRADEMGEN_Abstract::toStream (std::ostream & ioOut) const [pure virtual]

Dump a structure into an output stream.

Parameters:

ostream& the output stream.

Implemented in [TRADEMGEN::DBParams](#).

24.69.3.2 virtual void TRADEMGEN::TRADEMGEN_Abstract::fromStream (std::istream & ioIn) [pure virtual]

Read a structure from an input stream.

Parameters:

istream& the input stream.

Implemented in [TRADEMGEN::DBParams](#).

Referenced by [operator>>\(\)](#).

24.69.3.3 virtual std::string TRADEMGEN::TRADEMGEN_Abstract::toString () const [pure virtual]

Get the serialised version of the structure.

Implemented in [TRADEMGEN::DBParams](#).

The documentation for this struct was generated from the following file:

- [trademgen/TRADEMGEN_Abstract.hpp](#)

24.70 TRADEMGEN::TRADEMGEN_Service Class Reference

class holding the services related to Travel Demand Generation.

```
#include <trademgen/TRADEMGEN_Service.hpp>
```

Public Member Functions

- [TRADEMGEN_Service](#) (const stdair::BasLogParams &, const stdair::BasDBParams &, const stdair::RandomSeed_T &)

Constructor.

- [TRADEMGEN_Service](#) (const stdair::BasLogParams &, const stdair::RandomSeed_T &)
- [TRADEMGEN_Service](#) (stdair::STDAIR_ServicePtr_T, SEVMGR::SEVMGR_ServicePtr_T, const stdair::RandomSeed_T &)
- void [parseAndLoad](#) (const [DemandFilePath](#) &)
- [~TRADEMGEN_Service](#) ()
- void [buildSampleBom](#) ()
- void [clonePersistentBom](#) ()
- void [buildComplementaryLinks](#) (stdair::BomRoot &)
- stdair::BookingRequestStruct [buildSampleBookingRequest](#) (const bool isForCRS=false)
- void [displayAirlineListFromDB](#) () const
- const stdair::Count_T & [getExpectedTotalNumberOfRequestsToBeGenerated](#) () const
- const stdair::Count_T & [getActualTotalNumberOfRequestsToBeGenerated](#) () const
- const bool [stillHavingRequestsToBeGenerated](#) (const stdair::DemandStreamKeyStr_T &, stdair::ProgressStatusSet &, const stdair::DemandGenerationMethod &) const
- stdair::Count_T [generateFirstRequests](#) (const stdair::DemandGenerationMethod &) const
- stdair::BookingRequestPtr_T [generateNextRequest](#) (const stdair::DemandStreamKeyStr_T &, const stdair::DemandGenerationMethod &) const
- bool [hasDemandStream](#) (const stdair::DemandStreamKeyStr_T &) const
- stdair::ProgressStatusSet [popEvent](#) (stdair::EventStruct &) const
- bool [isQueueDone](#) () const
- bool [generateCancellation](#) (const stdair::TravelSolutionStruct &, const stdair::PartySize_T &, const stdair::DateTime_T &, const stdair::Date_T &) const

- void [reset](#) () const
- const stdair::ProgressStatus & [getProgressStatus](#) () const
- const stdair::ProgressStatus & [getProgressStatus](#) (const stdair::EventType::EN_EventType &) const
- std::string [jsonHandler](#) (const stdair::JSONString &) const
- std::string [csvDisplay](#) () const
- std::string [list](#) () const
- std::string [list](#) (const stdair::EventType::EN_EventType &) const
- std::string [displayDemandStream](#) () const

24.70.1 Detailed Description

class holding the services related to Travel Demand Generation.

Definition at line 44 of file [TRADEMGEN_Service.hpp](#).

24.70.2 Constructor & Destructor Documentation

24.70.2.1 TRADEMGEN::TRADEMGEN_Service::TRADEMGEN_Service (const stdair::BasLogParams & *iLogParams*, const stdair::BasDBParams & *iDBParams*, const stdair::RandomSeed_T & *iRandomSeed*)

Constructor. The `initTrademgenService()` method is called; see the corresponding documentation for more details.

A reference on an output stream is given, so that log outputs can be directed onto that stream.

Moreover, database connection parameters are given, so that a session can be created on the corresponding database.

Parameters:

const stdair::BasLogParams& Parameters for the output log stream.

const stdair::BasDBParams& Parameters for the database access.

const stdair::RandomSeed_T& Seed for the random generation.

Definition at line 78 of file [TRADEMGEN_Service.cpp](#).

24.70.2.2 TRADEMGEN::TRADEMGEN_Service::TRADEMGEN_Service (const stdair::BasLogParams & *iLogParams*, const stdair::RandomSeed_T & *iRandomSeed*)

Constructor.

The `initTrademgenService()` method is called; see the corresponding documentation for more details.

A reference on an output stream is given, so that log outputs can be directed onto that stream.

Parameters:

const stdair::BasLogParams& Parameters for the output log stream.

const stdair::RandomSeed_T& Seed for the random generation.

Definition at line 54 of file [TRADEMGEN_Service.cpp](#).

24.70.2.3 TRADEMGEN::TRADEMGEN_Service::TRADEMGEN_Service (stdair::STDAIR_ServicePtr_T ioSTDAIR_Service_ptr, SEVMGR::SEVMGR_ServicePtr_T ioSEVMGR_Service_ptr, const stdair::RandomSeed_T & iRandomSeed)

Constructor.

The initTrademgenService() method is called; see the corresponding documentation for more details.

Moreover, as no reference on any output stream is given, neither any database access parameter is given, it is assumed that the StdAir log service has already been initialised with the proper log output stream by some other methods in the calling chain (for instance, when the [TRADEMGEN_Service](#) is itself being initialised by another library service such as TVLSIM_Service).

Parameters:

stdair::STDAIR_ServicePtr_T Handler on the STDAIR_Service.

SEVMGR::SEVMGR_ServicePtr_T Handler on the SEVMGR_Service.

const stdair::RandomSeed_T & Seed for the random generation.

Definition at line 104 of file [TRADEMGEN_Service.cpp](#).

24.70.2.4 TRADEMGEN::TRADEMGEN_Service::~~TRADEMGEN_Service ()

Destructor.

Definition at line 125 of file [TRADEMGEN_Service.cpp](#).

24.70.3 Member Function Documentation

24.70.3.1 void TRADEMGEN::TRADEMGEN_Service::parseAndLoad (const DemandFilePath & iDemandFilePath)

Parse the demand input file.

The CSV file, describing the parameters of the demand to be generated for the simulator, is parsed and instantiated in memory accordingly.

Parameters:

const DemandFilePath & Filename of the input demand file.

Definition at line 241 of file [TRADEMGEN_Service.cpp](#).

References [buildComplementaryLinks\(\)](#), [clonePersistentBom\(\)](#), and [generateDemand\(\)](#).

Referenced by [TRADEMGEN::Trademgener::init\(\)](#), and [main\(\)](#).

24.70.3.2 void TRADEMGEN::TRADEMGEN_Service::buildSampleBom ()

Build a sample BOM tree, made of a single [DemandStream](#) object.

As of now (March 2011), it corresponds to:

- Origin: SIN

- Destination: BKK
 - Preferred departure date: 2011-02-14
 - Preferred cabin: Y (Economy)
 - POS distribution:
 - BKK: 30%
 - SIN: 70%
 - Channel distribution:
 - Direct Offline: 10%
 - Direct Online: 30%
 - Indirect Offline: 40%
 - Indirect Online: 20%
 - Trip type distribution:
 - Outbound: 60%
 - Inbound: 20%
 - One-way: 20%
 - Arrival pattern distribution:
 - 330 DTD: 0%
 - 40 DTD: 20%
 - 20 DTD: 60%
 - 1 DTD: 100%
- 15:0, 60:1
- Stay duration distribution:
 - 0 day: 10%
 - 1 day: 10%
 - 2 days: 15%
 - 3 days: 15%
 - 4 days: 15%
 - 5 days: 35%
 - Frequent flyer distribution:
 - Platinum: 1%
 - Gold: 5%
 - Silver: 15%
 - Member: 30%
 - No card: 49%
 - Preferred departure time (cumulative distribution):
 - 6am: 0%
 - 7am: 10%

- 9am: 30%
- 5pm: 40%
- 7pm: 80%
- 8pm: 95%
- 10pm: 100%
- Value of time distribution:
 - 15 min: 0%
 - 60 min: 100%
- WTP: 200
- Number of requests: Normal ($\mu = 10.0$, $\text{std_dev} = 1.0$)
- Change fee: 20; Non refundable; Saturday night stay

Definition at line 310 of file [TRADEMGEN_Service.cpp](#).

References [buildComplementaryLinks\(\)](#), and [clonePersistentBom\(\)](#).

Referenced by [TRADEMGEN::Trademgener::init\(\)](#), and [main\(\)](#).

24.70.3.3 void TRADEMGEN::TRADEMGEN_Service::clonePersistentBom ()

Clone the persistent BOM object.

Definition at line 382 of file [TRADEMGEN_Service.cpp](#).

References [buildComplementaryLinks\(\)](#).

Referenced by [buildSampleBom\(\)](#), and [parseAndLoad\(\)](#).

24.70.3.4 void TRADEMGEN::TRADEMGEN_Service::buildComplementaryLinks (stdair::BomRoot & ioBomRoot)

Build all the complementary links in the given bom root object.

Note:

Do nothing for now.

Definition at line 419 of file [TRADEMGEN_Service.cpp](#).

Referenced by [buildSampleBom\(\)](#), [clonePersistentBom\(\)](#), and [parseAndLoad\(\)](#).

24.70.3.5 stdair::BookingRequestStruct TRADEMGEN::TRADEMGEN_Service::buildSampleBookingRequest (const bool isForCRS = false)

Build a sample booking request structure.

As of now (March 2011), the sample booking request is made of the following parameters:

- Return trip (inbound): LHR-SYD (POS: LHR, Channel: DN),

- Departing 10-JUN-2011 around 8:00, staying 7 days
- Requested on 15-MAY-2011 at 10:00
- Economy cabin, 3 persons, FF member
- WTP: 1000.0 EUR
- Dis-utility: 100.0 EUR/hour

As of now (March 2011), the CRS-related booking request is made of the following parameters:

- Return trip (inbound): SIN-BKK (POS: SIN, Channel: IN),
- Departing 30-JAN-2010 around 10:00, staying 7 days
- Requested on 22-JAN-2010 at 10:00
- Economy cabin, 3 persons, FF member
- WTP: 1000.0 EUR
- Dis-utility: 100.0 EUR/hour

See also:

stdair::CmdBomManager for more details.

Parameters:

const bool isForCRS Whether the sample booking request is for CRS.

Returns:

BookingRequestStruct& Sample booking request structure.

Definition at line 425 of file [TRADEMGEN_Service.cpp](#).

24.70.3.6 void TRADEMGEN::TRADEMGEN_Service::displayAirlineListFromDB () const

Display the list of airlines, as held within the sample database.

Definition at line 535 of file [TRADEMGEN_Service.cpp](#).

Referenced by [main\(\)](#).

24.70.3.7 const stdair::Count_T & TRADEMGEN::TRADEMGEN_Service::getExpectedTotalNumberOfRequestsToBeGenerated () const

Get the expected number of events/requests to be generated for all the demand streams.

Calls the SEvMgr service with the same name "getExpectedTotalNumberOfRequestsToBeGenerated", which computes that number.

Note:

That number usually corresponds to an expectation (i.e., the mean value of a random distribution). The actual number will be drawn when calling the [generateFirstRequests\(\)](#) method.

Returns:

const stdair::Count_T& Expected number of events to be generated.

Definition at line 596 of file [TRADEMGEN_Service.cpp](#).

Referenced by [generateDemand\(\)](#), and [TRADEMGEN::Trademgener::trademgen\(\)](#).

24.70.3.8 const stdair::Count_T & TRADEMGEN::TRADEMGEN_Service::getActualTotalNumberOfRequestsToBeGenerated () const

Get the actual number of events/requests to be generated for all the demand streams.

Calls the SEvMgr service with the same name "getActualTotalNumberOfRequestsToBeGenerated", which computes that number.

Note:

That number has been drawn when calling the [generateFirstRequests\(\)](#) method.

Returns:

const stdair::Count_T& Expected number of events to be generated.

Definition at line 617 of file [TRADEMGEN_Service.cpp](#).

24.70.3.9 const bool TRADEMGEN::TRADEMGEN_Service::stillHavingRequestsToBeGenerated (const stdair::DemandStreamKeyStr_T & iKey, stdair::ProgressStatusSet & ioPSS, const stdair::DemandGenerationMethod & iDemandGenerationMethod) const

Check whether enough requests have already been generated for the demand stream which corresponds to the given key.

Parameters:

const DemandStreamKey& A string identifying uniquely the demand stream (e.g., "SIN-HND 2010-Feb-08 Y").

const stdair::DemandGenerationMethod& States whether the demand generation must be performed following the method based on statistic orders. The alternative method, while more "intuitive", is also a sequential algorithm.

Returns:

bool Whether or not there are still events to be generated for that demand stream.

Definition at line 638 of file [TRADEMGEN_Service.cpp](#).

24.70.3.10 stdair::Count_T TRADEMGEN::TRADEMGEN_Service::generateFirstRequests (const stdair::DemandGenerationMethod & iDemandGenerationMethod) const

Browse the list of demand streams and generate the first request of each stream.

Parameters:

const `stdair::DemandGenerationMethod&` States whether the demand generation must be performed following the method based on statistic orders. The alternative method, while more "intuitive", is also a sequential algorithm.

Returns:

`stdair::Count_T` The expected total number of events to be generated

Definition at line 663 of file [TRADEMGEN_Service.cpp](#).

Referenced by [generateDemand\(\)](#), and [TRADEMGEN::Trademgener::trademgen\(\)](#).

24.70.3.11 `stdair::BookingRequestPtr_T TRADEMGEN::TRADEMGEN_Service::generateNextRequest (const stdair::DemandStreamKeyStr_T & iKey, const stdair::DemandGenerationMethod & iDemandGenerationMethod) const`

Generate a request with the demand stream which corresponds to the given key.

Parameters:

const [DemandStreamKey](#)& A string identifying uniquely the demand stream (e.g., "SIN-HND 2010-Feb-08 Y").

const `stdair::DemandGenerationMethod&` States whether the demand generation must be performed following the method based on statistic orders. The alternative method, while more "intuitive", is also a sequential algorithm.

Returns:

`stdair::BookingRequestPtr_T` (Boost) shared pointer on the booking request structure, which has just been created.

Definition at line 689 of file [TRADEMGEN_Service.cpp](#).

Referenced by [generateDemand\(\)](#), and [TRADEMGEN::Trademgener::trademgen\(\)](#).

24.70.3.12 `bool TRADEMGEN::TRADEMGEN_Service::hasDemandStream (const stdair::DemandStreamKeyStr_T & iDemandStreamKey) const`

States whether a demand stream with the given key is used to generate demand.

Parameters:

const [DemandStreamKey](#)& A string identifying uniquely the demand stream (e.g., "SIN-HND 2010-Feb-08 Y").

Definition at line 868 of file [TRADEMGEN_Service.cpp](#).

24.70.3.13 `stdair::ProgressStatusSet TRADEMGEN::TRADEMGEN_Service::popEvent (stdair::EventStruct & ioEventStruct) const`

Pop the next coming (in time) event, and remove it from the event queue thanks to the SEvMgr service.

- The next coming (in time) event corresponds to the event having the earliest date-time stamp. In other words, it is the first/front element of the event queue.
- That (first) event/element is then removed from the event queue
- The progress status is updated for the corresponding demand stream.

Returns:

stdair::EventStruct A copy of the event structure, which comes first in time from within the event queue.

Definition at line 713 of file [TRADEMGEN_Service.cpp](#).

Referenced by [generateDemand\(\)](#), and [TRADEMGEN::Trademgener::trademgen\(\)](#).

24.70.3.14 bool TRADEMGEN::TRADEMGEN_Service::isQueueDone () const

States whether the event queue has reached the end.

Calls the SEvMgr service with the same name "isQueueDone", which states whether the event queue has reached the end.

For now, that method states whether the event queue is empty.

Definition at line 729 of file [TRADEMGEN_Service.cpp](#).

Referenced by [generateDemand\(\)](#), and [TRADEMGEN::Trademgener::trademgen\(\)](#).

24.70.3.15 bool TRADEMGEN::TRADEMGEN_Service::generateCancellation (const stdair::TravelSolutionStruct & *iTravelSolution*, const stdair::PartySize_T & *iPartySize*, const stdair::DateTime_T & *iRequestTime*, const stdair::Date_T & *iDepartureDate*) const

Generate the potential cancellation event.

Definition at line 749 of file [TRADEMGEN_Service.cpp](#).

24.70.3.16 void TRADEMGEN::TRADEMGEN_Service::reset () const

Reset the context of the demand streams for another demand generation without having to reparse the demand input file.

Definition at line 811 of file [TRADEMGEN_Service.cpp](#).

Referenced by [generateDemand\(\)](#), and [TRADEMGEN::Trademgener::trademgen\(\)](#).

24.70.3.17 const stdair::ProgressStatus & TRADEMGEN::TRADEMGEN_Service::getProgressStatus () const

Get the overall progress status (for the whole event queue).

Definition at line 832 of file [TRADEMGEN_Service.cpp](#).

24.70.3.18 `const stdair::ProgressStatus & TRADEMGEN::TRADEMGEN_Service::getProgressStatus (const stdair::EventType::EN_EventType & iEventType) const`

Get the progress status for the given event type (e.g., booking request, optimisation notification, schedule change, break point).

Definition at line 850 of file [TRADEMGEN_Service.cpp](#).

24.70.3.19 `std::string TRADEMGEN::TRADEMGEN_Service::jsonHandler (const stdair::JSONString & iJSONString) const`

Dispatch the JSon command string to the SEvMgr service. (Only SEvMgr has json export commands for now).

Parameters:

const stdair::JSONString& Input string which contained the JSon command string.

Returns:

std::string Output string in which the asking objects are logged/dumped with a JSon format.

Definition at line 447 of file [TRADEMGEN_Service.cpp](#).

24.70.3.20 `std::string TRADEMGEN::TRADEMGEN_Service::csvDisplay () const`

Recursively display (dump in the returned string) the objects of the BOM tree.

Returns:

std::string Output string in which the BOM tree is logged/dumped.

Definition at line 468 of file [TRADEMGEN_Service.cpp](#).

Referenced by [generateDemand\(\)](#), and [TRADEMGEN::Trademgener::trademgen\(\)](#).

24.70.3.21 `std::string TRADEMGEN::TRADEMGEN_Service::list () const`

Display (dump in the returned string) the event list of the event queue.

Returns:

std::string Output string in which the events are logged/dumped.

Definition at line 490 of file [TRADEMGEN_Service.cpp](#).

24.70.3.22 `std::string TRADEMGEN::TRADEMGEN_Service::list (const stdair::EventType::EN_EventType & iEventType) const`

Display (dump in the returned string) the events with the given type

Returns:

std::string Output string in which the events are logged/dumped.

Definition at line 513 of file [TRADEMGEN_Service.cpp](#).

24.70.3.23 std::string TRADEMGEN::TRADEMGEN_Service::displayDemandStream () const

Display (dump in the returned string) the demand streams

Returns:

std::string Output string in which the demand streams are logged/dumped.

Definition at line 885 of file [TRADEMGEN_Service.cpp](#).

References [TRADEMGEN::DemandStream::describeKey\(\)](#).

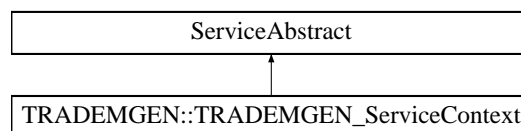
The documentation for this class was generated from the following files:

- [trademgen/TRADEMGEN_Service.hpp](#)
- [trademgen/service/TRADEMGEN_Service.cpp](#)

24.71 TRADEMGEN::TRADEMGEN_ServiceContext Class Reference

Class holding the context of the Trademgen services.

`#include <trademgen/service/TRADEMGEN_ServiceContext.hpp>` Inheritance diagram for TRADEMGEN::TRADEMGEN_ServiceContext::

**Friends**

- class [TRADEMGEN_Service](#)
- class [FacTRADEMGENServiceContext](#)

24.71.1 Detailed Description

Class holding the context of the Trademgen services.

Definition at line 34 of file [TRADEMGEN_ServiceContext.hpp](#).

24.71.2 Friends And Related Function Documentation**24.71.2.1 friend class TRADEMGEN_Service [friend]**

The [TRADEMGEN_Service](#) class should be the sole class to get access to ServiceContext content: general users do not want to bother with a context interface.

Definition at line 40 of file [TRADEMGEN_ServiceContext.hpp](#).

24.71.2.2 friend class FacTRADEMGENServiceContext [**friend**]

Definition at line 41 of file [TRADEMGEN_ServiceContext.hpp](#).

The documentation for this class was generated from the following files:

- [trademgen/service/TRADEMGEN_ServiceContext.hpp](#)
- [trademgen/service/TRADEMGEN_ServiceContext.cpp](#)

24.72 TRADEMGEN::Trademgener Struct Reference

Wrapper structure around the C++ API, so as to expose a Python API.

Public Member Functions

- `std::string trademgen` (const `NbOfRuns_T` &iNbOfRuns, const `std::string` &iDemandGeneration-MethodString)
- `Trademgener` ()
- `Trademgener` (const `Trademgener` &iTrademgener)
- `~Trademgener` ()
- `bool init` (const `std::string` &iLogFilepath, const `stdair::RandomSeed_T` &iRandomSeed, const `bool` isBuiltin, const `stdair::Filename_T` &iDemandInputFilename)

24.72.1 Detailed Description

Wrapper structure around the C++ API, so as to expose a Python API.

Definition at line 70 of file [pytrademgen.cpp](#).

24.72.2 Constructor & Destructor Documentation

24.72.2.1 TRADEMGEN::Trademgener::Trademgener () [**inline**]

Default constructor.

Definition at line 266 of file [pytrademgen.cpp](#).

24.72.2.2 TRADEMGEN::Trademgener::Trademgener (const Trademgener & iTrademgener) [**inline**]

Default copy constructor.

Definition at line 270 of file [pytrademgen.cpp](#).

24.72.2.3 TRADEMGEN::Trademgener::~~Trademgener () [inline]

Default constructor.

Definition at line 276 of file [pytrademgen.cpp](#).**24.72.3 Member Function Documentation****24.72.3.1 std::string TRADEMGEN::Trademgener::trademgen (const NbOfRuns_T & iNbOfRuns, const std::string & iDemandGenerationMethodString) [inline]**

Wrapper around the travel demand generation use case.

Definition at line 76 of file [pytrademgen.cpp](#).

References [TRADEMGEN::TRADEMGEN_Service::csvDisplay\(\)](#), [TRADEMGEN::TRADEMGEN_Service::generateFirstRequests\(\)](#), [TRADEMGEN::TRADEMGEN_Service::generateNextRequest\(\)](#), [TRADEMGEN::TRADEMGEN_Service::getExpectedTotalNumberOfRequestsToBeGenerated\(\)](#), [TRADEMGEN::TRADEMGEN_Service::isQueueDone\(\)](#), [TRADEMGEN::TRADEMGEN_Service::popEvent\(\)](#), [TRADEMGEN::TRADEMGEN_Service::reset\(\)](#), and [TRADEMGEN::stat_display\(\)](#).

Referenced by [BOOST_PYTHON_MODULE\(\)](#).**24.72.3.2 bool TRADEMGEN::Trademgener::init (const std::string & iLogFilepath, const stdair::RandomSeed_T & iRandomSeed, const bool isBuiltin, const stdair::Filename_T & iDemandInputFilename) [inline]**

Wrapper around the search use case.

Definition at line 284 of file [pytrademgen.cpp](#).

References [TRADEMGEN::TRADEMGEN_Service::buildSampleBom\(\)](#), and [TRADEMGEN::TRADEMGEN_Service::parseAndLoad\(\)](#).

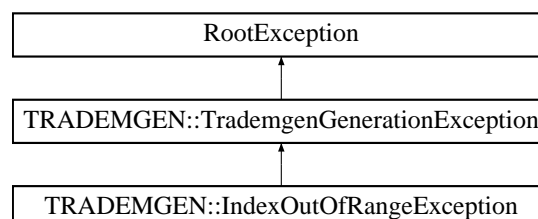
Referenced by [BOOST_PYTHON_MODULE\(\)](#).

The documentation for this struct was generated from the following file:

- [trademgen/python/pytrademgen.cpp](#)

24.73 TRADEMGEN::TrademgenGenerationException Class Reference

#include <trademgen/TRADEMGEN_Exceptions.hpp> Inheritance diagram for TRADEMGEN::TrademgenGenerationException::



Public Member Functions

- [TrademgenGenerationException](#) (const std::string &iWhat)

24.73.1 Detailed Description

Root exception for the TraDemGen component

Definition at line 18 of file [TRADEMGEN_Exceptions.hpp](#).

24.73.2 Constructor & Destructor Documentation

24.73.2.1 TRADEMGEN::TrademgenGenerationException::TrademgenGenerationException (const std::string & *iWhat*) [inline]

Constructor.

Definition at line 23 of file [TRADEMGEN_Exceptions.hpp](#).

The documentation for this class was generated from the following file:

- trademgen/[TRADEMGEN_Exceptions.hpp](#)

25 File Documentation

- 25.1 doc/local/authors.doc File Reference**
- 25.2 doc/local/codingrules.doc File Reference**
- 25.3 doc/local/copyright.doc File Reference**
- 25.4 doc/local/documentation.doc File Reference**
- 25.5 doc/local/features.doc File Reference**
- 25.6 doc/local/help_wanted.doc File Reference**
- 25.7 doc/local/howto_release.doc File Reference**
- 25.8 doc/local/index.doc File Reference**
- 25.9 doc/local/installation.doc File Reference**
- 25.10 doc/local/linking.doc File Reference**
- 25.11 doc/local/test.doc File Reference**
- 25.12 doc/local/users_guide.doc File Reference**
- 25.13 doc/local/verification.doc File Reference**
- 25.14 doc/tutorial/tutorial.doc File Reference**
- 25.15 test/trademgen/DemandGenerationTestSuite.cpp File Reference**

25.16 DemandGenerationTestSuite.cpp

```

00001
00005 // //////////////////////////////////////
00006 // Import section
00007 // //////////////////////////////////////
00008 // STL
00009 #include <sstream>
00010 #include <fstream>
00011 #include <map>
00012 #include <cmath>
00013 // Boost Unit Test Framework (UTF)
00014 #define BOOST_TEST_DYN_LINK
00015 #define BOOST_TEST_MAIN
00016 #define BOOST_TEST_MODULE DemandGenerationTest
00017 #include <boost/test/unit_test.hpp>
00018 // StdAir
00019 #include <stdair/stdair_basic_types.hpp>
00020 #include <stdair/basic/BasConst_General.hpp>
00021 #include <stdair/basic/BasLogParams.hpp>
00022 #include <stdair/basic/BasDBParams.hpp>
00023 #include <stdair/basic/BasFileMgr.hpp>
00024 #include <stdair/basic/ProgressStatusSet.hpp>
00025 #include <stdair/bom/EventStruct.hpp>
00026 #include <stdair/bom/BookingRequestStruct.hpp>
00027 #include <stdair/service/Logger.hpp>
00028 // TraDemGen
00029 #include <trademgen/TRADEMGEN_Exceptions.hpp>
00030 #include <trademgen/TRADEMGEN_Service.hpp>
00031 #include <trademgen/bom/DemandStreamKey.hpp>
00032 #include <trademgen/config/trademgen-paths.hpp>
00033
00034 namespace boost_utf = boost::unit_test;
00035
00036 // (Boost) Unit Test XML Report
00037 std::ofstream utfReportStream ("DemandGenerationTestSuite_utfresults.xml");
00038
00042 struct UnitTestConfig {
00044     UnitTestConfig() {
00045         boost_utf::unit_test_log.set_stream (utfReportStream);
00046         boost_utf::unit_test_log.set_format (boost_utf::XML);
00047         boost_utf::unit_test_log.set_threshold_level (boost_utf::log_test_units);
00048         //boost_utf::unit_test_log.set_threshold_level (boost_utf::log_successful_tes
00049     ts);
00049     }
00050
00052     ~UnitTestConfig() {
00053     }
00054 };
00055
00056 // Specific type definitions
00057 typedef std::pair<stdair::Count_T, stdair::Count_T> NbOfEventsPair_T;
00058 typedef std::map<const stdair::DemandStreamKeyStr_T,
00059                 NbOfEventsPair_T> NbOfEventsByDemandStreamMap_T;
00060
00061 // //////////////////////////////////////
00065 void testDemandGenerationHelper (const unsigned short iTestFlag,
00066                                 const stdair::Filename_T& iDemandInputFilename,
00067                                 const stdair::DemandGenerationMethod& iDemandGen
00068                                 erationMethod,
00069                                 const bool isBuiltin) {
00069
00070     // Seed for the random generation
00071     const stdair::RandomSeed_T lRandomSeed = stdair::DEFAULT_RANDOM_SEED;
00072
00073     // Output log File
00074     std::ostringstream oStr;

```

```

00075 oStr << "DemandGenerationTestSuite_" << iTestFlag << ".log";
00076 const stdair::Filename_T lLogFilename (oStr.str());
00077
00078 // Set the log parameters
00079 std::ofstream logOutputFile;
00080 // Open and clean the log outputfile
00081 logOutputFile.open (lLogFilename.c_str());
00082 logOutputFile.clear();
00083
00084 // Initialise the TraDemGen service object
00085 const stdair::BasLogParams lLogParams (stdair::LOG::DEBUG, logOutputFile);
00086 TRADEMGEN::TRADEMGEN_Service trademgenService (lLogParams, lRandomSeed);
00087
00097 NbOfEventsByDemandStreamMap_T lNbOfEventsMap;
00098
00099 // Total number of events
00100 stdair::Count_T lRefExpectedNbOfEvents (0);
00101 stdair::Count_T lRefActualNbOfEvents (0);
00102
00103 // Check whether or not a (CSV) input file should be read
00104 if (isBuiltin == true) {
00105
00106     // Build the default sample BOM tree (filled with demand streams) for TraDemG
    en
00107     trademgenService.buildSampleBom();
00108
00109     lNbOfEventsMap.insert (NbOfEventsByDemandStreamMap_T::
00110         value_type ("SIN-BKK 2010-Feb-08 Y",
00111             NbOfEventsPair_T (4, 60)));
00112     lNbOfEventsMap.insert (NbOfEventsByDemandStreamMap_T::
00113         value_type ("BKK-HKG 2010-Feb-08 Y",
00114             NbOfEventsPair_T (4, 60)));
00115     lNbOfEventsMap.insert (NbOfEventsByDemandStreamMap_T::
00116         value_type ("SIN-HKG 2010-Feb-08 Y",
00117             NbOfEventsPair_T (4, 60)));
00118
00119     // Total number of events, for the 3 demand streams: 180
00120     lRefExpectedNbOfEvents = 180;
00121     lRefActualNbOfEvents = 186;
00122
00123 } else {
00124
00125     // Create the DemandStream objects, and insert them within the BOM tree
00126     const TRADEMGEN::DemandFilePath lDemandFilePath (iDemandInputFilename);
00127     trademgenService.parseAndLoad (lDemandFilePath);
00128
00129     lNbOfEventsMap.insert (NbOfEventsByDemandStreamMap_T::
00130         value_type ("SIN-HND 2010-Feb-08 Y",
00131             NbOfEventsPair_T (1, 10)));
00132     lNbOfEventsMap.insert (NbOfEventsByDemandStreamMap_T::
00133         value_type ("SIN-HND 2010-Feb-09 Y",
00134             NbOfEventsPair_T (1, 10)));
00135     lNbOfEventsMap.insert (NbOfEventsByDemandStreamMap_T::
00136         value_type ("SIN-BKK 2010-Feb-08 Y",
00137             NbOfEventsPair_T (1, 10)));
00138     lNbOfEventsMap.insert (NbOfEventsByDemandStreamMap_T::
00139         value_type ("SIN-BKK 2010-Feb-09 Y",
00140             NbOfEventsPair_T (1, 10)));
00141
00142     // Total number of events, for the 4 demand streams: 40
00143     lRefExpectedNbOfEvents = 40;
00144     lRefActualNbOfEvents = 40;
00145 }
00146
00147 // Retrieve the expected (mean value of the) number of events to be
00148 // generated
00149 const stdair::Count_T& lExpectedNbOfEventsToBeGenerated =

```

```

00150     trademgenService.getExpectedTotalNumberOfRequestsToBeGenerated();
00151
00152     BOOST_CHECK_EQUAL (lRefExpectedNbOfEvents,
00153         std::floor (lExpectedNbOfEventsToBeGenerated));
00154
00155     BOOST_CHECK_MESSAGE (lRefExpectedNbOfEvents ==
00156         std::floor (lExpectedNbOfEventsToBeGenerated),
00157         "Expected total number of requests to be generated: "
00158         << lExpectedNbOfEventsToBeGenerated
00159         << " (=> "
00160         << std::floor (lExpectedNbOfEventsToBeGenerated)
00161         << "). Reference value: " << lRefExpectedNbOfEvents);
00162
00163     const stdair::Count_T& lActualNbOfEventsToBeGenerated =
00164         trademgenService.generateFirstRequests(iDemandGenerationMethod);
00165
00166     // DEBUG
00167     STDAIR_LOG_DEBUG ("Expected number of events: "
00168         << lExpectedNbOfEventsToBeGenerated << ", actual: "
00169         << lActualNbOfEventsToBeGenerated);
00170
00171     // Total number of events, for all the demand streams:
00172     BOOST_CHECK_EQUAL (lRefActualNbOfEvents, lActualNbOfEventsToBeGenerated);
00173
00174     BOOST_CHECK_MESSAGE (lRefActualNbOfEvents == lActualNbOfEventsToBeGenerated,
00175         "Actual total number of requests to be generated: "
00176         << lExpectedNbOfEventsToBeGenerated
00177         << " (=> "
00178         << std::floor (lExpectedNbOfEventsToBeGenerated)
00179         << "). Reference value: " << lRefActualNbOfEvents);
00180
00181     const bool isQueueDone = trademgenService.isQueueDone();
00182     BOOST_REQUIRE_MESSAGE (isQueueDone == false,
00183         "The event queue should not be empty.");
00184
00185     stdair::Count_T idx = 1;
00186     while (trademgenService.isQueueDone() == false) {
00187
00188         // Get the next event from the event queue
00189         stdair::EventStruct lEventStruct;
00190         stdair::ProgressStatusSet lPPS = trademgenService.popEvent (lEventStruct);
00191
00192         // DEBUG
00193         STDAIR_LOG_DEBUG ("Popped event: '" << lEventStruct.describe() << "'");
00194
00195         // Extract the corresponding demand/booking request
00196         const stdair::BookingRequestStruct& lPoppedRequest =
00197             lEventStruct.getBookingRequest();
00198
00199         // DEBUG
00200         STDAIR_LOG_DEBUG ("Popped booking request: '"
00201             << lPoppedRequest.describe() << "'");
00202
00203         // Retrieve the corresponding demand stream
00204         const stdair::DemandGeneratorKey_T& lDemandStreamKey =
00205             lPoppedRequest.getDemandGeneratorKey();
00206
00207         // Check that the number of booking requests to be generated are correct
00208         const NbOfEventsByDemandStreamMap_T::iterator itNbOfEventsMap =
00209             lNbOfEventsMap.find (lDemandStreamKey);
00210         BOOST_REQUIRE_MESSAGE (itNbOfEventsMap != lNbOfEventsMap.end(),
00211             "The demand stream key '" << lDemandStreamKey
00212             << "' is not expected in that test");
00213
00214         const NbOfEventsPair_T& lNbOfEventsPair = itNbOfEventsMap->second;
00215         stdair::Count_T lCurrentNbOfEvents = lNbOfEventsPair.first;
00216         const stdair::Count_T& lExpectedTotalNbOfEvents = lNbOfEventsPair.second;

```

```

00239
00240 // Assess whether more events should be generated for that demand stream
00241 const bool stillHavingRequestsToBeGenerated = trademgenService.
00242     stillHavingRequestsToBeGenerated (lDemandStreamKey, lPPS,
00243         iDemandGenerationMethod);
00244
00251 if (lCurrentNbOfEvents == 1) {
00252     const stdair::ProgressStatus& lDemandStreamProgressStatus =
00253         lPPS.getSpecificGeneratorStatus();
00254     const stdair::Count_T& lNbOfRequests =
00255         lDemandStreamProgressStatus.getExpectedNb();
00261
00262     BOOST_CHECK_EQUAL (lNbOfRequests, lExpectedTotalNbOfEvents);
00263     BOOST_CHECK_MESSAGE (lNbOfRequests == lExpectedTotalNbOfEvents,
00264         "[" << lDemandStreamKey
00265         << "] Total number of requests to be generated: "
00266         << lNbOfRequests << "). Expected value: "
00267         << lExpectedTotalNbOfEvents);
00268 }
00269
00270 // DEBUG
00271 STDAIR_LOG_DEBUG ("=> [" << lDemandStreamKey << "]"[" << lCurrentNbOfEvents
00272     << "/" << lExpectedTotalNbOfEvents
00273     << "] is now processed. "
00274     << "Still generate events for that demand stream? "
00275     << stillHavingRequestsToBeGenerated);
00276
00277 // If there are still events to be generated for that demand stream,
00278 // generate and add them to the event queue
00279 if (stillHavingRequestsToBeGenerated == true) {
00280     const stdair::BookingRequestPtr_T lNextRequest_ptr =
00281         trademgenService.generateNextRequest (lDemandStreamKey,
00282             iDemandGenerationMethod);
00283     assert (lNextRequest_ptr != NULL);
00284
00290     const stdair::Duration_T lDuration =
00291         lNextRequest_ptr->getRequestDateTime()
00292         - lPoppedRequest.getRequestDateTime();
00293     BOOST_REQUIRE_GT (lDuration.total_milliseconds(), 0);
00294     BOOST_REQUIRE_MESSAGE (lDuration.total_milliseconds() > 0,
00295         "[" << lDemandStreamKey
00296         << "] The date-time of the generated event ("
00297         << lNextRequest_ptr->getRequestDateTime()
00298         << ") is lower than the date-time "
00299         << "of the current event ("
00300         << lPoppedRequest.getRequestDateTime() << ")");
00301
00302 // DEBUG
00303 STDAIR_LOG_DEBUG ("[" << lDemandStreamKey << "]"[" << lCurrentNbOfEvents
00304     << "/" << lExpectedTotalNbOfEvents
00305     << "] Added request: '" << lNextRequest_ptr->describe()
00306     << "'. Is queue done? "
00307     << trademgenService.isQueueDone());
00308
00309 // Keep, within the dedicated map, the current counters of events updated.
00310 ++lCurrentNbOfEvents;
00311 itNbOfEventsMap->second = NbOfEventsPair_T (lCurrentNbOfEvents,
00312     lExpectedTotalNbOfEvents);
00313 }
00314
00315 // Iterate
00316 ++idx;
00317 }
00318 // Compensate for the last iteration
00319 --idx;
00320
00321 if (iDemandGenerationMethod == stdair::DemandGenerationMethod::STA_ORD) {

```

```

00322     //
00323     BOOST_CHECK_EQUAL (idx, lRefActualNbOfEvents);
00324     BOOST_CHECK_MESSAGE (idx == lRefActualNbOfEvents,
00325                          "The total actual number of events is "
00326                          "<< lRefActualNbOfEvents << ", but " << idx
00327                          "<< " events have been generated");
00328 }
00329
00332 trademgenService.reset();
00333
00334 // DEBUG
00335 STDAIR_LOG_DEBUG ("End of the simulation");
00336
00337 // Close the log file
00338 logOutputFile.close();
00339
00340 }
00341
00342
00343 // ////////////////////////////////// Main: Unit Test Suite //////////////////////////////////
00344
00345 // Set the UTF configuration (re-direct the output to a specific file)
00346 BOOST_GLOBAL_FIXTURE (UnitTestFixture);
00347
00348 // Start the test suite
00349 BOOST_AUTO_TEST_SUITE (master_test_suite)
00350
00351
00354 BOOST_AUTO_TEST_CASE (trademgen_simple_simulation_test) {
00355
00356     // Input file name
00357     const stdair::Filename_T lInputFilename (STDAIR_SAMPLE_DIR "/demand01.csv");
00358
00359     // Generate the date time of the requests with the statistic order method.
00360     const stdair::DemandGenerationMethod lDemandGenerationMethod (stdair::DemandGen
erationMethod::STA_ORD);
00361
00362     // State whether the BOM tree should be built-in or parsed from an input file
00363     const bool isBuiltin = false;
00364     BOOST_CHECK_NO_THROW (testDemandGenerationHelper(0,
00365                                                     lInputFilename,
00366                                                     lDemandGenerationMethod,
00367                                                     isBuiltin));
00368 }
00369
00370
00374 BOOST_AUTO_TEST_CASE (trademgen_missing_input_file_test) {
00375
00376     // Input file name
00377     const stdair::Filename_T lInputFilename (STDAIR_SAMPLE_DIR "/missingFile.csv");
00378
00379     // Generate the date time of the requests with the statistic order method.
00380     const stdair::DemandGenerationMethod lDemandGenerationMethod (stdair::DemandGen
erationMethod::STA_ORD);
00381
00382     // State whether the BOM tree should be built-in or parsed from an input file
00383     const bool isBuiltin = false;
00384     BOOST_CHECK_THROW (testDemandGenerationHelper(1,
00385                                                     lInputFilename,
00386                                                     lDemandGenerationMethod,
00387                                                     isBuiltin),
TRADEMGEN::DemandInputFileNotFoundException);
00388
00389 }
00390
00391
00395 BOOST_AUTO_TEST_CASE (trademgen_default_bom_simulation_test) {

```



```
00396
00397 // Generate the date time of the requests with the statistic order method.
00398 const stdair::DemandGenerationMethod lDemandGenerationMethod (stdair::DemandGen
erationMethod::STA_ORD);
00399
00400 // State whether the BOM tree should be built-in or parsed from an input file
00401 const bool isBuiltin = true;
00402 BOOST_CHECK_NO_THROW (testDemandGenerationHelper(2,
00403 " " ,
00404 lDemandGenerationMethod,
00405 isBuiltin));
00406
00407 }
00408
00412 BOOST_AUTO_TEST_CASE (trademgen_poisson_process_test) {
00413
00414 // Generate the date time of the requests with the poisson process.
00415 const stdair::DemandGenerationMethod lDemandGenerationMethod (stdair::DemandGen
erationMethod::POI_PRO);
00416
00417 // State whether the BOM tree should be built-in or parsed from an input file
00418 const bool isBuiltin = true;
00419 BOOST_CHECK_NO_THROW (testDemandGenerationHelper(3,
00420 " " ,
00421 lDemandGenerationMethod,
00422 isBuiltin));
00423
00424 }
00425
00426 // End the test suite
00427 BOOST_AUTO_TEST_SUITE_END()
00428
00429
```

25.17 test/trademgen/DemandGenerationTestSuite.hpp File Reference

```
#include <iosfwd>
```

```
#include <cppunit/extensions/HelperMacros.h>
```

Classes

- class [DemandGenerationTestSuite](#)

Functions

- [CPPUNIT_TEST_SUITE_REGISTRATION](#) ([DemandGenerationTestSuite](#))

25.17.1 Function Documentation

25.17.1.1 CPPUNIT_TEST_SUITE_REGISTRATION ([DemandGenerationTestSuite](#))

25.18 DemandGenerationTestSuite.hpp

```
00001 // STL
00002 #include <iosfwd>
00003 // CPPUNIT
00004 #include <cppunit/extensions/HelperMacros.h>
00005
00006 class DemandGenerationTestSuite : public CppUnit::TestFixture {
00007     CPPUNIT_TEST_SUITE (DemandGenerationTestSuite);
00008     CPPUNIT_TEST (simpleEventGeneration);
00009     // CPPUNIT_TEST (errorCase);
00010     CPPUNIT_TEST_SUITE_END ();
00011 public:
00012
00013     void simpleEventGeneration();
00014
00015     // void errorCase ();
00016
00017     DemandGenerationTestSuite ();
00018
00019 private:
00020     void simpleEventGenerationHelper();
00021
00022 protected:
00023     std::stringstream _describeKey;
00024 };
00025
00026 CPPUNIT_TEST_SUITE_REGISTRATION (DemandGenerationTestSuite);
```

25.19 test/trademgen/generateEvents.cpp File Reference

```
#include <cassert>
#include <string>
#include <map>
#include <iostream>
#include <sstream>
#include <test/trademgen/EventStream.hpp>
#include <test/trademgen/CategoricalAttribute.hpp>
```

Functions

- int [main](#) (int argc, char *const argv[])

25.19.1 Function Documentation

25.19.1.1 int main (int *argc*, char *const *argv*[])

Definition at line [12](#) of file [generateEvents.cpp](#).

25.20 generateEvents.cpp

```
00001 // STL
00002 #include <cassert>
00003 #include <string>
00004 #include <map>
00005 #include <iostream>
00006 #include <sstream>
00007 // TraDemGen
00008 #include <test/trademgen/EventStream.hpp>
00009 #include <test/trademgen/CategoricalAttribute.hpp>
00010
00011 // ////////////////////////////////// M A I N //////////////////////////////////
00012 int main (int argc, char* const argv[]) {
00013     // input: seed, rate
00014     unsigned long int seed = 2;
00015
00016     if (argc >= 2) {
00017         std::istringstream iStream (argv[1]);
00018         iStream >> seed;
00019     }
00020
00021     // create event stream
00022     TRADEMGEN::EventStream e (seed);
00023     e.setKey("hello");
00024     e.setRate(2.0);
00025
00026     // get rate
00027     // const double r = e.getRate();
00028     std::cout << "Seed: " << seed << std::endl << std::endl;
00029
00030     // create instances
00031     for (int i=0; i<10; i++) {
00032         e.generateNext();
00033     }
00034
00035     // display events
00036     e.displayAllEvents(std::cout);
00037
00038
00039     // //////////////////////////////////
00040     // attributes
00041     std::map<int, float> M;
00042     M[1] = 0.1;
00043     M[17] = 0.7;
00044     M[77] = 0.2;
00045     TRADEMGEN::CategoricalAttribute C (M);
00046
00047     return 0;
00048 }
```

25.21 trademgen/basic/BasConst.cpp File Reference

```
#include <stdair/basic/BasConst_General.hpp>
#include <trademgen/basic/BasConst_TRADEMGEN_Service.hpp>
#include <trademgen/basic/BasConst_DemandGeneration.hpp>
```

Namespaces

- namespace [TRADEMGEN](#)

Functions

- stdair::BaseGenerator_T [TRADEMGEN::DEFAULT_BASE_GENERATOR](#) (stdair::DEFAULT_RANDOM_SEED)
- stdair::UniformGenerator_T [TRADEMGEN::DEFAULT_UNIFORM_GENERATOR](#) (DEFAULT_BASE_GENERATOR, DEFAULT_UNIFORM_REAL_DISTRIBUTION)

Variables

- const POSProbabilityMassFunction_T [TRADEMGEN::DEFAULT_POS_PROBALILITY_MASS](#)
- const stdair::FloatDuration_T [TRADEMGEN::DEFAULT_LAST_LOWER_BOUND_ARRIVAL_PATTERN](#) = -1
- const FRAT5Pattern_T [TRADEMGEN::DEFAULT_FRAT5_PATTERN](#) = Default-Map::createFRAT5Pattern()
- const double [TRADEMGEN::DEFAULT_MAX_ADVANCE_PURCHASE](#) = 330.0
- const stdair::UniformDistribution_T [TRADEMGEN::DEFAULT_UNIFORM_REAL_DISTRIBUTION](#)

25.22 BasConst.cpp

```

00001 // //////////////////////////////////////
00002 // Import section
00003 // //////////////////////////////////////
00004 // StdAir
00005 #include <stdair/basic/BasConst_General.hpp>
00006 // TraDemGen
00007 #include <trademgen/basic/BasConst_TRADEMGEN_Service.hpp>
00008 #include <trademgen/basic/BasConst_DemandGeneration.hpp>
00009
00010 namespace TRADEMGEN {
00011
00012     // const std::string DEFAULT_TRADEMGEN_SERVICE_NAME = "trademgen";
00013
00014     const POSProbabilityMassFunction_T DEFAULT_POS_PROBALILITY_MASS =
00015         DefaultMap::createPOSProbMass();
00016
00017     POSProbabilityMassFunction_T DefaultMap::createPOSProbMass() {
00018         POSProbabilityMassFunction_T oMap;
00019         // oMap["SIN"] = 0.44; oMap["HKG"] = 0.04; oMap["CGK"] = 0.04;
00020         // oMap["SYD"] = 0.04; oMap["BKK"] = 0.04; oMap["LHR"] = 0.03;
00021         // oMap["MEL"] = 0.03; oMap["KUL"] = 0.03; oMap["MNL"] = 0.03;
00022         // oMap["PVG"] = 0.03; oMap["PER"] = 0.02; oMap["BNE"] = 0.02;
00023         // oMap["NRT"] = 0.02; oMap["DPS"] = 0.02; oMap["SGN"] = 0.02;
00024         // oMap["PEN"] = 0.02; oMap["FRA"] = 0.02; oMap["PEK"] = 0.02;
00025         // oMap["HKT"] = 0.02; oMap["AKT"] = 0.02; oMap["SFO"] = 0.01;
00026         // oMap["ICN"] = 0.01; oMap["TPE"] = 0.01; oMap["row"] = 0.02;
00027         oMap["row"] = 1.0;
00028         return oMap;
00029     }
00030
00031     const stdair::FloatDuration_T DEFAULT_LAST_LOWER_BOUND_ARRIVAL_PATTERN = -1;
00032
00033     const FRAT5Pattern_T DEFAULT_FRAT5_PATTERN = DefaultMap::createFRAT5Pattern();
00034
00035     FRAT5Pattern_T DefaultMap::createFRAT5Pattern() {
00036         FRAT5Pattern_T oMap;
00037         // oMap[1.10] = 0.0; oMap[1.40] = 0.80909; oMap[1.45] = 0.8303;
00038         // oMap[1.50] = 0.85152; oMap[1.55] = 0.87273; oMap[1.60] = 0.89394;
00039         // oMap[1.70] = 0.90606; oMap[1.80] = 0.91818; oMap[2.00] = 0.9303;
00040         // oMap[2.30] = 0.94242; oMap[2.60] = 0.95152; oMap[3.00] = 0.96061;
00041         // oMap[3.30] = 0.96970; oMap[3.40] = 0.97879; oMap[3.44] = 0.98485;
00042         // oMap[3.47] = 0.99091; oMap[3.50] = 0.99697; oMap[3.500000001] = 1.0;
00043         // oMap[1.10] = -365; oMap[1.40] = -63; oMap[1.45] = -56;
00044         // oMap[1.50] = -49; oMap[1.55] = -42; oMap[1.60] = -35;
00045         // oMap[1.70] = -31; oMap[1.80] = -27; oMap[2.00] = -23;
00046         // oMap[2.30] = -19; oMap[2.60] = -16; oMap[3.00] = -13;
00047         // oMap[3.30] = -10; oMap[3.40] = -7; oMap[3.44] = -5;
00048         // oMap[3.47] = -3; oMap[3.50] = -1; oMap[3.500000001] = 0;
00049         // oMap[1.0] = -365; oMap[1.10] = -63; oMap[1.13] = -56;
00050         // oMap[1.17] = -49; oMap[1.22] = -42; oMap[1.28] = -35;
00051         // oMap[1.32] = -31; oMap[1.37] = -27; oMap[1.43] = -23;
00052         // oMap[1.51] = -19; oMap[1.60] = -16; oMap[1.70] = -13;
00053         // oMap[1.80] = -10; oMap[1.90] = -7; oMap[1.93] = -5;
00054         // oMap[1.96] = -3; oMap[2.00] = -1; oMap[2.000000001] = 0;
00055         // oMap[1.0] = -365; oMap[1.05] = -63; oMap[1.07] = -56;
00056         // oMap[1.09] = -49; oMap[1.11] = -42; oMap[1.14] = -35;
00057         // oMap[1.16] = -31; oMap[1.18] = -27; oMap[1.21] = -23;
00058         // oMap[1.24] = -19; oMap[1.27] = -16; oMap[1.30] = -13;
00059         // oMap[1.33] = -10; oMap[1.37] = -7; oMap[1.40] = -5;
00060         // oMap[1.45] = -3; oMap[1.50] = -1; oMap[1.500000001] = 0;
00061         oMap[1.10] = -365; oMap[1.20] = -63;
00062         oMap[1.30] = -49; oMap[1.40] = -35; oMap[1.70] = -23;
00063         oMap[2.00] = -16; oMap[2.30] = -10; oMap[2.44] = -5;
00064         oMap[2.50] = -1; oMap[2.500000001] = 0;
00065         return oMap;
00066     }
00067

```

```
00072     }
00073
00074     00075     const double DEFAULT_MAX_ADVANCE_PURCHASE = 330.0;
00076
00077     stdair::BaseGenerator_T DEFAULT_BASE_GENERATOR (stdair::DEFAULT_RANDOM_SEED);
00078
00079     00081     const stdair::UniformDistribution_T DEFAULT_UNIFORM_REAL_DISTRIBUTION;
00082
00083     stdair::UniformGenerator_T
00084     00085     DEFAULT_UNIFORM_GENERATOR (DEFAULT_BASE_GENERATOR,
00086                                DEFAULT_UNIFORM_REAL_DISTRIBUTION);
00087
00088 }
```


25.23 trademgen/basic/BasConst_DemandGeneration.hpp File Reference

```
#include <string>
#include <stdair/stdair_maths_types.hpp>
#include <stdair/stdair_date_time_types.hpp>
#include <trademgen/basic/DemandCharacteristicsTypes.hpp>
```

Classes

- struct [TRADEMGEN::DefaultMap](#)

Namespaces

- namespace [TRADEMGEN](#)

Variables

- stdair::BaseGenerator_T [TRADEMGEN::DEFAULT_BASE_GENERATOR](#)
- stdair::UniformGenerator_T [TRADEMGEN::DEFAULT_UNIFORM_GENERATOR](#)

25.24 BasConst_DemandGeneration.hpp

```

00001 #ifndef __TRADEMGEN_BAS_BASCONST_DEMANDGENERATION_HPP
00002 #define __TRADEMGEN_BAS_BASCONST_DEMANDGENERATION_HPP
00003
00004 // //////////////////////////////////////
00005 // Import section
00006 // //////////////////////////////////////
00007 // STL
00008 #include <string>
00009 // StdAir
00010 #include <stdair/stdair_maths_types.hpp>
00011 #include <stdair/stdair_date_time_types.hpp>
00012 // TraDemGen
00013 #include <trademgen/basic/DemandCharacteristicsTypes.hpp>
00014
00015 namespace TRADEMGEN {
00016
00018     extern const POSProbabilityMassFunction_T DEFAULT_POS_PROBALILITY_MASS;
00019
00021     extern const FRAT5Pattern_T DEFAULT_FRAT5_PATTERN;
00022
00024     struct DefaultMap {
00025         static POSProbabilityMassFunction_T createPOSProbMass();
00026         static FRAT5Pattern_T createFRAT5Pattern();
00027     };
00028
00030     extern const stdair::FloatDuration_T DEFAULT_LAST_LOWER_BOUND_ARRIVAL_PATTERN;
00031
00033     extern const double DEFAULT_MAX_ADVANCE_PURCHASE;
00034
00039     extern stdair::BaseGenerator_T DEFAULT_BASE_GENERATOR;
00040
00045     extern stdair::UniformGenerator_T DEFAULT_UNIFORM_GENERATOR;
00046
00048     extern const stdair::UniformDistribution_T DEFAULT_UNIFORM_REAL_DISTRIBUTION;
00049
00050 }
00051 #endif // __TRADEMGEN_BAS_BASCONST_DEMANDGENERATION_HPP

```

25.25 trademgen/basic/BasConst_TRADEMGEN_Service.hpp File Reference

```
#include <string>
```

Namespaces

- namespace [TRADEMGEN](#)

25.26 BasConst_TRADEMGEN_Service.hpp

```
00001 #ifndef __TRADEMGEN_BAS_BASCONST_TRADEMGEN_SERVICE_HPP
00002 #define __TRADEMGEN_BAS_BASCONST_TRADEMGEN_SERVICE_HPP
00003
00004 // //////////////////////////////////////
00005 // Import section
00006 // //////////////////////////////////////
00007 #include <string>
00008
00009 namespace TRADEMGEN {
00010
00012     // extern const std::string DEFAULT_TRADEMGEN_SERVICE_NAME;
00013
00014 }
00015 #endif // __TRADEMGEN_BAS_BASCONST_TRADEMGEN_SERVICE_HPP
```

25.27 trademgen/basic/BasParserTypes.hpp File Reference

```
#include <string>
#include <boost/spirit/home/classic/core.hpp>
#include <boost/spirit/home/classic/utility/loops.hpp>
#include <boost/spirit/home/classic/utility/chset.hpp>
#include <boost/spirit/home/classic/utility/config.hpp>
#include <boost/spirit/home/classic/iterator/file_iterator.hpp>
```

Namespaces

- namespace [TRADEMGEN](#)

Typedefs

- typedef char [TRADEMGEN::char_t](#)
- typedef boost::spirit::classic::file_iterator< char_t > [TRADEMGEN::iterator_t](#)
- typedef boost::spirit::classic::scanner< iterator_t > [TRADEMGEN::scanner_t](#)
- typedef boost::spirit::classic::rule< scanner_t > [TRADEMGEN::rule_t](#)
- typedef boost::spirit::classic::int_parser< unsigned int, 10, 1, 1 > [TRADEMGEN::int1_p_t](#)
- typedef boost::spirit::classic::uint_parser< unsigned int, 10, 2, 2 > [TRADEMGEN::uint2_p_t](#)
- typedef boost::spirit::classic::uint_parser< unsigned int, 10, 1, 2 > [TRADEMGEN::uint1_2_p_t](#)
- typedef boost::spirit::classic::uint_parser< unsigned int, 10, 1, 3 > [TRADEMGEN::uint1_3_p_t](#)
- typedef boost::spirit::classic::uint_parser< unsigned int, 10, 4, 4 > [TRADEMGEN::uint4_p_t](#)
- typedef boost::spirit::classic::uint_parser< unsigned int, 10, 1, 4 > [TRADEMGEN::uint1_4_p_t](#)
- typedef boost::spirit::classic::chset< char_t > [TRADEMGEN::chset_t](#)
- typedef boost::spirit::classic::impl::loop_traits< chset_t, unsigned int, unsigned int >::type [TRADEMGEN::repeat_p_t](#)
- typedef boost::spirit::classic::bounded< uint2_p_t, unsigned int > [TRADEMGEN::bounded2_p_t](#)
- typedef boost::spirit::classic::bounded< uint1_2_p_t, unsigned int > [TRADEMGEN::bounded1_2_p_t](#)
- typedef boost::spirit::classic::bounded< uint1_3_p_t, unsigned int > [TRADEMGEN::bounded1_3_p_t](#)
- typedef boost::spirit::classic::bounded< uint4_p_t, unsigned int > [TRADEMGEN::bounded4_p_t](#)
- typedef boost::spirit::classic::bounded< uint1_4_p_t, unsigned int > [TRADEMGEN::bounded1_4_p_t](#)

25.28 BasParserTypes.hpp

```

00001 #ifndef __TRADEMGEN_BAS_BASCOMPARSERTYPES_HPP
00002 #define __TRADEMGEN_BAS_BASCOMPARSERTYPES_HPP
00003
00004 // //////////////////////////////////////
00005 // Import section
00006 // //////////////////////////////////////
00007 // STL
00008 #include <string>
00009 // Boost
00010 // #define BOOST_SPIRIT_DEBUG
00011 #include <boost/spirit/home/classic/core.hpp>
00012 // #include <boost/spirit/home/classic/attribute.hpp>
00013 // #include <boost/spirit/home/classic/utility/functor_parser.hpp>
00014 #include <boost/spirit/home/classic/utility/loops.hpp>
00015 #include <boost/spirit/home/classic/utility/chset.hpp>
00016 #include <boost/spirit/home/classic/utility/confix.hpp>
00017 #include <boost/spirit/home/classic/iterator/file_iterator.hpp>
00018 // #include <boost/spirit/home/classic/actor/push_back_actor.hpp>
00019 // #include <boost/spirit/home/classic/actor/assign_actor.hpp>
00020
00021 namespace TRADEMGEN {
00022
00023     // //////////////////////////////////////
00024     //
00025     // Definition of Basic Types
00026     //
00027     // //////////////////////////////////////
00028     // For a file, the parsing unit is the character (char). For a string,
00029     // it is a "char const *".
00030     // typedef char const* iterator_t;
00031     typedef char char_t;
00032
00033     // The types of iterator, scanner and rule are then derived from
00034     // the parsing unit.
00035     typedef boost::spirit::classic::file_iterator<char_t> iterator_t;
00036     typedef boost::spirit::classic::scanner<iterator_t> scanner_t;
00037     typedef boost::spirit::classic::rule<scanner_t> rule_t;
00038
00039     // //////////////////////////////////////
00040     //
00041     // Parser related types
00042     //
00043     // //////////////////////////////////////
00044     typedef boost::spirit::classic::int_parser<unsigned int, 10, 1, 1> int1_p_t;
00045
00046     typedef boost::spirit::classic::uint_parser<unsigned int, 10, 2, 2> uint2_p_t;
00047
00048     typedef boost::spirit::classic::uint_parser<unsigned int, 10, 1, 2>
00049     uint1_2_p_t;
00050
00051     typedef boost::spirit::classic::uint_parser<unsigned int, 10, 1, 3>
00052     uint1_3_p_t;
00053
00054     typedef boost::spirit::classic::uint_parser<unsigned int, 10, 4, 4> uint4_p_t;
00055
00056     typedef boost::spirit::classic::uint_parser<unsigned int, 10, 1, 4>
00057     uint1_4_p_t;
00058
00059     typedef boost::spirit::classic::chset<char_t> chset_t;
00060
00061     typedef boost::spirit::classic::impl::loop_traits<chset_t,
00062     unsigned int,
00063     unsigned int>::type repeat_p_t;
00064
00065     typedef boost::spirit::classic::bounded<uint2_p_t, unsigned int> bounded2_p_t;

```

```
00073     typedef boost::spirit::classic::bounded<uint1_2_p_t, unsigned int>
        bounded1_2_p_t;
00074     typedef boost::spirit::classic::bounded<uint1_3_p_t, unsigned int>
        bounded1_3_p_t;
00075     typedef boost::spirit::classic::bounded<uint4_p_t, unsigned int> bounded4_p_t;
00076     typedef boost::spirit::classic::bounded<uint1_4_p_t, unsigned int>
        bounded1_4_p_t;
00077 }
00078 #endif // __TRADEMGEN_BAS_BASCOMPARSERTYPES_HPP
```

25.29 trademgen/basic/CategoricalAttribute.hpp File Reference

```
#include <map>
#include <iosfwd>
#include <stdair/STDAIR_Types.hpp>
#include <stdair/basic/DictionaryManager.hpp>
```

Classes

- struct [stdair::CategoricalAttribute< T >](#)

Class modeling the distribution of values that can be taken by a categorical attribute.

Namespaces

- namespace [stdair](#)

Forward declarations.

25.30 CategoricalAttribute.hpp

```

00001 #ifndef __STDAIR_BAS_CATEGORICALATTRIBUTE_HPP
00002 #define __STDAIR_BAS_CATEGORICALATTRIBUTE_HPP
00003
00004 // //////////////////////////////////////
00005 // Import section
00006 // //////////////////////////////////////
00007 // STL
00008 #include <map>
00009 #include <iosfwd>
00010 // STDAIR
00011 #include <stdair/STDAIR_Types.hpp>
00012 #include <stdair/basic/DictionaryManager.hpp>
00013
00014 namespace stdair {
00015
00020     template <typename T>
00021     struct CategoricalAttribute {
00022
00023     public:
00024         // ////////////////////////////////// Type definitions //////////////////////////////////
00028         typedef std::map<T, DictionaryKey_T> ProbabilityMassFunction_T;
00029
00033         typedef std::map<DictionaryKey_T, T> InverseCumulativeDistribution_T;
00034
00035     private:
00037         // ////////////////////////////////// Getters //////////////////////////////////
00041         const ProbabilityMassFunction_T& getProbabilityMassFunction() const {
00042             return _probabilityMassFunction;
00043         }
00044
00048         const InverseCumulativeDistribution_T& getInverseCumulativeDistribution() con
00049 st {
00049             return _inverseCumulativeDistribution;
00050         }
00051
00052         // ////////////////////////////////// Setters //////////////////////////////////
00056         void setProbabilityMassFunction (const ProbabilityMassFunction_T& iProbabilit
00057 yMassFunction) {
00057             _probabilityMassFunction = iProbabilityMassFunction;
00058             determineInverseCumulativeDistributionFromProbabilityMassFunction();
00059         }
00060
00061     public:
00063         // ////////////////////////////////// Business Methods //////////////////////////////////
00067         const T& getValue (const Probability_T& iCumulativeProbability) const {
00068
00069             const DictionaryKey_T& lKey =
00070                 DictionaryManager::valueToKey (iCumulativeProbability);
00071
00072             InverseCumulativeDistribution_T::const_iterator itT =
00073                 _inverseCumulativeDistribution.find (lKey);
00074
00075             if (itT == _inverseCumulativeDistribution.end()) {
00076                 std::ostringstream oStr;
00077                 oStr << "The following cumulative probability is out of range: "
00078                     << iCumulativeProbability << displayInverseCumulativeDistribution();
00079
00079                 throw IndexOutOfRangeException (oStr.str());
00080             }
00081
00082             return itT->second;
00083         }
00084

```

```

00085
00086 public:
00087 // //////////// Display Support Methods ////////////
00091 const std::string displayProbabilityMassFunction() const {
00092     std::ostringstream ostr;
00093     unsigned int idx = 0;
00094
00095     for (typename ProbabilityMassFunction_T::const_iterator it =
00096         _probabilityMassFunction.begin();
00097         it != _probabilityMassFunction.end(); ++it, ++idx) {
00098         if (idx != 0) {
00099             ostr << ", ";
00100         }
00101         ostr << it->first << ":"
00102             << DictionaryManager::keyToValue (it->second);
00103     }
00104
00105     return ostr.str();
00106 }
00107
00111 const std::string displayInverseCumulativeDistribution() const {
00112     std::ostringstream ostr;
00113
00114     for (typename InverseCumulativeDistribution_T::const_iterator it =
00115         _inverseCumulativeDistribution.begin();
00116         it != _inverseCumulativeDistribution.end(); ++it) {
00117         ostr << "cumulative prob: " << DictionaryManager::keyToValue (it->first)
00118             << " value: " << it->second << std::endl;
00119     }
00120
00121     return ostr.str();
00122 }
00123
00124 public:
00125 // //////////// Constructors and destructors ////////////
00129 CategoricalAttribute (const ProbabilityMassFunction_T& iProbabilityMassFunction)
00130 on)
00131 : _probabilityMassFunction (iProbabilityMassFunction) {
00132     determineInverseCumulativeDistributionFromProbabilityMassFunction();
00133 }
00137 CategoricalAttribute() { }
00138
00142 CategoricalAttribute (const CategoricalAttribute& iCategoricalAttribute)
00143 : _probabilityMassFunction (iCategoricalAttribute._probabilityMassFunction)
00144 {
00145     determineInverseCumulativeDistributionFromProbabilityMassFunction();
00146 }
00150 virtual ~CategoricalAttribute() { }
00151
00152
00157 void determineInverseCumulativeDistributionFromProbabilityMassFunction() {
00158
00159     Probability_T cumulative_probability_so_far = 0.0;
00160     for (typename ProbabilityMassFunction_T::const_iterator
00161         itProbabilityMassFunction = _probabilityMassFunction.begin();
00162         itProbabilityMassFunction != _probabilityMassFunction.end();
00163         ++itProbabilityMassFunction) {
00164
00165         Probability_T attribute_probability_mass =
00166             DictionaryManager::keyToValue (itProbabilityMassFunction->second);
00167
00168         if (attribute_probability_mass > 0) {
00169             T attribute_value = itProbabilityMassFunction->first;
00170             cumulative_probability_so_far += attribute_probability_mass;
00171

```

```
00172         const DictionaryKey_T& lKey =
00173             DictionaryManager::valueToKey (cumulative_probability_so_far);
00174
00175         //_inverseCumulativeDistribution[lKey] = attribute_value;
00176         _inverseCumulativeDistribution.
00177             insert (typename InverseCumulativeDistribution_T::
00178                 value_type (lKey, attribute_value));
00179     }
00180 }
00181 }
00182
00183 private:
00184     // ////////// Attributes //////////
00188     ProbabilityMassFunction_T _probabilityMassFunction;
00189
00193     InverseCumulativeDistribution_T _inverseCumulativeDistribution;
00194 };
00195 }
00196 #endif // __STDAIR_BAS_CATEGORICALATTRIBUTE_HPP
```

25.31 trademgen/basic/CategoricalAttributeLite.hpp File Reference

```
#include <cassert>
#include <sstream>
#include <string>
#include <vector>
#include <map>
#include <stdair/stdair_basic_types.hpp>
#include <stdair/service/Logger.hpp>
#include <trademgen/TRADEMGEN_Exceptions.hpp>
#include <trademgen/basic/DictionaryManager.hpp>
```

Classes

- struct [TRADEMGEN::CategoricalAttributeLite< T >](#)
Class modeling the distribution of values that can be taken by a categorical attribute.

Namespaces

- namespace [TRADEMGEN](#)

25.32 CategoricalAttributeLite.hpp

```

00001 #ifndef __TRADEMGEN_BAS_CATEGORICALATTRIBUTELITE_HPP
00002 #define __TRADEMGEN_BAS_CATEGORICALATTRIBUTELITE_HPP
00003
00004 // //////////////////////////////////////
00005 // Import section
00006 // //////////////////////////////////////
00007 // STL
00008 #include <cassert>
00009 #include <sstream>
00010 #include <string>
00011 #include <vector>
00012 #include <map>
00013 // StdAir
00014 #include <stdair/stdair_basic_types.hpp>
00015 #include <stdair/service/Logger.hpp>
00016 // TraDemGen
00017 #include <trademgen/TRADEMGEN_Exceptions.hpp>
00018 #include <trademgen/basic/DictionaryManager.hpp>
00019
00020 namespace TRADEMGEN {
00021
00022     template <typename T>
00023     struct CategoricalAttributeLite {
00024     public:
00025         // ////////////////////////////////// Type definitions //////////////////////////////////
00026         typedef std::map<T, stdair::Probability_T> ProbabilityMassFunction_T;
00027
00028     public:
00029         // ////////////////////////////////// Business Methods //////////////////////////////////
00030         const T& getValue (const stdair::Probability_T& iCumulativeProbability) const
00031         {
00032             const DictionaryKey_T& lKey =
00033                 DictionaryManager::valueToKey (iCumulativeProbability);
00034
00035             for (unsigned int idx = 0; idx < _size; ++idx) {
00036                 if (_cumulativeDistribution.at(idx) >= lKey) {
00037                     const T& oValue = _valueArray.at(idx);
00038                     return oValue;
00039                 }
00040             }
00041
00042             std::ostringstream oStr;
00043             oStr << "The following cumulative probability is out of range: "
00044                 << iCumulativeProbability << displayProbabilityMass();
00045             throw IndexOutOfRangeException (oStr.str());
00046         }
00047
00048         bool checkValue (const T& iValue) const {
00049             for (unsigned int idx = 0; idx < _size; ++idx) {
00050                 if (_valueArray.at(idx) == iValue) {
00051                     return true;
00052                 }
00053             }
00054             return false;
00055         }
00056     public:
00057         // ////////////////////////////////// Display Support Methods //////////////////////////////////
00058         const std::string displayProbabilityMass() const {
00059             std::ostringstream oStr;
00060
00061             for (unsigned int idx = 0; idx < _size; ++idx) {
00062                 if (idx != 0) {

```

```

00081         ostr << ", ";
00082     }
00083     ostr << _valueArray.at(idx) << ":"
00084         << DictionaryManager::keyToValue (_cumulativeDistribution[idx]);
00085 }
00086 return ostr.str();
00087 }
00088
00089
00090 public:
00091     // ////////// Constructors and destructors //////////
00092     CategoricalAttributeLite (const ProbabilityMassFunction_T& iValueMap)
00093         : _size (iValueMap.size()) {
00094         init (iValueMap);
00095     }
00096
00097     CategoricalAttributeLite() : _size(1) {
00098     }
00099
00100     CategoricalAttributeLite (const CategoricalAttributeLite& iCAL)
00101         : _size (iCAL._size),
00102           _cumulativeDistribution (iCAL._cumulativeDistribution),
00103           _valueArray (iCAL._valueArray) {
00104     }
00105
00106     CategoricalAttributeLite& operator= (const CategoricalAttributeLite& iCAL) {
00107         _size = iCAL._size;
00108         _cumulativeDistribution = iCAL._cumulativeDistribution;
00109         _valueArray = iCAL._valueArray;
00110         return *this;
00111     }
00112
00113     virtual ~CategoricalAttributeLite() {
00114     }
00115
00116 private:
00117     void init (const ProbabilityMassFunction_T& iValueMap) {
00118
00119         const unsigned int lSize = iValueMap.size();
00120         _cumulativeDistribution.reserve (lSize);
00121         _valueArray.reserve (lSize);
00122
00123         stdair::Probability_T cumulative_probability_so_far = 0.0;
00124
00125         // Browse the map to retrieve the values and to build the
00126         // cumulative probabilities.
00127         for (typename ProbabilityMassFunction_T::const_iterator
00128             itProbabilityMassFunction = iValueMap.begin();
00129             itProbabilityMassFunction != iValueMap.end();
00130             ++itProbabilityMassFunction) {
00131
00132             stdair::Probability_T attribute_probability_mass =
00133                 itProbabilityMassFunction->second;
00134
00135             if (attribute_probability_mass > 0) {
00136                 const T& attribute_value = itProbabilityMassFunction->first;
00137                 cumulative_probability_so_far += attribute_probability_mass;
00138
00139                 const DictionaryKey_T& lKey =
00140                     DictionaryManager::valueToKey (cumulative_probability_so_far);
00141
00142                 // Build the two arrays.
00143                 _cumulativeDistribution.push_back (lKey);
00144                 _valueArray.push_back (attribute_value);
00145             }
00146         }
00147     }

```

```
00166         // Remember the actual array size.
00167         _size = _valueArray.size();
00168     }
00169
00170     private:
00171         // ////////// Attributes //////////
00172         unsigned int _size;
00173
00174         std::vector<DictionaryKey_T> _cumulativeDistribution;
00175
00176         std::vector<T> _valueArray;
00177     };
00178 }
00179 #endif // __TRADEMGEN_BAS_CATEGORICALATTRIBUTE_LITE_HPP
```

25.33 trademgen/basic/ContinuousAttribute.hpp File Reference

```
#include <string>
#include <map>
#include <stdair/stdair_date_time_types.hpp>
#include <stdair/service/Logger.hpp>
#include <trademgen/basic/DictionaryManager.hpp>
```

Classes

- struct [TRADEMGEN::ContinuousAttribute< T >](#)

Namespaces

- namespace [TRADEMGEN](#)

25.34 ContinuousAttribute.hpp

```

00001 #ifndef __TRADEMGEN_BAS_CONTINUOUSATTRIBUTE_HPP
00002 #define __TRADEMGEN_BAS_CONTINUOUSATTRIBUTE_HPP
00003
00004 // //////////////////////////////////////
00005 // Import section
00006 // //////////////////////////////////////
00007 // STL
00008 #include <string>
00009 #include <map>
00010 // StdAir
00011 #include <stdair/stdair_date_time_types.hpp>
00012 #include <stdair/service/Logger.hpp>
00013 // TraDemGen
00014 #include <trademgen/basic/DictionaryManager.hpp>
00015
00016 namespace TRADEMGEN {
00017
00020     template <class T>
00021     struct ContinuousAttribute {
00022     public:
00023
00024         // ////////////////////////////////// Type definitions //////////////////////////////////
00026         typedef std::multimap<T, DictionaryKey_T> ContinuousDistribution_T;
00027         typedef std::multimap<DictionaryKey_T, T> ContinuousInverseDistribution_T;
00028
00029     private:
00030         // ////////////////////////////////// Getters //////////////////////////////////
00032         const ContinuousDistribution_T& getCumulativeDistribution() const {
00033             return _cumulativeDistribution;
00034         }
00035
00037         const ContinuousInverseDistribution_T& getInverseCumulativeDistribution () const {
00038             return _inverseCumulativeDistribution;
00039         }
00040
00041     private:
00042         // ////////////////////////////////// Setters //////////////////////////////////
00044         void setCumulativeDistribution (const ContinuousDistribution_T& iCumulativeDi
00045         stribution) {
00046             _cumulativeDistribution = iCumulativeDistribution;
00047             determineInverseCumulativeDistributionFromCumulativeDistribution();
00048         }
00049     public:
00050         // ////////////////////////////////// Business Methods //////////////////////////////////
00052         const T getValue (const stdair::Probability_T& iCumulativeProbability) const{
00053
00054             const DictionaryKey_T lKey =
00055                 DictionaryManager::valueToKey (iCumulativeProbability);
00056             typename ContinuousInverseDistribution_T::const_iterator it =
00057                 _inverseCumulativeDistribution.lower_bound (lKey);
00058
00059             stdair::Probability_T cumulativeProbabilityNextPoint =
00060                 DictionaryManager::keyToValue (it->first);
00061             T valueNextPoint = it->second;
00062
00063             if (it == _inverseCumulativeDistribution.begin()) {
00064                 STDAIR_LOG_DEBUG ("Last element");
00065                 return valueNextPoint;
00066             }
00067             --it;
00068
00069             stdair::Probability_T cumulativeProbabilityPreviousPoint =
00070                 DictionaryManager::keyToValue (it->first);

```

```

00070     T valuePreviousPoint = it->second;
00071     if (cumulativeProbabilityNextPoint == cumulativeProbabilityPreviousPoint) {

00072         return valuePreviousPoint;
00073     }
00074
00075     return valuePreviousPoint + (valueNextPoint - valuePreviousPoint)
00076         * (iCumulativeProbability - cumulativeProbabilityPreviousPoint)
00077         / (cumulativeProbabilityNextPoint - cumulativeProbabilityPreviousPoint);
00078 }
00079
00080 public:
00081 // //////////// Display Support Methods ////////////
00082 const std::string displayCumulativeDistribution() const {
00083     std::ostringstream oStr;
00084     unsigned int idx = 0;
00085     for (typename ContinuousDistribution_T::const_iterator it =
00086         _cumulativeDistribution.begin();
00087         it != _cumulativeDistribution.end(); ++it, ++idx) {
00088         if (idx != 0) {
00089             oStr << ", ";
00090         }
00091         oStr << it->first << ":"
00092             << DictionaryManager::keyToValue (it->second);
00093     }
00094     return oStr.str();
00095 }
00096
00097 const std::string displayInverseCumulativeDistribution() const {
00098     std::ostringstream oStr;
00099     for (typename ContinuousInverseDistribution_T::const_iterator it =
00100         _inverseCumulativeDistribution.begin();
00101         it != _inverseCumulativeDistribution.end(); ++it) {
00102         oStr << "cumulative prob: " << DictionaryManager::keyToValue (it->first)
00103             << " value: " << it->second << std::endl;
00104     }
00105     return oStr.str();
00106 }
00107
00108 public:
00109 // //////////// Constructors and destructors ////////////
00110 ContinuousAttribute () { }
00111
00112 ContinuousAttribute (const ContinuousDistribution_T& iCumulativeDistribution)
00113     : _cumulativeDistribution (iCumulativeDistribution) {
00114     determineInverseCumulativeDistributionFromCumulativeDistribution();
00115 }
00116
00117 ContinuousAttribute (const ContinuousAttribute& iContinuousAttribute)
00118     : _cumulativeDistribution (iContinuousAttribute._cumulativeDistribution),
00119       _inverseCumulativeDistribution (iContinuousAttribute._inverseCumulativeDi
00120 stribution) {
00121 }
00122
00123 virtual ~ContinuousAttribute () { }
00124
00125 void determineInverseCumulativeDistributionFromCumulativeDistribution () {
00126     for (typename ContinuousDistribution_T::iterator itCumulativeDistribution =
00127         _cumulativeDistribution.begin();
00128         itCumulativeDistribution != _cumulativeDistribution.end();
00129         ++itCumulativeDistribution) {
00130         _inverseCumulativeDistribution.
00131             insert (typename ContinuousInverseDistribution_T::
00132                 value_type (itCumulativeDistribution->second,
00133                     itCumulativeDistribution->first));
00134     }
00135 }

```

```
00141     }
00142 }
00143
00144 private:
00145     // ////////// Attributes //////////
00146
00147     ContinuousDistribution_T _cumulativeDistribution;
00148
00149     ContinuousInverseDistribution_T _inverseCumulativeDistribution;
00150 };
00151
00152 }
00153
00154 }
00155 #endif // __STDAIR_BAS_CONTINUOUSATTRIBUTE_HPP
```

25.35 trademgen/basic/ContinuousAttributeLite.hpp File Reference

```
#include <cassert>
#include <iosfwd>
#include <string>
#include <vector>
#include <map>
#include <stdair/stdair_basic_types.hpp>
#include <trademgen/TRADEMGEN_Exceptions.hpp>
#include <trademgen/basic/DictionaryManager.hpp>
```

Classes

- struct [TRADEMGEN::ContinuousAttributeLite< T >](#)
Class modeling the distribution of values that can be taken by a continuous attribute.

Namespaces

- namespace [TRADEMGEN](#)

25.36 ContinuousAttributeLite.hpp

```

00001 #ifndef __TRADEMGEN_BAS_CONTINUOUSATTRIBUTE_LITE_HPP
00002 #define __TRADEMGEN_BAS_CONTINUOUSATTRIBUTE_LITE_HPP
00003
00004 // //////////////////////////////////////
00005 // Import section
00006 // //////////////////////////////////////
00007 // STL
00008 #include <cassert>
00009 #include <iosfwd>
00010 #include <string>
00011 #include <vector>
00012 #include <map>
00013 // StdAir
00014 #include <stdair/stdair_basic_types.hpp>
00015 // TraDemGen
00016 #include <trademgen/TRADEMGEN_Exceptions.hpp>
00017 #include <trademgen/basic/DictionaryManager.hpp>
00018
00019 namespace TRADEMGEN {
00020
00021     template <typename T>
00022     struct ContinuousAttributeLite {
00023     public:
00024         // ////////////////////////////////// Type definitions //////////////////////////////////
00025         typedef std::map<T, stdair::Probability_T> ContinuousDistribution_T;
00026
00027     public:
00028         // ////////////////////////////////// Business Methods //////////////////////////////////
00029         const T getValue(const stdair::Probability_T& iCumulativeProbability) const {
00030             const DictionaryKey_T& lKey =
00031                 DictionaryManager::valueToKey (iCumulativeProbability);
00032
00033             // Find the first cumulative probability value greater or equal to lKey.
00034             unsigned int idx = 0;
00035             for (; idx < _size; ++idx) {
00036                 if (_cumulativeDistribution.at(idx) > lKey) {
00037                     break;
00038                 }
00039             }
00040
00041             if (idx == 0) {
00042                 return _valueArray.at(idx);
00043             }
00044             if (idx == _size) {
00045                 return _valueArray.at(idx-1);
00046             }
00047
00048             //
00049             const stdair::Probability_T& lCumulativeCurrentPoint =
00050                 DictionaryManager::keyToValue (_cumulativeDistribution.at(idx));
00051             const T& lValueCurrentPoint = _valueArray.at(idx);
00052
00053             //
00054             const stdair::Probability_T& lCumulativePreviousPoint =
00055                 DictionaryManager::keyToValue (_cumulativeDistribution.at(idx-1));
00056             const T& lValuePreviousPoint = _valueArray.at(idx-1);
00057
00058             if (lCumulativePreviousPoint == lCumulativeCurrentPoint) {
00059                 return lValuePreviousPoint;
00060             }
00061
00062             T oValue= lValuePreviousPoint + (lValueCurrentPoint - lValuePreviousPoint)
00063                 * (iCumulativeProbability - lCumulativePreviousPoint)
00064                 / (lCumulativeCurrentPoint - lCumulativePreviousPoint);
00065
00066         }
00067     };
00068
00069     T oValue= lValuePreviousPoint + (lValueCurrentPoint - lValuePreviousPoint)
00070         * (iCumulativeProbability - lCumulativePreviousPoint)
00071         / (lCumulativeCurrentPoint - lCumulativePreviousPoint);
00072
00073 }
00074
00075

```

```

00076         return oValue;
00077     }
00078
00082     const double getDerivativeValue(const T iKey) const{
00083
00084         // Find the first key value greater or equal to iKey.
00085         unsigned int idx = 0;
00086         for (; idx < _size; ++idx) {
00087             if (_valueArray.at(idx) > iKey) {
00088                 break;
00089             }
00090         }
00091
00092         assert (idx != 0);
00093         assert (idx != _size);
00094
00095         //
00096         stdair::Probability_T& lCumulativeCurrentPoint =
00097             DictionaryManager::keyToValue (_cumulativeDistribution.at(idx));
00098         const T& lValueCurrentPoint = _valueArray.at(idx);
00099
00100         //
00101         stdair::Probability_T& lCumulativePreviousPoint =
00102             DictionaryManager::keyToValue (_cumulativeDistribution.at(idx-1));
00103         const T& lValuePreviousPoint = _valueArray.at(idx-1);
00104
00105         assert (lValueCurrentPoint != lValuePreviousPoint);
00106
00107         const double oValue= (lCumulativeCurrentPoint - lCumulativePreviousPoint)
00108             / (lValueCurrentPoint - lValuePreviousPoint);
00109
00110         return oValue;
00111     }
00112
00116     const T getUpperBound (const T iKey) const {
00117         // Find the first key value greater or equal to iKey.
00118         unsigned int idx = 0;
00119         for (; idx < _size; ++idx) {
00120             if (_valueArray.at(idx) > iKey) {
00121                 break;
00122             }
00123         }
00124         assert (idx != 0);
00125         assert (idx != _size);
00126
00127         return _valueArray.at (idx);
00128     }
00129
00130 public:
00131     // //////////// Display Support Methods ////////////
00135     const std::string displayCumulativeDistribution() const {
00136         std::ostream oStr;
00137
00138         for (unsigned int idx = 0; idx < _size; ++idx) {
00139             if (idx != 0) {
00140                 oStr << ", ";
00141             }
00142
00143             const stdair::Probability_T& lProbability =
00144                 DictionaryManager::keyToValue (_cumulativeDistribution.at(idx));
00145
00146             oStr << _valueArray.at(idx) << ":" << lProbability;
00147         }
00148         return oStr.str();
00149     }
00150
00151

```

```

00152 public:
00153     // ////////// Constructors and destructors //////////
00157     ContinuousAttributeLite (const ContinuousDistribution_T& iValueMap)
00158         : _size (iValueMap.size()) {
00159         init (iValueMap);
00160     }
00161
00165     ContinuousAttributeLite (const ContinuousAttributeLite& iCAL)
00166         : _size (iCAL._size),
00167         _cumulativeDistribution (iCAL._cumulativeDistribution),
00168         _valueArray (iCAL._valueArray) {
00169     }
00170
00174     ContinuousAttributeLite& operator= (const ContinuousAttributeLite& iCAL) {
00175         _size = iCAL._size;
00176         _cumulativeDistribution = iCAL._cumulativeDistribution;
00177         _valueArray = iCAL._valueArray;
00178         return *this;
00179     }
00180
00184     virtual ~ContinuousAttributeLite() {
00185     }
00186
00187 private:
00191     ContinuousAttributeLite() : _size(1) {
00192     }
00193
00198     void init (const ContinuousDistribution_T& iValueMap) {
00199         //
00200         const unsigned int lSize = iValueMap.size();
00201         _cumulativeDistribution.reserve (lSize);
00202         _valueArray.reserve (lSize);
00203
00204         // Browse the map to retrieve the values and cumulative probabilities.
00205         for (typename ContinuousDistribution_T::const_iterator it =
00206             iValueMap.begin(); it != iValueMap.end(); ++it) {
00207
00208             const T& attributeValue = it->first;
00209             const DictionaryKey_T& lKey = DictionaryManager::valueToKey (it->second);
00210
00211             // Build the two arrays.
00212             _cumulativeDistribution.push_back (lKey);
00213             _valueArray.push_back (attributeValue);
00214         }
00215     }
00216
00217 private:
00219     // ////////// Attributes //////////
00223     unsigned int _size;
00224
00228     std::vector<DictionaryKey_T> _cumulativeDistribution;
00229
00233     std::vector<T> _valueArray;
00234 };
00235
00236 }
00237 #endif // __TRADEMGEN_BAS_CONTINUOUSATTRIBUTE_LITE_HPP

```

25.37 trademgen/basic/DemandCharacteristics.cpp File Reference

```
#include <cassert>
#include <sstream>
#include <stdair/stdair_basic_types.hpp>
#include <trademgen/basic/BasConst_DemandGeneration.hpp>
#include <trademgen/basic/DemandCharacteristics.hpp>
```

Namespaces

- namespace [TRADEMGEN](#)

25.38 DemandCharacteristics.cpp

```

00001 // //////////////////////////////////////
00002 // Import section
00003 // //////////////////////////////////////
00004 // STL
00005 #include <cassert>
00006 #include <sstream>
00007 // StdAir
00008 #include <stdair/stdair_basic_types.hpp>
00009 // TraDemGen
00010 #include <trademgen/basic/BasConst_DemandGeneration.hpp>
00011 #include <trademgen/basic/DemandCharacteristics.hpp>
00012
00013 namespace TRADEMGEN {
00014
00015 // //////////////////////////////////////
00016 DemandCharacteristics::DemandCharacteristics()
00017 : _arrivalPattern (ArrivalPatternCumulativeDistribution_T()),
00018   _posProbabilityMass (POSPProbabilityMassFunction_T()),
00019   _channelProbabilityMass (ChannelProbabilityMassFunction_T()),
00020   _tripTypeProbabilityMass (TripTypeProbabilityMassFunction_T()),
00021   _stayDurationProbabilityMass (StayDurationProbabilityMassFunction_T()),
00022   _frequentFlyerProbabilityMass (FrequentFlyerProbabilityMassFunction_T()),
00023   _changeFeeProb (0.5), _changeFeeDisutility (0.0),
00024   _nonRefundableProb (0.5), _nonRefundableDisutility (0.0),
00025   _preferredDepartureTimeCumulativeDistribution (
PreferredDepartureTimeContinuousDistribution_T()),
00026   _minWTP (stdair::WTP_T()), _frat5Pattern (DEFAULT_FRAT5_PATTERN),
00027   _valueOfTimeCumulativeDistribution (ValueOfTimeContinuousDistribution_T())
00028 {
00029
00030 // //////////////////////////////////////
00031 DemandCharacteristics::
00032 DemandCharacteristics (const DemandCharacteristics& iDC)
00033 : _arrivalPattern (iDC._arrivalPattern),
00034   _posProbabilityMass (iDC._posProbabilityMass),
00035   _channelProbabilityMass (iDC._channelProbabilityMass),
00036   _tripTypeProbabilityMass (iDC._tripTypeProbabilityMass),
00037   _stayDurationProbabilityMass (iDC._stayDurationProbabilityMass),
00038   _frequentFlyerProbabilityMass (iDC._frequentFlyerProbabilityMass),
00039   _changeFeeProb (iDC._changeFeeProb),
00040   _changeFeeDisutility (iDC._changeFeeDisutility),
00041   _nonRefundableProb (iDC._nonRefundableProb),
00042   _nonRefundableDisutility (iDC._nonRefundableDisutility),
00043   _preferredDepartureTimeCumulativeDistribution (iDC._preferredDepartureTimeCumulativeDistribution),
00044   _minWTP (iDC._minWTP), _frat5Pattern (iDC._frat5Pattern),
00045   _valueOfTimeCumulativeDistribution (iDC._valueOfTimeCumulativeDistribution)
00046 {
00047
00048 // //////////////////////////////////////
00049 DemandCharacteristics::
00050 DemandCharacteristics (const ArrivalPatternCumulativeDistribution_T& iArrivalPattern,
00051                       const POSProbabilityMassFunction_T& iPOSProbMass,
00052                       const ChannelProbabilityMassFunction_T& iChannelProbMass,
00053                       const TripTypeProbabilityMassFunction_T& iTripTypeProbMass,
00054                       const StayDurationProbabilityMassFunction_T& iStayDurationProbMass,
00055                       const FrequentFlyerProbabilityMassFunction_T& iFrequentFlyerProbMass,
00056                       const stdair::ChangeFeesRatio_T& iChangeFeeProb,

```

```

00057             const stdair::Disutility_T& iChangeFeeDisutility,
00058             const stdair::NonRefundableRatio_T& iNonRefundableProb,
00059             const stdair::Disutility_T& iNonRefundableDisutility,
00060             const PreferredDepartureTimeContinuousDistribution_T& iP
referredDepartureTimeContinuousDistribution,
00061             const stdair::WTP_T& iMinWTP,
00062             const ValueOfTimeContinuousDistribution_T& iValueOfTimeC
ontinuousDistribution)
00063     : _arrivalPattern (iArrivalPattern),
00064       _posProbabilityMass (iPOSProbMass),
00065       _channelProbabilityMass (iChannelProbMass),
00066       _tripTypeProbabilityMass (iTripTypeProbMass),
00067       _stayDurationProbabilityMass (iStayDurationProbMass),
00068       _frequentFlyerProbabilityMass (iFrequentFlyerProbMass),
00069       _changeFeeProb (iChangeFeeProb),
00070       _changeFeeDisutility (iChangeFeeDisutility),
00071       _nonRefundableProb (iNonRefundableProb),
00072       _nonRefundableDisutility (iNonRefundableDisutility),
00073       _preferredDepartureTimeCumulativeDistribution (iPreferredDepartureTimeConti
nuousDistribution),
00074       _minWTP (iMinWTP), _frat5Pattern (DEFAULT_FRAT5_PATTERN),
00075       _valueOfTimeCumulativeDistribution (iValueOfTimeContinuousDistribution) {
00076     }
00077
00078     // //////////////////////////////////////
00079     DemandCharacteristics::~DemandCharacteristics() {
00080     }
00081
00082     // //////////////////////////////////////
00083     const stdair::AirportCode_T& DemandCharacteristics::
00084     getPOSValue (const stdair::Probability_T& iCumulativeProbability) const {
00085         return _posProbabilityMass.getValue (iCumulativeProbability);
00086     }
00087
00088     // //////////////////////////////////////
00089     bool DemandCharacteristics::
00090     checkPOSValue (const stdair::AirportCode_T& iPOS) const {
00091         return _posProbabilityMass.checkValue (iPOS);
00092     }
00093
00094     // //////////////////////////////////////
00095     const std::string DemandCharacteristics::describe() const {
00096         std::ostringstream ostr;
00097
00098         //
00099         ostr << "***** Demand characteristics *****"
00100         << std::endl;
00101         ostr << "Arrival pattern (days from departure, proportion): ";
00102         ostr << _arrivalPattern.displayCumulativeDistribution() << std::endl;
00103         ostr << "POS probability mass (POS, propotion): ";
00104         ostr << _posProbabilityMass.displayProbabilityMass()
00105         << std::endl;
00106         ostr << "Channel probability mass (channel, propotion): ";
00107         ostr << _channelProbabilityMass.displayProbabilityMass()
00108         << std::endl;
00109         ostr << "Trip type probability mass (trip type, propotion): ";
00110         ostr << _tripTypeProbabilityMass.displayProbabilityMass()
00111         << std::endl;
00112         ostr << "Stay duration probability mass (number of days, propotion): ";
00113         ostr << _stayDurationProbabilityMass.displayProbabilityMass()
00114         << std::endl;
00115         ostr << "Frequent flyer probability mass (frequent flyer, propotion): ";
00116         ostr << _frequentFlyerProbabilityMass.displayProbabilityMass()
00117         << std::endl;
00118         ostr << "Change fee acceptance probability: "
00119         << _changeFeeProb
00120         << std::endl;

```

```
00121     ostr << "Change fee disutility: "
00122         << _changeFeeDisutility
00123         << std::endl;
00124     ostr << "Non refundable acceptance probability: "
00125         << _nonRefundableProb
00126         << std::endl;
00127     ostr << "Non refundable disutility: "
00128         << _nonRefundableDisutility
00129         << std::endl;
00130     ostr << "Preferred departure time cumulative distribution (time, proportion:
";
00131     ostr << _preferredDepartureTimeCumulativeDistribution.
displayCumulativeDistribution() << std::endl;
00132     ostr << "min WTP: " << _minWTP << std::endl;
00133     ostr << "Value of time cumulative distribution (value of time, proportion: ";

00134     ostr << _valueOfTimeCumulativeDistribution.displayCumulativeDistribution()
00135         << std::endl;
00136
00137
00138     return ostr.str();
00139 }
00140
00141 }
00142
```

25.39 trademgen/basic/DemandCharacteristics.hpp File Reference

```
#include <string>
#include <stdair/stdair_basic_types.hpp>
#include <stdair/stdair_date_time_types.hpp>
#include <stdair/basic/StructAbstract.hpp>
#include <trademgen/basic/DemandCharacteristicsTypes.hpp>
```

Classes

- struct [TRADEMGEN::DemandCharacteristics](#)
Class modeling the characteristics of a demand type.

Namespaces

- namespace [TRADEMGEN](#)

25.40 DemandCharacteristics.hpp

```

00001 #ifndef __TRADEMGEN_BAS_DEMAND_CHARACTERISTICS_HPP
00002 #define __TRADEMGEN_BAS_DEMAND_CHARACTERISTICS_HPP
00003
00004 // //////////////////////////////////////
00005 // Import section
00006 // //////////////////////////////////////
00007 // STL
00008 #include <string>
00009 // StdAir
00010 #include <stdair/stdair_basic_types.hpp>
00011 #include <stdair/stdair_date_time_types.hpp>
00012 #include <stdair/basic/StructAbstract.hpp>
00013 // TraDemGen
00014 #include <trademgen/basic/DemandCharacteristicsTypes.hpp>
00015
00016 namespace TRADEMGEN {
00017
00021     struct DemandCharacteristics : public stdair::StructAbstract {
00022
00023     public:
00024         // ////////////////////////////////// Business support methods //////////////////////////////////
00028         const stdair::AirportCode_T&
00029         getPOSValue (const stdair::Probability_T& iCumulativeProbability) const;
00030
00034         bool checkPOSValue (const stdair::AirportCode_T& iPOS) const;
00035
00036
00037     public:
00038         // ////////////////////////////////// Display support methods //////////////////////////////////
00042         const std::string describe() const;
00043
00044
00045     public:
00046         // ////////////////////////////////// Constructors and destructors //////////////////////////////////
00050         DemandCharacteristics (const ArrivalPatternCumulativeDistribution_T&,
00051                               const POSProbabilityMassFunction_T&,
00052                               const ChannelProbabilityMassFunction_T&,
00053                               const TripTypeProbabilityMassFunction_T&,
00054                               const StayDurationProbabilityMassFunction_T&,
00055                               const FrequentFlyerProbabilityMassFunction_T&,
00056                               const stdair::ChangeFeesRatio_T&,
00057                               const stdair::Disutility_T&,
00058                               const stdair::NonRefundableRatio_T&,
00059                               const stdair::Disutility_T&,
00060                               const PreferredDepartureTimeContinuousDistribution_T&,
00061
00062                               const stdair::WTP_T&,
00063                               const ValueOfTimeContinuousDistribution_T&);
00064
00067         DemandCharacteristics ();
00068
00072         DemandCharacteristics (const DemandCharacteristics&);
00073
00077         ~DemandCharacteristics ();
00078
00079
00080     public:
00081         // ////////////////////////////////// Attributes //////////////////////////////////
00087         ContinuousFloatDuration_T _arrivalPattern;
00088
00092         POSProbabilityMass_T _posProbabilityMass;
00093
00097         ChannelProbabilityMass_T _channelProbabilityMass;
00098
00102         TripTypeProbabilityMass_T _tripTypeProbabilityMass;

```

```
00103
00107     StayDurationProbabilityMass_T _stayDurationProbabilityMass;
00108
00112     FrequentFlyerProbabilityMass_T _frequentFlyerProbabilityMass;
00113
00117     stdair::ChangeFeesRatio_T _changeFeeProb;
00118
00122     stdair::Disutility_T _changeFeeDisutility;
00123
00127     stdair::NonRefundableRatio_T _nonRefundableProb;
00128
00132     stdair::Disutility_T _nonRefundableDisutility;
00133
00137     PreferredDepartureTimeCumulativeDistribution_T
    _preferredDepartureTimeCumulativeDistribution;
00138
00143     stdair::WTP_T _minWTP;
00144
00148     CumulativeDistribution_T _frat5Pattern;
00149
00153     ValueOfTimeCumulativeDistribution_T _valueOfTimeCumulativeDistribution;
00154 };
00155
00156 }
00157 #endif // __TRADEMGEN_BAS_DEMAND_CHARACTERISTICS_HPP
```

25.41 trademgen/basic/DemandCharacteristicsTypes.hpp File Reference

```
#include <stdair/stdair_basic_types.hpp>
#include <stdair/stdair_date_time_types.hpp>
#include <stdair/stdair_demand_types.hpp>
#include <trademgen/basic/ContinuousAttributeLite.hpp>
#include <trademgen/basic/CategoricalAttributeLite.hpp>
```

Namespaces

- namespace [TRADEMG](#)

Typedefs

- typedef ContinuousAttributeLite< stdair::FloatDuration_T > [TRADEMG::ContinuousFloatDuration_-T](#)
- typedef ContinuousFloatDuration_T::ContinuousDistribution_T [TRADEMG::ArrivalPatternCumulativeDistribution_T](#)
- typedef CategoricalAttributeLite< stdair::AirportCode_T > [TRADEMG::POSProbabilityMass_-T](#)
- typedef POSProbabilityMass_T::ProbabilityMassFunction_T [TRADEMG::POSProbabilityMassFunction_-T](#)
- typedef CategoricalAttributeLite< stdair::ChannelLabel_T > [TRADEMG::ChannelProbabilityMass_-T](#)
- typedef ChannelProbabilityMass_T::ProbabilityMassFunction_T [TRADEMG::ChannelProbabilityMassFunction_-T](#)
- typedef CategoricalAttributeLite< stdair::TripType_T > [TRADEMG::TripTypeProbabilityMass_-T](#)
- typedef [TripTypeProbabilityMass_T::ProbabilityMassFunction_T](#)
[TRADEMG::TripTypeProbabilityMassFunction_T](#)
- typedef CategoricalAttributeLite< stdair::DayDuration_T > [TRADEMG::StayDurationProbabilityMass_-T](#)
- typedef [StayDurationProbabilityMass_T::ProbabilityMassFunction_T](#)
[TRADEMG::StayDurationProbabilityMassFunction_T](#)
- typedef CategoricalAttributeLite< stdair::FrequentFlyer_T > [TRADEMG::FrequentFlyerProbabilityMass_-T](#)
- typedef [FrequentFlyerProbabilityMass_T::ProbabilityMassFunction_T](#)
[TRADEMG::FrequentFlyerProbabilityMassFunction_T](#)
- typedef ContinuousAttributeLite< stdair::IntDuration_T > [TRADEMG::PreferredDepartureTimeCumulativeDistribution_T](#)
- typedef [PreferredDepartureTimeCumulativeDistribution_T::ContinuousDistribution_T](#)
[TRADEMG::PreferredDepartureTimeContinuousDistribution_T](#)
- typedef ContinuousAttributeLite< stdair::PriceValue_T > [TRADEMG::ValueOfTimeCumulativeDistribution_-T](#)
- typedef [ValueOfTimeCumulativeDistribution_T::ContinuousDistribution_T](#)
[TRADEMG::ValueOfTimeContinuousDistribution_T](#)
- typedef ContinuousAttributeLite< stdair::RealNumber_T > [TRADEMG::CumulativeDistribution_-T](#)
- typedef CumulativeDistribution_T::ContinuousDistribution_T [TRADEMG::FRAT5Pattern_T](#)

25.42 DemandCharacteristicsTypes.hpp

```

00001 #ifndef __TRADEMGEN_BAS_DEMANDCHARACTERISTICSTYPES_HPP
00002 #define __TRADEMGEN_BAS_DEMANDCHARACTERISTICSTYPES_HPP
00003
00004 // //////////////////////////////////////
00005 // Import section
00006 // //////////////////////////////////////
00007 // StdAir
00008 #include <stdair/stdair_basic_types.hpp>
00009 #include <stdair/stdair_date_time_types.hpp>
00010 #include <stdair/stdair_demand_types.hpp>
00011 // TraDemGen
00012 #include <trademgen/basic/ContinuousAttributeLite.hpp>
00013 #include <trademgen/basic/CategoricalAttributeLite.hpp>
00014
00015 namespace TRADEMGEN {
00016
00019     typedef ContinuousAttributeLite<stdair::FloatDuration_T>
        ContinuousFloatDuration_T;
00020
00022     typedef ContinuousFloatDuration_T::ContinuousDistribution_T
        ArrivalPatternCumulativeDistribution_T;
00023
00025     typedef CategoricalAttributeLite<stdair::AirportCode_T> POSProbabilityMass_T;
00026
00028     typedef POSProbabilityMass_T::ProbabilityMassFunction_T
        POSProbabilityMassFunction_T;
00029
00031     typedef CategoricalAttributeLite<stdair::ChannelLabel_T>
        ChannelProbabilityMass_T;
00032
00034     typedef ChannelProbabilityMass_T::ProbabilityMassFunction_T
        ChannelProbabilityMassFunction_T;
00035
00037     typedef CategoricalAttributeLite<stdair::TripType_T> TripTypeProbabilityMass_T;
00038
00040     typedef TripTypeProbabilityMass_T::ProbabilityMassFunction_T
        TripTypeProbabilityMassFunction_T;
00041
00043     typedef CategoricalAttributeLite<stdair::DayDuration_T>
        StayDurationProbabilityMass_T;
00044
00046     typedef StayDurationProbabilityMass_T::ProbabilityMassFunction_T
        StayDurationProbabilityMassFunction_T;
00047
00049     typedef CategoricalAttributeLite<stdair::FrequentFlyer_T>
        FrequentFlyerProbabilityMass_T;
00050
00052     typedef FrequentFlyerProbabilityMass_T::ProbabilityMassFunction_T
        FrequentFlyerProbabilityMassFunction_T;
00053
00055     typedef ContinuousAttributeLite<stdair::IntDuration_T>
        PreferredDepartureTimeCumulativeDistribution_T;
00056
00058     typedef
        PreferredDepartureTimeCumulativeDistribution_T::ContinuousDistribution_T
        PreferredDepartureTimeContinuousDistribution_T;
00059
00061     typedef ContinuousAttributeLite<stdair::PriceValue_T>
        ValueOfTimeCumulativeDistribution_T;
00062
00064     typedef ValueOfTimeCumulativeDistribution_T::ContinuousDistribution_T
        ValueOfTimeContinuousDistribution_T;
00065
00067     typedef ContinuousAttributeLite<stdair::RealNumber_T> CumulativeDistribution_T;

```



```
00068     typedef CumulativeDistribution_T::ContinuousDistribution_T FRAT5Pattern_T;
00069 }
00070 #endif // __TRADEMGEN_BAS_DEMANDCHARACTERISTICSTYPES_HPP
```

25.43 trademgen/basic/DemandDistribution.cpp File Reference

```
#include <cassert>
#include <sstream>
#include <stdair/stdair_date_time_types.hpp>
#include <trademgen/basic/DemandDistribution.hpp>
```

Namespaces

- namespace [TRADEMGEN](#)

25.44 DemandDistribution.cpp

```

00001 // //////////////////////////////////////
00002 // Import section
00003 // //////////////////////////////////////
00004 // STL
00005 #include <cassert>
00006 #include <sstream>
00007 // StdAir
00008 #include <stdair/stdair_date_time_types.hpp>
00009 // TraDemGen
00010 #include <trademgen/basic/DemandDistribution.hpp>
00011
00012 namespace TRADEMGEN {
00013
00014     // //////////////////////////////////////
00015     DemandDistribution::DemandDistribution (const stdair::NbOfRequests_T& iMean,
00016                                             const stdair::StdDevValue_T& iStdDev)
00017         : _meanNumberOfRequests (iMean),
00018           _stdDevNumberOfRequests (iStdDev) {
00019     }
00020
00021     // //////////////////////////////////////
00022     DemandDistribution::DemandDistribution() {
00023     }
00024
00025     // //////////////////////////////////////
00026     DemandDistribution::~DemandDistribution() {
00027     }
00028
00029     // //////////////////////////////////////
00030     DemandDistribution::
00031     DemandDistribution (const DemandDistribution& iDemandDistribution)
00032         : _meanNumberOfRequests (iDemandDistribution._meanNumberOfRequests),
00033           _stdDevNumberOfRequests (iDemandDistribution._stdDevNumberOfRequests) {
00034     }
00035
00036     // //////////////////////////////////////
00037     void DemandDistribution::fromStream (std::istream& ioIn) {
00038     }
00039
00040     // //////////////////////////////////////
00041     const std::string DemandDistribution::describe() const {
00042         std::ostringstream oStr;
00043         oStr << "N (" << _meanNumberOfRequests << ", "
00044             << _stdDevNumberOfRequests << ")";
00045         return oStr.str();
00046     }
00047
00048     // //////////////////////////////////////
00049     std::string DemandDistribution::display() const {
00050         std::ostringstream oStr;
00051         oStr << describe() << std::endl;
00052         return oStr.str();
00053     }
00054
00055 }
00056

```

25.45 trademgen/basic/DemandDistribution.hpp File Reference

```
#include <string>
#include <stdair/stdair_basic_types.hpp>
#include <stdair/basic/StructAbstract.hpp>
#include <trademgen/basic/ContinuousAttribute.hpp>
```

Classes

- struct [TRADEMGEN::DemandDistribution](#)
Class modeling the distribution of a demand type.

Namespaces

- namespace [TRADEMGEN](#)

25.46 DemandDistribution.hpp

```

00001 #ifndef __TRADEMGEN_BAS_DEMAND_DISTRIBUTION_HPP
00002 #define __TRADEMGEN_BAS_DEMAND_DISTRIBUTION_HPP
00003
00004 // //////////////////////////////////////
00005 // Import section
00006 // //////////////////////////////////////
00007 // STL
00008 #include <string>
00009 // StdAir
00010 #include <stdair/stdair_basic_types.hpp>
00011 #include <stdair/basic/StructAbstract.hpp>
00012 // TraDemGen
00013 #include <trademgen/basic/ContinuousAttribute.hpp>
00014
00015 namespace TRADEMGEN {
00016
00020     struct DemandDistribution : public stdair::StructAbstract {
00021     public:
00022         // ////////////////////////////////// Constructors and destructors //////////////////////////////////
00026         DemandDistribution (const stdair::NbOfRequests_T& iMean,
00027                             const stdair::StdDevValue_T& iStdDev);
00031         DemandDistribution();
00035         DemandDistribution (const DemandDistribution&);
00039         ~DemandDistribution();
00040
00041
00042     public:
00043         // ////////////////////////////////// Display Support Methods //////////////////////////////////
00049         void fromStream (std::istream& ioIn);
00050
00054         const std::string describe() const;
00055
00059         std::string display() const;
00060
00061
00062     public:
00063         // ////////////////////////////////// Attributes //////////////////////////////////
00067         stdair::NbOfRequests_T _meanNumberOfRequests;
00068
00072         stdair::StdDevValue_T _stdDevNumberOfRequests;
00073     };
00074
00075 }
00076 #endif // __TRADEMGEN_BAS_DEMAND_DISTRIBUTION_HPP

```

25.47 trademgen/basic/DictionaryManager.cpp File Reference

```
#include <trademgen/basic/DictionaryManager.hpp>
```

Namespaces

- namespace [TRADEMGEN](#)

25.48 DictionaryManager.cpp

```
00001 // ////////////////////////////////////////
00002 // Import section
00003 // ////////////////////////////////////////
00004 // TRADEMGEN
00005 #include <trademgen/basic/DictionaryManager.hpp>
00006
00007 namespace TRADEMGEN {
00008     // ////////////////////////////////////////
00009     const stdair::Probability_T DictionaryManager::
00010     keyToValue (const DictionaryKey_T iKey) {
00011         // return static_cast<stdair::Probability_T>(iKey) / 1000;
00012         return iKey;
00013     }
00014
00015     // ////////////////////////////////////////
00016     const DictionaryKey_T DictionaryManager::
00017     valueToKey (const stdair::Probability_T iValue) {
00018         // return iValue * 1000;
00019         return iValue;
00020     }
00021 }
```

25.49 trademgen/basic/DictionaryManager.hpp File Reference

```
#include <stdair/stdair_maths_types.hpp>
```

Classes

- class [TRADEMGEN::DictionaryManager](#)
Class wrapper of dictionary business methods.

Namespaces

- namespace [TRADEMGEN](#)

Typedefs

- typedef stdair::Probability_T [TRADEMGEN::DictionaryKey_T](#)

25.50 DictionaryManager.hpp

```
00001 ///////////////////////////////////////////////////////////////////
00002 #ifndef __TRADEMGEN_BASIC_DICTIONARYMANAGER_HPP
00003 #define __TRADEMGEN_BASIC_DICTIONARYMANAGER_HPP
00004
00005 ///////////////////////////////////////////////////////////////////
00006 // Import section
00007 ///////////////////////////////////////////////////////////////////
00008 // StdAir
00009 #include <stdair/stdair_maths_types.hpp>
00010
00011 namespace TRADEMGEN {
00012
00013     ///////////////////////////////////////////////////////////////////
00014     // Type definitions ///////////////////////////////////////////////////////////////////
00015     //typedef unsigned short DictionaryKey_T;
00016     typedef stdair::Probability_T DictionaryKey_T;
00017
00021     class DictionaryManager {
00022     public:
00023         ///////////////////////////////////////////////////////////////////
00024         // Business methods ///////////////////////////////////////////////////////////////////
00027         static const stdair::Probability_T keyToValue (const DictionaryKey_T);
00028
00032         static const DictionaryKey_T valueToKey (const stdair::Probability_T);
00033     };
00034 }
00035 #endif // __TRADEMGEN_BASIC_DICTIONARYMANAGER_HPP
```

25.51 trademgen/basic/RandomGenerationContext.cpp File Reference

```
#include <cassert>
#include <sstream>
#include <trademgen/basic/RandomGenerationContext.hpp>
```

Namespaces

- namespace [TRADEMGEN](#)

25.52 RandomGenerationContext.cpp

```

00001 // //////////////////////////////////////
00002 // Import section
00003 // //////////////////////////////////////
00004 // STL
00005 #include <cassert>
00006 #include <sstream>
00007 // TraDemGen
00008 #include <trademgen/basic/RandomGenerationContext.hpp>
00009
00010 namespace TRADEMGEN {
00011
00012 // //////////////////////////////////////
00013 RandomGenerationContext::RandomGenerationContext ()
00014 : _numberOfRequestsGeneratedSoFar (0),
00015   _cumulativeProbabilitySoFar (0.0) {
00016 }
00017
00018 // //////////////////////////////////////
00019 RandomGenerationContext::
00020 RandomGenerationContext (const RandomGenerationContext& iRGC)
00021 : _numberOfRequestsGeneratedSoFar (iRGC._numberOfRequestsGeneratedSoFar),
00022   _cumulativeProbabilitySoFar (iRGC._cumulativeProbabilitySoFar) {
00023 }
00024
00025 // //////////////////////////////////////
00026 RandomGenerationContext::~RandomGenerationContext () {
00027 }
00028
00029 // //////////////////////////////////////
00030 const std::string RandomGenerationContext::describe() const {
00031     std::ostringstream oStr;
00032     oStr << _numberOfRequestsGeneratedSoFar
00033         << " => " << _cumulativeProbabilitySoFar;
00034     return oStr.str();
00035 }
00036
00037 // //////////////////////////////////////
00038 void RandomGenerationContext::incrementGeneratedRequestsCounter() {
00039     ++_numberOfRequestsGeneratedSoFar;
00040 }
00041
00042 // //////////////////////////////////////
00043 void RandomGenerationContext::reset () {
00044     _cumulativeProbabilitySoFar = 0.0;
00045     _numberOfRequestsGeneratedSoFar = 0;
00046 }
00047
00048 }

```

25.53 trademgen/basic/RandomGenerationContext.hpp File Reference

```
#include <iosfwd>
#include <string>
#include <stdair/stdair_basic_types.hpp>
#include <stdair/stdair_maths_types.hpp>
#include <stdair/basic/StructAbstract.hpp>
```

Classes

- struct [TRADEMGEN::RandomGenerationContext](#)

Namespaces

- namespace [TRADEMGEN](#)

25.54 RandomGenerationContext.hpp

```

00001 #ifndef __TRADEMGEN_BAS_RANDOM_GENERATION_CONTEXT_HPP
00002 #define __TRADEMGEN_BAS_RANDOM_GENERATION_CONTEXT_HPP
00003
00004 // //////////////////////////////////////
00005 // Import section
00006 // //////////////////////////////////////
00007 // STL
00008 #include <iosfwd>
00009 #include <string>
00010 // StdAir
00011 #include <stdair/stdair_basic_types.hpp>
00012 #include <stdair/stdair_maths_types.hpp>
00013 #include <stdair/basic/StructAbstract.hpp>
00014
00015 namespace TRADEMGEN {
00016
00020     struct RandomGenerationContext : public stdair::StructAbstract {
00021     public:
00022         // ////////////////////////////////// Getters //////////////////////////////////
00026         const stdair::Count_T& getNumberOfRequestsGeneratedSoFar() const {
00027             return _numberOfRequestsGeneratedSoFar;
00028         }
00029
00034         const stdair::Probability_T& getCumulativeProbabilitySoFar() const {
00035             return _cumulativeProbabilitySoFar;
00036         }
00037
00038     public:
00039         // ////////////////////////////////// Setters //////////////////////////////////
00043         void setNumberOfRequestsGeneratedSoFar (const stdair::Count_T& iCount) {
00044             _numberOfRequestsGeneratedSoFar = iCount;
00045         }
00046
00051         void setCumulativeProbabilitySoFar (const stdair::Probability_T& iProb) {
00052             _cumulativeProbabilitySoFar = iProb;
00053         }
00054
00055
00056     public:
00057         // ////////////////////////////////// Constructors and destructors //////////////////////////////////
00061         RandomGenerationContext ();
00062
00066         RandomGenerationContext (const RandomGenerationContext&);
00067
00071         ~RandomGenerationContext ();
00072
00073
00074     public:
00075         // ////////////////////////////////// Business Methods //////////////////////////////////
00079         void incrementGeneratedRequestsCounter ();
00080
00084         void reset ();
00085
00086
00087     public:
00088         // ////////////////////////////////// Display Support Methods //////////////////////////////////
00092         const std::string describe() const;
00093
00094
00095     private:
00096         // ////////////////////////////////// Attributes //////////////////////////////////
00100         stdair::Count_T _numberOfRequestsGeneratedSoFar;
00101
00106         stdair::Probability_T _cumulativeProbabilitySoFar;
00107     };

```

```
00108  
00109 }  
00110 #endif // __STDAIR_BAS_RANDOM_GENERATION_CONTEXT_HPP
```

25.55 trademgen/batches/trademgen_generateDemand.cpp File Reference

```
#include <cassert>
#include <sstream>
#include <fstream>
#include <vector>
#include <list>
#include <string>
#include <boost/tokenizer.hpp>
#include <boost/program_options.hpp>
#include <boost/accumulators/accumulators.hpp>
#include <boost/accumulators/statistics.hpp>
#include <stdair/stdair_basic_types.hpp>
#include <stdair/basic/BasConst_General.hpp>
#include <stdair/basic/ProgressStatusSet.hpp>
#include <stdair/basic/DemandGenerationMethod.hpp>
#include <stdair/bom/EventStruct.hpp>
#include <stdair/bom/BookingRequestStruct.hpp>
#include <stdair/bom/BomDisplay.hpp>
#include <stdair/service/Logger.hpp>
#include <trademgen/TRADEMGEN_Service.hpp>
#include <trademgen/config/trademgen-paths.hpp>
```

Typedefs

- typedef unsigned int [NbOfRuns_T](#)
- typedef ba::accumulator_set< double, ba::stats< ba::tag::min, ba::tag::max, ba::tag::mean(ba::immediate), ba::tag::sum, ba::tag::variance > > [stat_acc_type](#)

Functions

- const stdair::Filename_T [K_TRADEMGEN_DEFAULT_LOG_FILENAME](#) ("trademgen_generateDemand.log")
- const stdair::Filename_T [K_TRADEMGEN_DEFAULT_INPUT_FILENAME](#) (STDAIR_SAMPLE_DIR"/demand01.csv")
- const stdair::Filename_T [K_TRADEMGEN_DEFAULT_OUTPUT_FILENAME](#) ("request.csv")
- void [stat_display](#) (std::ostream &oStream, const [stat_acc_type](#) &iStatAcc)
- template<class T >
std::ostream &[operator<<](#) (std::ostream &os, const std::vector< T > &v)
- int [readConfiguration](#) (int argc, char *argv[], bool &ioIsBuiltin, stdair::RandomSeed_T &ioRandomSeed, [NbOfRuns_T](#) &ioRandomRuns, stdair::Filename_T &ioInputFilename, stdair::Filename_T &ioOutputFilename, stdair::Filename_T &ioLogFilename, stdair::DemandGenerationMethod &ioDemandGenerationMethod)

- void [generateDemand](#) ([TRADEMGEN::TRADEMGEN_Service](#) &ioTrademgenService, const [stdair::Filename_T](#) &iOutputFilename, const [NbOfRuns_T](#) &iNbOfRuns, const [stdair::DemandGenerationMethod](#) &iDemandGenerationMethod)
- int [main](#) (int argc, char *argv[])

Variables

- const [stdair::DemandGenerationMethod](#) [K_TRADEMGEN_DEFAULT_DEMAND_GENERATION_METHOD](#)
- const char [K_TRADEMGEN_DEFAULT_DEMAND_GENERATION_METHOD_CHAR](#)
- const [stdair::RandomSeed_T](#) [K_TRADEMGEN_DEFAULT_RANDOM_SEED](#)
- const [NbOfRuns_T](#) [K_TRADEMGEN_DEFAULT_RANDOM_DRAWS](#) = 1
- const bool [K_TRADEMGEN_DEFAULT_BUILT_IN_INPUT](#) = false
- const int [K_TRADEMGEN_EARLY_RETURN_STATUS](#) = 99

25.55.1 Typedef Documentation

25.55.1.1 typedef unsigned int NbOfRuns_T

Definition at line 38 of file [trademgen_generateDemand.cpp](#).

25.55.1.2 typedef ba::accumulator_set<double, ba::stats<ba::tag::min, ba::tag::max, ba::tag::mean (ba::immediate), ba::tag::sum, ba::tag::variance> > stat_acc_type

Type definition to gather statistics.

Definition at line 47 of file [trademgen_generateDemand.cpp](#).

25.55.2 Function Documentation

25.55.2.1 const stdair::Filename_T K_TRADEMGEN_DEFAULT_LOG_FILENAME ("trademgen_generateDemand.log")

Default name and location for the log file.

Referenced by [readConfiguration\(\)](#).

25.55.2.2 const stdair::Filename_T K_TRADEMGEN_DEFAULT_INPUT_FILENAME (STDAIR_SAMPLE_DIR"/demand01.csv")

Default name and location for the (CSV) input file.

Referenced by [readConfiguration\(\)](#).

25.55.2.3 const stdair::Filename_T K_TRADEMGEN_DEFAULT_OUTPUT_FILENAME ("request.csv")

Default name and location for the (CSV) output file.

Referenced by [readConfiguration\(\)](#).

25.55.2.4 void stat_display (std::ostream & oStream, const stat_acc_type & iStatAcc)

Display the statistics held by the dedicated accumulator.

Definition at line 105 of file [trademgen_generateDemand.cpp](#).

Referenced by [generateDemand\(\)](#).

25.55.2.5 template<class T > std::ostream& operator<< (std::ostream & os, const std::vector< T > & v) [inline]

Definition at line 127 of file [trademgen_generateDemand.cpp](#).

25.55.2.6 int readConfiguration (int argc, char * argv[], bool & ioIsBuiltin, stdair::RandomSeed_T & ioRandomSeed, NbOfRuns_T & ioRandomRuns, stdair::Filename_T & ioInputFilename, stdair::Filename_T & ioOutputFilename, stdair::Filename_T & ioLogFilename, stdair::DemandGenerationMethod & ioDemandGenerationMethod)

Read and parse the command line options.

Definition at line 136 of file [trademgen_generateDemand.cpp](#).

References [K_TRADEMGEN_DEFAULT_DEMAND_GENERATION_METHOD_CHAR](#), [K_TRADEMGEN_DEFAULT_INPUT_FILENAME\(\)](#), [K_TRADEMGEN_DEFAULT_LOG_FILENAME\(\)](#), and [K_TRADEMGEN_DEFAULT_OUTPUT_FILENAME\(\)](#).

Referenced by [main\(\)](#).

25.55.2.7 void generateDemand (TRADEMGEN::TRADEMGEN_Service & ioTrademgenService, const stdair::Filename_T & iOutputFilename, const NbOfRuns_T & iNbOfRuns, const stdair::DemandGenerationMethod & iDemandGenerationMethod)

Definition at line 276 of file [trademgen_generateDemand.cpp](#).

References [TRADEMGEN::TRADEMGEN_Service::csvDisplay\(\)](#), [TRADEMGEN::TRADEMGEN_Service::generateFirstRequests\(\)](#), [TRADEMGEN::TRADEMGEN_Service::generateNextRequest\(\)](#), [TRADEMGEN::TRADEMGEN_Service::getExpectedTotalNumberOfRequestsToBeGenerated\(\)](#), [TRADEMGEN::TRADEMGEN_Service::isQueueDone\(\)](#), [TRADEMGEN::TRADEMGEN_Service::popEvent\(\)](#), [TRADEMGEN::TRADEMGEN_Service::reset\(\)](#), and [stat_display\(\)](#).

Referenced by [main\(\)](#), and [TRADEMGEN::TRADEMGEN_Service::parseAndLoad\(\)](#).

25.55.2.8 int main (int argc, char * argv[])

Definition at line 420 of file [trademgen_generateDemand.cpp](#).

References [TRADEMGEN::TRADEMGEN_Service::buildSampleBom\(\)](#), [generateDemand\(\)](#), [K_TRADEMGEN_DEFAULT_DEMAND_GENERATION_METHOD](#), [TRADEMGEN::TRADEMGEN_Service::parseAndLoad\(\)](#), and [readConfiguration\(\)](#).

25.55.3 Variable Documentation

25.55.3.1 `const stdair::DemandGenerationMethod K_TRADEMGEN_DEFAULT_DEMAND_GENERATION_METHOD`

Initial value:

```
stdair::DemandGenerationMethod::POI_PRO
```

Default demand generation method: Poisson Process.

Definition at line 70 of file [trademgen_generateDemand.cpp](#).

Referenced by [main\(\)](#).

25.55.3.2 `const char K_TRADEMGEN_DEFAULT_DEMAND_GENERATION_METHOD_CHAR`

Initial value:

```
K_TRADEMGEN_DEFAULT_DEMAND_GENERATION_METHOD.getMethodAsChar()
```

Default demand generation method name: 'P' for Poisson Process.

Definition at line 76 of file [trademgen_generateDemand.cpp](#).

Referenced by [readConfiguration\(\)](#).

25.55.3.3 `const stdair::RandomSeed_T K_TRADEMGEN_DEFAULT_RANDOM_SEED`

Initial value:

```
stdair::DEFAULT_RANDOM_SEED
```

Default random generation seed (e.g., 120765987).

Definition at line 82 of file [trademgen_generateDemand.cpp](#).

Referenced by [readConfiguration\(\)](#).

25.55.3.4 `const NbOfRuns_T K_TRADEMGEN_DEFAULT_RANDOM_DRAWS = 1`

Default number of random draws to be generated (best if over 100).

Definition at line 88 of file [trademgen_generateDemand.cpp](#).

25.55.3.5 `const bool K_TRADEMGEN_DEFAULT_BUILT_IN_INPUT = false`

Default for the input type. It can be either built-in or provided by an input file. That latter must then be given with the -i option.

Definition at line 94 of file [trademgen_generateDemand.cpp](#).

Referenced by [readConfiguration\(\)](#).

25.55.3.6 const int K_TRADEMGEN_EARLY_RETURN_STATUS = 99

Early return status (so that it can be differentiated from an error).

Definition at line 99 of file [trademgen_generateDemand.cpp](#).

Referenced by [main\(\)](#), and [readConfiguration\(\)](#).

25.56 trademgen_generateDemand.cpp

```

00001 // //////////////////////////////////////
00002 // Import section
00003 // //////////////////////////////////////
00004 // STL
00005 #include <cassert>
00006 #include <sstream>
00007 #include <fstream>
00008 #include <vector>
00009 #include <list>
00010 #include <string>
00011 // //// Boost (Extended STL) ////
00012 // Boost Tokeniser
00013 #include <boost/tokenizer.hpp>
00014 // Boost Program Options
00015 #include <boost/program_options.hpp>
00016 // Boost Accumulators
00017 #include <boost/accumulators/accumulators.hpp>
00018 #include <boost/accumulators/statistics.hpp>
00019 // Boost Progress
00020 // #include <boost/progress.hpp>
00021 // StdAir
00022 #include <stdair/stdair_basic_types.hpp>
00023 #include <stdair/basic/BasConst_General.hpp>
00024 #include <stdair/basic/ProgressStatusSet.hpp>
00025 #include <stdair/basic/DemandGenerationMethod.hpp>
00026 #include <stdair/bom/EventStruct.hpp>
00027 #include <stdair/bom/BookingRequestStruct.hpp>
00028 #include <stdair/bom/BomDisplay.hpp>
00029 #include <stdair/service/Logger.hpp>
00030 // TraDemGen
00031 #include <trademgen/TRADEMGEN_Service.hpp>
00032 #include <trademgen/config/trademgen-paths.hpp>
00033
00034 // Aliases for namespaces
00035 namespace ba = boost::accumulators;
00036
00037 // ////////// Specific type definitions //////////
00038 typedef unsigned int NbofRuns_T;
00039
00040 typedef ba::accumulator_set<double,
00041                             ba::stats<ba::tag::min, ba::tag::max,
00042                                         ba::tag::mean (ba::immediate),
00043                                         ba::tag::sum,
00044                                         ba::tag::variance> > stat_acc_type;
00045
00046
00047
00048
00049 // ////////// Constants //////////
00050 const stdair::Filename_T K_TRADEMGEN_DEFAULT_LOG_FILENAME ("trademgen_generateDem
and.log");
00051
00052
00053
00054
00055 const stdair::Filename_T K_TRADEMGEN_DEFAULT_INPUT_FILENAME (STDAIR_SAMPLE_DIR
"/demand01.csv");
00056
00057
00058
00059
00060 const stdair::Filename_T K_TRADEMGEN_DEFAULT_OUTPUT_FILENAME ("request.csv");
00061
00062
00063
00064
00065 const stdair::DemandGenerationMethod
00066 K_TRADEMGEN_DEFAULT_DEMAND_GENERATION_METHOD =
00067     stdair::DemandGenerationMethod::POI_PRO;
00068
00069
00070
00071
00072 const char K_TRADEMGEN_DEFAULT_DEMAND_GENERATION_METHOD_CHAR =
00073     K_TRADEMGEN_DEFAULT_DEMAND_GENERATION_METHOD.getMethodAsChar();
00074
00075
00076
00077
00078 const stdair::RandomSeed_T K_TRADEMGEN_DEFAULT_RANDOM_SEED =
00079     stdair::DEFAULT_RANDOM_SEED;
00080
00081
00082
00083
00084 const NbofRuns_T K_TRADEMGEN_DEFAULT_RANDOM_DRAWS = 1;

```

```

00089
00094 const bool K_TRADEMGEN_DEFAULT_BUILT_IN_INPUT = false;
00095
00099 const int K_TRADEMGEN_EARLY_RETURN_STATUS = 99;
00100
00101
00105 void stat_display (std::ostream& oStream, const stat_acc_type& iStatAcc) {
00106
00107     // Store current formatting flags of the output stream
00108     std::ios::fmtflags oldFlags = oStream.flags();
00109
00110     //
00111     oStream.setf (std::ios::fixed);
00112
00113     //
00114     oStream << "Statistics for the demand generation runs: " << std::endl;
00115     oStream << "   minimum   = " << ba::min (iStatAcc) << std::endl;
00116     oStream << "     mean     = " << ba::mean (iStatAcc) << std::endl;
00117     oStream << "   maximum   = " << ba::max (iStatAcc) << std::endl;
00118     oStream << "     count    = " << ba::count (iStatAcc) << std::endl;
00119     oStream << "   variance  = " << ba::variance (iStatAcc) << std::endl;
00120
00121     // Reset formatting flags of output stream
00122     oStream.flags (oldFlags);
00123 }
00124
00125 // /////////// Parsing of Options & Configuration ///////////
00126 // A helper function to simplify the main part.
00127 template<class T> std::ostream& operator<< (std::ostream& os,
00128                                           const std::vector<T>& v) {
00129     std::copy (v.begin(), v.end(), std::ostream_iterator<T> (std::cout, " "));
00130     return os;
00131 }
00132
00136 int readConfiguration (int argc, char* argv[], bool& ioIsBuiltin,
00137                       stdair::RandomSeed_T& ioRandomSeed,
00138                       NbOfRuns_T& ioRandomRuns,
00139                       stdair::Filename_T& ioInputFilename,
00140                       stdair::Filename_T& ioOutputFilename,
00141                       stdair::Filename_T& ioLogFilename,
00142                       stdair::DemandGenerationMethod& ioDemandGenerationMethod)
00143 {
00144     // Demand generation method as a single char (e.g., 'P' or 'S').
00145     char lDemandGenerationMethodChar;
00146
00147     // Default for the built-in input
00148     ioIsBuiltin = K_TRADEMGEN_DEFAULT_BUILT_IN_INPUT;
00149
00150     // Declare a group of options that will be allowed only on command line
00151     boost::program_options::options_description generic ("Generic options");
00152     generic.add_options()
00153         ("prefix", "print installation prefix")
00154         ("version,v", "print version string")
00155         ("help,h", "produce help message");
00156
00157     // Declare a group of options that will be allowed both on command
00158     // line and in config file
00159     boost::program_options::options_description config ("Configuration");
00160     config.add_options()
00161         ("builtin,b",
00162          "The sample BOM tree can be either built-in or parsed from an input file. Th
00163          at latter must then be given with the -i/--input option")
00164         ("seed,s",
00165          boost::program_options::value<stdair::RandomSeed_T>(&ioRandomSeed)->default_
00166          value(K_TRADEMGEN_DEFAULT_RANDOM_SEED),
00167          "Seed for the random generation")

```

```

00166     ("draws,d",
00167     boost::program_options::value<NbOfRuns_T>(&ioRandomRuns)->default_value(K_TRADEMGEN_DEFAULT_RANDOM_DRAWS),
00168     "Number of runs for the demand generations")
00169     ("demandgeneration,G",
00170     boost::program_options::value< char >(&lDemandGenerationMethodChar)->default_value(K_TRADEMGEN_DEFAULT_DEMAND_GENERATION_METHOD_CHAR),
00171     "Method used to generate the demand (i.e., the booking requests): Poisson Process (P) or Order Statistics (S)")
00172     ("input,i",
00173     boost::program_options::value< std::string >(&ioInputFilename)->default_value(K_TRADEMGEN_DEFAULT_INPUT_FILENAME),
00174     "(CSV) input file for the demand distributions")
00175     ("output,o",
00176     boost::program_options::value< std::string >(&ioOutputFilename)->default_value(K_TRADEMGEN_DEFAULT_OUTPUT_FILENAME),
00177     "(CSV) output file for the generated requests")
00178     ("log,l",
00179     boost::program_options::value< std::string >(&ioLogFilename)->default_value(K_TRADEMGEN_DEFAULT_LOG_FILENAME),
00180     "Filepath for the logs")
00181     ;
00182
00183     // Hidden options, will be allowed both on command line and
00184     // in config file, but will not be shown to the user.
00185     boost::program_options::options_description hidden ("Hidden options");
00186     hidden.add_options()
00187         ("copyright",
00188         boost::program_options::value< std::vector<std::string> >(),
00189         "Show the copyright (license)");
00190
00191     boost::program_options::options_description cmdline_options;
00192     cmdline_options.add(generic).add(config).add(hidden);
00193
00194     boost::program_options::options_description config_file_options;
00195     config_file_options.add(config).add(hidden);
00196
00197     boost::program_options::options_description visible ("Allowed options");
00198     visible.add(generic).add(config);
00199
00200     boost::program_options::positional_options_description p;
00201     p.add ("copyright", -1);
00202
00203     boost::program_options::variables_map vm;
00204     boost::program_options::store (boost::program_options::command_line_parser (argc, argv).options (cmdline_options).positional(p).run(), vm);
00205
00206     std::ifstream ifs ("trademgen.cfg");
00207     boost::program_options::store (parse_config_file (ifs, config_file_options), vm);
00208     boost::program_options::notify (vm);
00209
00210     if (vm.count ("help")) {
00211         std::cout << visible << std::endl;
00212         return K_TRADEMGEN_EARLY_RETURN_STATUS;
00213     }
00214
00215     if (vm.count ("version")) {
00216         std::cout << PACKAGE_NAME << ", version " << PACKAGE_VERSION << std::endl;
00217         return K_TRADEMGEN_EARLY_RETURN_STATUS;
00218     }
00219
00220     if (vm.count ("prefix")) {
00221         std::cout << "Installation prefix: " << PREFIXDIR << std::endl;
00222         return K_TRADEMGEN_EARLY_RETURN_STATUS;
00223     }
00224
00225
00226

```

```

00227
00228     if (vm.count ("builtin")) {
00229         ioIsBuiltin = true;
00230     }
00231     const std::string isBuiltinStr = (ioIsBuiltin == true)? "yes": "no";
00232     std::cout << "The BOM should be built-in? " << isBuiltinStr << std::endl;
00233
00234     if (ioIsBuiltin == false) {
00235
00236         // The BOM tree should be built from parsing a demand input file
00237         if (vm.count ("input")) {
00238             ioInputFilename = vm["input"].as< std::string >();
00239             std::cout << "Input filename is: " << ioInputFilename << std::endl;
00240
00241         } else {
00242             // The built-in option is not selected. However, no demand input file
00243             // is specified
00244             std::cerr << "Either one among the -b/--builtin and -i/--input "
00245                 << "options must be specified" << std::endl;
00246         }
00247     }
00248
00249     if (vm.count ("output")) {
00250         ioOutputFilename = vm["output"].as< std::string >();
00251         std::cout << "Output filename is: " << ioOutputFilename << std::endl;
00252     }
00253
00254     if (vm.count ("log")) {
00255         ioLogFilename = vm["log"].as< std::string >();
00256         std::cout << "Log filename is: " << ioLogFilename << std::endl;
00257     }
00258
00259     if (vm.count ("demandgeneration")) {
00260         ioDemandGenerationMethod =
00261             stdair::DemandGenerationMethod (lDemandGenerationMethodChar);
00262         std::cout << "Date-time request generation method is: "
00263             << ioDemandGenerationMethod.describe() << std::endl;
00264     }
00265
00266     //
00267     std::cout << "The random generation seed is: " << ioRandomSeed << std::endl;
00268
00269     //
00270     std::cout << "The number of runs is: " << ioRandomRuns << std::endl;
00271
00272     return 0;
00273 }
00274
00275 // //////////////////////////////////////
00276 void generateDemand (TRADEMGEN::TRADEMGEN_Service& ioTrademgenService,
00277                     const stdair::Filename_T& iOutputFilename,
00278                     const NbOfRuns_T& iNbOfRuns,
00279                     const stdair::DemandGenerationMethod& iDemandGenerationMetho
00280 d) {
00281     // Open and clean the .csv output file
00282     std::ofstream output;
00283     output.open (iOutputFilename.c_str());
00284     output.clear();
00285
00286     // Initialise the statistics collector/accumulator
00287     stat_acc_type lStatAccumulator;
00288
00289     // Retrieve the expected (mean value of the) number of events to be
00290     // generated
00291     const stdair::Count_T& lExpectedNbOfEventsToBeGenerated =
00292         ioTrademgenService.getExpectedTotalNumberOfRequestsToBeGenerated();

```

```

00293
00294 // Initialise the (Boost) progress display object
00295 boost::progress_display lProgressDisplay (lExpectedNbOfEventsToBeGenerated
00296                                         * iNbOfRuns);
00297
00298 for (NbOfRuns_T runIdx = 1; runIdx <= iNbOfRuns; ++runIdx) {
00299     // //////////////////////////////////////
00300     output << "Run number: " << runIdx << std::endl;
00301
00302     const stdair::Count_T& lActualNbOfEventsToBeGenerated =
00303         ioTrademgenService.generateFirstRequests (iDemandGenerationMethod);
00304
00305     // DEBUG
00306     STDAIR_LOG_DEBUG ("[" << runIdx << "] Expected: "
00307                     << lExpectedNbOfEventsToBeGenerated << ", actual: "
00308                     << lActualNbOfEventsToBeGenerated);
00309
00310     while (ioTrademgenService.isQueueDone() == false) {
00311
00312         // Extract the next event from the event queue
00313         stdair::EventStruct lEventStruct;
00314         stdair::ProgressStatusSet lProgressStatusSet =
00315             ioTrademgenService.popEvent (lEventStruct);
00316
00317         // DEBUG
00318         // STDAIR_LOG_DEBUG ("[" << runIdx << "] Poped event: '"
00319                             << lEventStruct.describe() << "'");
00320
00321         // Extract the corresponding demand/booking request
00322         const stdair::BookingRequestStruct& lPoppedRequest =
00323             lEventStruct.getBookingRequest();
00324
00325         // DEBUG
00326         STDAIR_LOG_DEBUG ("[" << runIdx << "] Poped booking request: '"
00327                             << lPoppedRequest.describe() << "'");
00328
00329         // Dump the request into the dedicated CSV file
00330         // stdair::BomDisplay::csvDisplay (output, lPoppedRequest);
00331
00332         // Retrieve the corresponding demand stream key
00333         const stdair::DemandGeneratorKey_T& lDemandStreamKey =
00334             lPoppedRequest.getDemandGeneratorKey();
00335
00336         // Assess whether more events should be generated for that demand stream
00337         const bool stillHavingRequestsToBeGenerated = ioTrademgenService.
00338             stillHavingRequestsToBeGenerated (lDemandStreamKey,
00339                                             lProgressStatusSet,
00340                                             iDemandGenerationMethod);
00341
00342         // DEBUG
00343         STDAIR_LOG_DEBUG (lProgressStatusSet.describe());
00344         STDAIR_LOG_DEBUG ("=> [" << lDemandStreamKey << "] is now processed. "
00345                             << "Still generate events for that demand stream? "
00346                             << stillHavingRequestsToBeGenerated);
00347
00348         // If there are still events to be generated for that demand stream,
00349         // generate and add them to the event queue
00350         if (stillHavingRequestsToBeGenerated == true) {
00351
00352             stdair::BookingRequestPtr_T lNextRequest_ptr =
00353                 ioTrademgenService.generateNextRequest (lDemandStreamKey,
00354                                                         iDemandGenerationMethod);
00355
00356             assert (lNextRequest_ptr != NULL);
00357
00358             // Sanity check
00359             const stdair::Duration_T lDuration =
00360

```



```

00371         lNextRequest_ptr->getRequestDateTime()
00372         - lPoppedRequest.getRequestDateTime();
00373     if (lDuration.total_milliseconds() < 0) {
00374         STDAIR_LOG_ERROR ("[" << lDemandStreamKey
00375             << "]" The date-time of the generated event ("
00376             << lNextRequest_ptr->getRequestDateTime()
00377             << ") is lower than the date-time "
00378             << "of the current event ("
00379             << lPoppedRequest.getRequestDateTime() << ")");
00380         assert (false);
00381     }
00382
00383     // DEBUG
00384     STDAIR_LOG_DEBUG ("[" << lDemandStreamKey << "]" Added request: '"
00385         << lNextRequest_ptr->describe()
00386         << "'. Is queue done? "
00387         << ioTrademgenService.isQueueDone());
00388     }
00389     // DEBUG
00390     STDAIR_LOG_DEBUG ("");
00391
00392     // Update the progress display
00393     ++lProgressDisplay;
00394 }
00395
00396 // Add the number of events to the statistics accumulator
00397 lStatAccumulator (lActualNbOfEventsToBeGenerated);
00398
00399 // Reset the service (including the event queue) for the next run
00400 ioTrademgenService.reset();
00401 }
00402
00403 // DEBUG
00404 STDAIR_LOG_DEBUG ("End of the demand generation. Following are some "
00405     "statistics for the " << iNbOfRuns << " runs.");
00406 std::ostringstream oStatStr;
00407 stat_display (oStatStr, lStatAccumulator);
00408 STDAIR_LOG_DEBUG (oStatStr.str());
00409
00410 // DEBUG
00411 const std::string& lBOMStr = ioTrademgenService.csvDisplay();
00412 STDAIR_LOG_DEBUG (lBOMStr);
00413
00414 // Close the output file
00415 output.close();
00416 }
00417
00418
00419 // ////////////////////////////////// M A I N //////////////////////////////////
00420 int main (int argc, char* argv[]) {
00421
00422     // State whether the BOM tree should be built-in or parsed from an input file
00423     bool isBuiltin;
00424
00425     // Random generation seed
00426     stdair::RandomSeed_T lRandomSeed;
00427
00428     // Number of random draws to be generated (best if greater than 100)
00429     NbOfRuns_T lNbOfRuns;
00430
00431     // Input file name
00432     stdair::Filename_T lInputFilename;
00433
00434     // Output file name
00435     stdair::Filename_T lOutputFilename;
00436
00437     // Output log File

```

```
00438     stdair::Filename_T lLogFilename;
00439
00440     // Demand generation method.
00441     stdair::DemandGenerationMethod
00442         lDemandGenerationMethod (K_TRADEMGEN_DEFAULT_DEMAND_GENERATION_METHOD);
00443
00444     // Call the command-line option parser
00445     const int lOptionParserStatus =
00446         readConfiguration (argc, argv, isBuiltin, lRandomSeed, lNbOfRuns,
00447             lInputFilename, lOutputFilename, lLogFilename,
00448             lDemandGenerationMethod);
00449
00450     if (lOptionParserStatus == K_TRADEMGEN_EARLY_RETURN_STATUS) {
00451         return 0;
00452     }
00453
00454     // Set the log parameters
00455     std::ofstream logOutputFile;
00456     // Open and clean the log outputfile
00457     logOutputFile.open (lLogFilename.c_str());
00458     logOutputFile.clear();
00459
00460     // Set up the log parameters
00461     const stdair::BasLogParams lLogParams (stdair::LOG::DEBUG, logOutputFile);
00462
00463     // Initialise the TraDemGen service object
00464     TRADEMGEN::TRADEMGEN_Service trademgenService (lLogParams, lRandomSeed);
00465
00466     // Check wether or not a (CSV) input file should be read
00467     if (isBuiltin == true) {
00468         // Create a sample DemandStream object, and insert it within the BOM tree
00469         trademgenService.buildSampleBom();
00470     } else {
00471         // Create the DemandStream objects, and insert them within the BOM tree
00472         const TRADEMGEN::DemandFilePath lDemandFilePath (lInputFilename);
00473         trademgenService.parseAndLoad (lDemandFilePath);
00474     }
00475
00476     // Calculate the expected number of events to be generated.
00477     generateDemand (trademgenService, lOutputFilename, lNbOfRuns,
00478         lDemandGenerationMethod);
00479
00480     // Close the Log outputFile
00481     logOutputFile.close();
00482
00483     /*
00484     \note: as that program is not intended to be run on a server in
00485     production, it is better not to catch the exceptions. When it
00486     happens (that an exception is throwned), that way we get the
00487     call stack.
00488     */
00489
00490     return 0;
00491 }
00492 }
```

25.57 trademgen/batches/trademgen_with_db.cpp File Reference

```
#include <cassert>
#include <iostream>
#include <sstream>
#include <fstream>
#include <vector>
#include <string>
#include <boost/date_time/posix_time/posix_time.hpp>
#include <boost/date_time/gregorian/gregorian.hpp>
#include <boost/tokenizer.hpp>
#include <boost/program_options.hpp>
#include <stdair/stdair_basic_types.hpp>
#include <stdair/basic/BasConst_General.hpp>
#include <stdair/basic/BasDBParams.hpp>
#include <stdair/basic/BasLogParams.hpp>
#include <trademgen/TRADEMGEN_Service.hpp>
#include <trademgen/config/trademgen-paths.hpp>
```

Typedefs

- typedef std::vector< std::string > [WordList_T](#)

Functions

- const std::string [K_TRADEMGEN_DEFAULT_LOG_FILENAME](#) ("trademgen_with_db.log")
- const std::string [K_TRADEMGEN_DEFAULT_INPUT_FILENAME](#) (STDAIR_SAMPLE_DIR"/demand01.csv")
- const std::string [K_TRADEMGEN_DEFAULT_QUERY_STRING](#) ("my good old query")
- const std::string [K_TRADEMGEN_DEFAULT_DB_USER](#) ("dsim")
- const std::string [K_TRADEMGEN_DEFAULT_DB_PASSWD](#) ("dsim")
- const std::string [K_TRADEMGEN_DEFAULT_DB_DBNAME](#) ("sim_dsim")
- const std::string [K_TRADEMGEN_DEFAULT_DB_HOST](#) ("localhost")
- const std::string [K_TRADEMGEN_DEFAULT_DB_PORT](#) ("3306")
- void [tokeniseStringIntoWordList](#) (const std::string &iPhrase, [WordList_T](#) &ioWordList)
- std::string [createStringFromWordList](#) (const [WordList_T](#) &iWordList)
- template<class T >
std::ostream & [operator<<](#) (std::ostream &os, const std::vector< T > &v)
- int [readConfiguration](#) (int argc, char *argv[], bool &ioIsBuiltin, stdair::RandomSeed_T &ioRandomSeed, std::string &ioQueryString, stdair::Filename_T &ioInputFilename, std::string &ioLogFilename, std::string &ioDBUser, std::string &ioDBPasswd, std::string &ioDBHost, std::string &ioDBPort, std::string &ioDBDBName)
- int [main](#) (int argc, char *argv[])

Variables

- const bool [K_TRADEMGEN_DEFAULT_BUILT_IN_INPUT](#) = false
- const stdair::RandomSeed_T [K_TRADEMGEN_DEFAULT_RANDOM_SEED](#)
- const int [K_TRADEMGEN_EARLY_RETURN_STATUS](#) = 99

25.57.1 Typedef Documentation

25.57.1.1 typedef std::vector<std::string> WordList_T

Definition at line 24 of file [trademgen_with_db.cpp](#).

25.57.2 Function Documentation

25.57.2.1 const std::string K_TRADEMGEN_DEFAULT_LOG_FILENAME ("trademgen_with_db.log")

Default name and location for the log file.

25.57.2.2 const std::string K_TRADEMGEN_DEFAULT_INPUT_FILENAME (STDAIR_SAMPLE_DIR"/demand01.csv")

Default name and location for the (CSV) input file.

25.57.2.3 const std::string K_TRADEMGEN_DEFAULT_QUERY_STRING ("my good old query")

Default query string.

Referenced by [readConfiguration\(\)](#).

25.57.2.4 const std::string K_TRADEMGEN_DEFAULT_DB_USER ("dsim")

Default parameters for the database connection.

Referenced by [readConfiguration\(\)](#).

25.57.2.5 const std::string K_TRADEMGEN_DEFAULT_DB_PASSWD ("dsim")

Referenced by [readConfiguration\(\)](#).

25.57.2.6 const std::string K_TRADEMGEN_DEFAULT_DB_DBNAME ("sim_dsim")

Referenced by [readConfiguration\(\)](#).

25.57.2.7 `const std::string K_TRADEMGEN_DEFAULT_DB_HOST ("localhost")`

Referenced by [readConfiguration\(\)](#).

25.57.2.8 `const std::string K_TRADEMGEN_DEFAULT_DB_PORT ("3306")`

Referenced by [readConfiguration\(\)](#).

25.57.2.9 `void tokeniseStringIntoWordList (const std::string & iPhrase, WordList_T & ioWordList)`

Definition at line 67 of file [trademgen_with_db.cpp](#).

Referenced by [readConfiguration\(\)](#).

25.57.2.10 `std::string createStringFromWordList (const WordList_T & iWordList)`

Definition at line 89 of file [trademgen_with_db.cpp](#).

Referenced by [readConfiguration\(\)](#).

25.57.2.11 `template<class T > std::ostream& operator<< (std::ostream & os, const std::vector< T > & v) [inline]`

Definition at line 108 of file [trademgen_with_db.cpp](#).

25.57.2.12 `int readConfiguration (int argc, char * argv[], bool & ioIsBuiltin, stdair::RandomSeed_T & ioRandomSeed, std::string & ioQueryString, stdair::Filename_T & ioInputFilename, std::string & ioLogFilename, std::string & ioDBUser, std::string & ioDBPasswd, std::string & ioDBHost, std::string & ioDBPort, std::string & ioDBDBName)`

Read and parse the command line options.

Definition at line 118 of file [trademgen_with_db.cpp](#).

References [createStringFromWordList\(\)](#), [K_TRADEMGEN_DEFAULT_BUILT_IN_INPUT](#), [K_TRADEMGEN_DEFAULT_DB_DBNAME\(\)](#), [K_TRADEMGEN_DEFAULT_DB_HOST\(\)](#), [K_TRADEMGEN_DEFAULT_DB_PASSWD\(\)](#), [K_TRADEMGEN_DEFAULT_DB_PORT\(\)](#), [K_TRADEMGEN_DEFAULT_DB_USER\(\)](#), [K_TRADEMGEN_DEFAULT_INPUT_FILENAME\(\)](#), [K_TRADEMGEN_DEFAULT_LOG_FILENAME\(\)](#), [K_TRADEMGEN_DEFAULT_QUERY_STRING\(\)](#), [K_TRADEMGEN_DEFAULT_RANDOM_SEED](#), [K_TRADEMGEN_EARLY_RETURN_STATUS](#), and [tokeniseStringIntoWordList\(\)](#).

25.57.2.13 `int main (int argc, char * argv[])`

Definition at line 288 of file `trademgen_with_db.cpp`.

References `TRADEMGEN::TRADEMGEN_Service::buildSampleBom()`, `TRADEMGEN::TRADEMGEN_Service::displayAirlineListFromDB()`, `K_TRADEMGEN_EARLY_RETURN_STATUS`, `TRADEMGEN::TRADEMGEN_Service::parseAndLoad()`, and `readConfiguration()`.

25.57.3 Variable Documentation

25.57.3.1 `const bool K_TRADEMGEN_DEFAULT_BUILT_IN_INPUT = false`

Default for the input type. It can be either built-in or provided by an input file. That latter must then be given with the `-i` option.

Definition at line 43 of file `trademgen_with_db.cpp`.

25.57.3.2 `const stdair::RandomSeed_T K_TRADEMGEN_DEFAULT_RANDOM_SEED`

Initial value:

```
stdair::DEFAULT_RANDOM_SEED
```

Default random generation seed (e.g., 120765987).

Definition at line 48 of file `trademgen_with_db.cpp`.

25.57.3.3 `const int K_TRADEMGEN_EARLY_RETURN_STATUS = 99`

Early return status (so that it can be differentiated from an error).

Definition at line 115 of file `trademgen_with_db.cpp`.

25.58 trademgen_with_db.cpp

```

00001 // STL
00002 #include <cassert>
00003 #include <iostream>
00004 #include <sstream>
00005 #include <fstream>
00006 #include <vector>
00007 #include <string>
00008 // Boost (Extended STL)
00009 #include <boost/date_time/posix_time/posix_time.hpp>
00010 #include <boost/date_time/gregorian/gregorian.hpp>
00011 #include <boost/tokenizer.hpp>
00012 #include <boost/program_options.hpp>
00013 // StdAir
00014 #include <stdair/stdair_basic_types.hpp>
00015 #include <stdair/basic/BasConst_General.hpp>
00016 #include <stdair/basic/BasDBParams.hpp>
00017 #include <stdair/basic/BasLogParams.hpp>
00018 // TraDemGen
00019 #include <trademgen/TRADEMGEN_Service.hpp>
00020 #include <trademgen/config/trademgen-paths.hpp>
00021
00022
00023 // ////////// Type definitions //////////
00024 typedef std::vector<std::string> WordList_T;
00025
00026
00027 // ////////// Constants //////////
00031 const std::string K_TRADEMGEN_DEFAULT_LOG_FILENAME ("trademgen_with_db.log");
00032
00036 const std::string K_TRADEMGEN_DEFAULT_INPUT_FILENAME (STDAIR_SAMPLE_DIR
00037                                                         "/demand01.csv");
00038
00043 const bool K_TRADEMGEN_DEFAULT_BUILT_IN_INPUT = false;
00044
00048 const stdair::RandomSeed_T K_TRADEMGEN_DEFAULT_RANDOM_SEED =
00049     stdair::DEFAULT_RANDOM_SEED;
00050
00054 const std::string K_TRADEMGEN_DEFAULT_QUERY_STRING ("my good old query");
00055
00059 const std::string K_TRADEMGEN_DEFAULT_DB_USER ("dsim");
00060 const std::string K_TRADEMGEN_DEFAULT_DB_PASSWD ("dsim");
00061 const std::string K_TRADEMGEN_DEFAULT_DB_DBNAME ("sim_dsim");
00062 const std::string K_TRADEMGEN_DEFAULT_DB_HOST ("localhost");
00063 const std::string K_TRADEMGEN_DEFAULT_DB_PORT ("3306");
00064
00065
00066 // //////////////////////////////////////
00067 void tokeniseStringIntoWordList (const std::string& iPhrase,
00068                                 WordList_T& ioWordList) {
00069     // Empty the word list
00070     ioWordList.clear();
00071
00072     // Boost Tokeniser
00073     typedef boost::tokenizer<boost::char_separator<char> > Tokeniser_T;
00074
00075     // Define the separators
00076     const boost::char_separator<char> lSeparatorList (" .,:|+-*/_=#$%'^~^&(){}[]?'<>
00077 \");
00078
00079     // Initialise the phrase to be tokenised
00080     Tokeniser_T lTokens (iPhrase, lSeparatorList);
00081     for (Tokeniser_T::const_iterator tok_iter = lTokens.begin();
00082          tok_iter != lTokens.end(); ++tok_iter) {
00082         const std::string& lTerm = *tok_iter;
00083         ioWordList.push_back (lTerm);

```

```

00084     }
00085
00086 }
00087
00088 // //////////////////////////////////////
00089 std::string createStringFromWordList (const WordList_T& iWordList) {
00090     std::ostringstream ostr;
00091
00092     unsigned short idx = iWordList.size();
00093     for (WordList_T::const_iterator itWord = iWordList.begin();
00094          itWord != iWordList.end(); ++itWord, --idx) {
00095         const std::string& lWord = *itWord;
00096         ostr << lWord;
00097         if (idx > 1) {
00098             ostr << " ";
00099         }
00100     }
00101
00102     return ostr.str();
00103 }
00104
00105
00106 // ////////////////////////////////// Parsing of Options & Configuration //////////////////////////////////
00107 // A helper function to simplify the main part.
00108 template<class T> std::ostream& operator<< (std::ostream& os,
00109                                           const std::vector<T>& v) {
00110     std::copy (v.begin(), v.end(), std::ostream_iterator<T> (std::cout, " "));
00111     return os;
00112 }
00113
00115 const int K_TRADEMGEN_EARLY_RETURN_STATUS = 99;
00116
00118 int readConfiguration (int argc, char* argv[], bool& ioIsBuiltin,
00119                       stdair::RandomSeed_T& ioRandomSeed,
00120                       std::string& ioQueryString,
00121                       stdair::Filename_T& ioInputFilename,
00122                       std::string& ioLogFilename,
00123                       std::string& ioDBUser, std::string& ioDBPasswd,
00124                       std::string& ioDBHost, std::string& ioDBPort,
00125                       std::string& ioDBDBName) {
00126
00127     // Default for the built-in input
00128     ioIsBuiltin = K_TRADEMGEN_DEFAULT_BUILT_IN_INPUT;
00129
00130     // Initialise the travel query string, if that one is empty
00131     if (ioQueryString.empty() == true) {
00132         ioQueryString = K_TRADEMGEN_DEFAULT_QUERY_STRING;
00133     }
00134
00135     // Transform the query string into a list of words (STL strings)
00136     WordList_T lWordList;
00137     tokeniseStringIntoWordList (ioQueryString, lWordList);
00138
00139     // Declare a group of options that will be allowed only on command line
00140     boost::program_options::options_description generic ("Generic options");
00141     generic.add_options()
00142         ("prefix", "print installation prefix")
00143         ("version,v", "print version string")
00144         ("help,h", "produce help message");
00145
00146     // Declare a group of options that will be allowed both on command
00147     // line and in config file
00148     boost::program_options::options_description config ("Configuration");
00149     config.add_options()
00150         ("builtin,b",
00151          "The sample BOM tree can be either built-in or parsed from an input file. Th
00152          at latter must then be given with the -i/--input option")

```



```

00152     ("seed,s",
00153     boost::program_options::value<std::RandomSeed_T>(&ioRandomSeed)->default_
value(K_TRADEMGEN_DEFAULT_RANDOM_SEED),
00154     "Seed for the random generation")
00155     ("input,i",
00156     boost::program_options::value< std::string >(&ioInputFilename)->default_valu
e(K_TRADEMGEN_DEFAULT_INPUT_FILENAME),
00157     "(CVS) input file for the demand distributions")
00158     ("log,l",
00159     boost::program_options::value< std::string >(&ioLogFilename)->default_value(
K_TRADEMGEN_DEFAULT_LOG_FILENAME),
00160     "Filepath for the logs")
00161     ("user,u",
00162     boost::program_options::value< std::string >(&ioDBUser)->default_value(
K_TRADEMGEN_DEFAULT_DB_USER),
00163     "SQL database user (e.g., dsim)")
00164     ("passwd,p",
00165     boost::program_options::value< std::string >(&ioDBPasswd)->default_value(
K_TRADEMGEN_DEFAULT_DB_PASSWD),
00166     "SQL database password (e.g., dsim)")
00167     ("host,H",
00168     boost::program_options::value< std::string >(&ioDBHost)->default_value(
K_TRADEMGEN_DEFAULT_DB_HOST),
00169     "SQL database hostname (e.g., localhost)")
00170     ("port,P",
00171     boost::program_options::value< std::string >(&ioDBPort)->default_value(
K_TRADEMGEN_DEFAULT_DB_PORT),
00172     "SQL database port (e.g., 3306)")
00173     ("dbname,m",
00174     boost::program_options::value< std::string >(&ioDBDBName)->default_value(
K_TRADEMGEN_DEFAULT_DB_DBNAME),
00175     "SQL database name (e.g., sim_dsim)")
00176     ("query,q",
00177     boost::program_options::value< WordList_T >(&lWordList)->multitoken(),
00178     "Query word list")
00179     ;
00180
00181     // Hidden options, will be allowed both on command line and
00182     // in config file, but will not be shown to the user.
00183     boost::program_options::options_description hidden ("Hidden options");
00184     hidden.add_options()
00185         ("copyright",
00186         boost::program_options::value< std::vector<std::string> >(),
00187         "Show the copyright (license)");
00188
00189     boost::program_options::options_description cmdline_options;
00190     cmdline_options.add(generic).add(config).add(hidden);
00191
00192     boost::program_options::options_description config_file_options;
00193     config_file_options.add(config).add(hidden);
00194
00195     boost::program_options::options_description visible ("Allowed options");
00196     visible.add(generic).add(config);
00197
00198     boost::program_options::positional_options_description p;
00199     p.add ("copyright", -1);
00200
00201     boost::program_options::variables_map vm;
00202     boost::program_options::
00203     store (boost::program_options::command_line_parser (argc, argv).
00204     options (cmdline_options).positional(p).run(), vm);
00205
00206     std::ifstream ifs ("trademgen_with_db.cfg");
00207     boost::program_options::store (parse_config_file (ifs, config_file_options),
00208     vm);
00209     boost::program_options::notify (vm);
00210

```

```
00211     if (vm.count ("help")) {
00212         std::cout << visible << std::endl;
00213         return K_TRADEMGEN_EARLY_RETURN_STATUS;
00214     }
00215
00216     if (vm.count ("version")) {
00217         std::cout << PACKAGE_NAME << ", version " << PACKAGE_VERSION << std::endl;
00218         return K_TRADEMGEN_EARLY_RETURN_STATUS;
00219     }
00220
00221     if (vm.count ("prefix")) {
00222         std::cout << "Installation prefix: " << PREFIXDIR << std::endl;
00223         return K_TRADEMGEN_EARLY_RETURN_STATUS;
00224     }
00225
00226     if (vm.count ("builtin")) {
00227         ioIsBuiltin = true;
00228     }
00229     const std::string isBuiltinStr = (ioIsBuiltin == true)?"yes":"no";
00230     std::cout << "The BOM should be built-in? " << isBuiltinStr << std::endl;
00231
00232     if (ioIsBuiltin == false) {
00233
00234         // The BOM tree should be built from parsing a demand input file
00235         if (vm.count ("input")) {
00236             ioInputFilename = vm["input"].as< std::string >();
00237             std::cout << "Input filename is: " << ioInputFilename << std::endl;
00238         }
00239         else {
00240             // The built-in option is not selected. However, no demand input file
00241             // is specified
00242             std::cerr << "Either one among the -b/--builtin and -i/--input "
00243                 << "options must be specified" << std::endl;
00244         }
00245     }
00246
00247     if (vm.count ("log")) {
00248         ioLogFilename = vm["log"].as< std::string >();
00249         std::cout << "Log filename is: " << ioLogFilename << std::endl;
00250     }
00251
00252     if (vm.count ("user")) {
00253         ioDBUser = vm["user"].as< std::string >();
00254         std::cout << "SQL database user name is: " << ioDBUser << std::endl;
00255     }
00256
00257     if (vm.count ("passwd")) {
00258         ioDBPasswd = vm["passwd"].as< std::string >();
00259         // std::cout << "SQL database user password is: " << ioDBPasswd << std::endl;
00260     }
00261
00262     if (vm.count ("host")) {
00263         ioDBHost = vm["host"].as< std::string >();
00264         std::cout << "SQL database host name is: " << ioDBHost << std::endl;
00265     }
00266
00267     if (vm.count ("port")) {
00268         ioDBPort = vm["port"].as< std::string >();
00269         std::cout << "SQL database port number is: " << ioDBPort << std::endl;
00270     }
00271
00272     if (vm.count ("dbname")) {
00273         ioDBDBName = vm["dbname"].as< std::string >();
00274         std::cout << "SQL database name is: " << ioDBDBName << std::endl;
00275     }
00276
```

```

00277 //
00278 std::cout << "The random generation seed is: " << ioRandomSeed << std::endl;
00279
00280 ioQueryString = createStringFromWordList (lWordList);
00281 std::cout << "The query string is: " << ioQueryString << std::endl;
00282
00283 return 0;
00284 }
00285
00286
00287 // ////////////////////////////////// M A I N //////////////////////////////////
00288 int main (int argc, char* argv[]) {
00289
00290     // State whether the BOM tree should be built-in or parsed from an input file
00291     bool isBuiltin;
00292
00293     // Random generation seed
00294     stdair::RandomSeed_T lRandomSeed;
00295
00296     // Query
00297     std::string lQuery;
00298
00299     // Input file name
00300     stdair::Filename_T lInputFilename;
00301
00302     // Output log File
00303     std::string lLogFilename;
00304
00305     // SQL database parameters
00306     std::string lDBUser;
00307     std::string lDBPasswd;
00308     std::string lDBHost;
00309     std::string lDBPort;
00310     std::string lDBDBName;
00311
00312     // Airline code
00313     const stdair::AirlineCode_T lAirlineCode ("BA");
00314
00315     // Call the command-line option parser
00316     const int lOptionParserStatus =
00317         readConfiguration (argc, argv, isBuiltin, lRandomSeed, lQuery,
00318             lInputFilename, lLogFilename,
00319             lDBUser, lDBPasswd, lDBHost, lDBPort, lDBDBName);
00320
00321     if (lOptionParserStatus == K_TRADEMGEN_EARLY_RETURN_STATUS) {
00322         return 0;
00323     }
00324
00325     // Set the database parameters
00326     stdair::BasDBParams lDBParams (lDBUser, lDBPasswd, lDBHost, lDBPort,
00327         lDBDBName);
00328
00329     // Set the log parameters
00330     std::ofstream logOutputFile;
00331     // open and clean the log outputfile
00332     logOutputFile.open (lLogFilename.c_str());
00333     logOutputFile.clear();
00334
00335     // Set up the log parameters
00336     const stdair::BasLogParams lLogParams (stdair::LOG::DEBUG, logOutputFile);
00337
00338     // Initialise the TraDemGen service object
00339     TRADEMGEN::TRADEMGEN_Service trademgenService (lLogParams, lDBParams,
00340         lRandomSeed);
00341
00342     // Check wether or not a (CSV) input file should be read
00343     if (isBuiltin == true) {

```

```
00344     // Create a sample DemandStream object, and insert it within the BOM tree
00345     trademgenService.buildSampleBom();
00346
00347 } else {
00348     // Create the DemandStream objects, and insert them within the BOM tree
00349     const TRADEMGEN::DemandFilePath lDemandFilePath (lInputFilename);
00350     trademgenService.parseAndLoad (lDemandFilePath);
00351 }
00352
00353 // Query the database
00354 trademgenService.displayAirlineListFromDB();
00355
00356 // Close the Log outputFile
00357 logOutputFile.close();
00358
00359 return 0;
00360 }
```

25.59 trademgen/bom/BomDisplay.cpp File Reference

```
#include <cassert>
#include <ostream>
#include <stdair/basic/BasConst_BomDisplay.hpp>
#include <stdair/bom/BomManager.hpp>
#include <sevmgr/SEVMGR_Service.hpp>
#include <sevmgr/SEVMGR_Types.hpp>
#include <trademgen/bom/DemandStream.hpp>
#include <trademgen/bom/BomDisplay.hpp>
```

Classes

- struct [TRADEMGEN::FlagSaver](#)

Namespaces

- namespace [TRADEMGEN](#)

25.60 BomDisplay.cpp

```

00001 // //////////////////////////////////////
00002 // Import section
00003 // //////////////////////////////////////
00004 // STL
00005 #include <cassert>
00006 #include <ostream>
00007 // StdAir
00008 #include <stdair/basic/BasConst_BomDisplay.hpp>
00009 #include <stdair/bom/BomManager.hpp>
00010 // SEVMgr
00011 #include <sevmgr/SEVMGR_Service.hpp>
00012 #include <sevmgr/SEVMGR_Types.hpp>
00013 // TraDemGen
00014 #include <trademgen/bom/DemandStream.hpp>
00015 #include <trademgen/bom/BomDisplay.hpp>
00016
00017 namespace TRADEMGEN {
00018
00024     struct FlagSaver {
00025     public:
00027         FlagSaver (std::ostream& oStream)
00028             : _oStream (oStream), _streamFlags (oStream.flags()) {
00029         }
00030
00032         ~FlagSaver() {
00033             // Reset formatting flags of the given output stream
00034             _oStream.flags (_streamFlags);
00035         }
00036
00037     private:
00039         std::ostream& _oStream;
00041         std::ios::fmtflags _streamFlags;
00042     };
00043
00044 // //////////////////////////////////////
00045     std::string BomDisplay::csvDisplay (const SEVMGR::SEVMGR_ServicePtr_T iSEVMGR_S
ervicePtr) {
00046         std::ostringstream oStream;
00047
00048         //
00049         assert (iSEVMGR_ServicePtr != NULL);
00050
00054         oStream << std::endl;
00055         oStream << "=====
00056             << std::endl;
00057         oStream << "EventQueue: " << iSEVMGR_ServicePtr->describeKey() << std::endl;
00058         oStream << "=====
00059             << std::endl;
00060
00061         // Check whether there are DemandStream objects
00062         const bool hasEventGeneratorList =
00063             iSEVMGR_ServicePtr->hasEventGeneratorList<DemandStream>();
00064         if (hasEventGeneratorList == false) {
00065             return oStream.str();
00066         }
00067
00068         // Retrieve the DemandStream list
00069         const DemandStreamList_T& lDemandStreamList =
00070             iSEVMGR_ServicePtr->getEventGeneratorList<DemandStream>();
00071
00072         // Browse the inventories
00073         for (DemandStreamList_T::const_iterator itDemandStream =
00074             lDemandStreamList.begin();
00075             itDemandStream != lDemandStreamList.end(); ++itDemandStream) {
00076             DemandStream* lDemandStream_ptr = *itDemandStream;

```

```
00077         assert (lDemandStream_ptr != NULL);
00078
00079         // Display the demand stream
00080         csvDisplay (oStream, *lDemandStream_ptr);
00081     }
00082
00083     return oStream.str();
00084 }
00085
00086 // //////////////////////////////////////
00087 void BomDisplay::csvDisplay (std::ostream& oStream,
00088                             const DemandStream& iDemandStream) {
00089     // Save the formatting flags for the given STL output stream
00090     FlagSaver flagSaver (oStream);
00091
00092     oStream << "+++++" << std::endl;
00093     oStream << iDemandStream.display();
00094     oStream << "+++++" << std::endl;
00095 }
00096
00097 }
```

25.61 trademgen/bom/BomDisplay.hpp File Reference

```
#include <iosfwd>
#include <string>
#include <sevmgr/SEVMGR_Types.hpp>
```

Classes

- class [TRADEMGEN::BomDisplay](#)
Utility class to display TraDemGen objects with a pretty format.

Namespaces

- namespace [TRADEMGEN](#)

25.62 BomDisplay.hpp

```
00001 #ifndef __TRADEMGEN_BOM_BOMDISPLAY_HPP
00002 #define __TRADEMGEN_BOM_BOMDISPLAY_HPP
00003
00004 // //////////////////////////////////////
00005 // Import section
00006 // //////////////////////////////////////
00007 // STL
00008 #include <iosfwd>
00009 #include <string>
00010 // SEvMgr
00011 #include <sevmgr/SEVMGR_Types.hpp>
00012
00013
00014 namespace TRADEMGEN {
00015
00017     class DemandStream;
00018
00023     class BomDisplay {
00024     public:
00025         // ////////////////////////////////// Display support methods //////////////////////////////////
00036         static std::string csvDisplay (const SEVMGR::SEVMGR_ServicePtr_T);
00037
00046         static void csvDisplay (std::ostream&, const DemandStream&);
00047     };
00048
00049 }
00050 #endif // __TRADEMGEN_BOM_BOMDISPLAY_HPP
```

25.63 trademgen/bom/DemandStream.cpp File Reference

```
#include <cassert>
#include <sstream>
#include <cmath>
#include <iomanip>
#include <boost/make_shared.hpp>
#include <stdair/basic/BasConst_General.hpp>
#include <stdair/basic/BasConst_Inventory.hpp>
#include <stdair/basic/BasConst_Request.hpp>
#include <stdair/bom/BookingRequestStruct.hpp>
#include <stdair/service/Logger.hpp>
#include <trademgen/basic/BasConst_DemandGeneration.hpp>
#include <trademgen/bom/DemandStream.hpp>
```

Namespaces

- namespace [TRADEMGEN](#)

25.64 DemandStream.cpp

```

00001 // //////////////////////////////////////
00002 // Import section
00003 // //////////////////////////////////////
00004 // STL
00005 #include <cassert>
00006 #include <sstream>
00007 #include <cmath>
00008 #include <iomanip>
00009 // Boost
00010 #include <boost/make_shared.hpp>
00011 // StdAir
00012 #include <stdair/basic/BasConst_General.hpp>
00013 #include <stdair/basic/BasConst_Inventory.hpp>
00014 #include <stdair/basic/BasConst_Request.hpp>
00015 #include <stdair/bom/BookingRequestStruct.hpp>
00016 #include <stdair/service/Logger.hpp>
00017 // TraDemGen
00018 #include <trademgen/basic/BasConst_DemandGeneration.hpp>
00019 #include <trademgen/bom/DemandStream.hpp>
00020
00021 namespace TRADEMGEN {
00022
00023     // //////////////////////////////////////
00024     DemandStream::DemandStream()
00025     : _key (stdair::DEFAULT_ORIGIN, stdair::DEFAULT_DESTINATION,
00026            stdair::DEFAULT_DEPARTURE_DATE, stdair::DEFAULT_CABIN_CODE),
00027       _parent (NULL),
00028       _demandCharacteristics (ArrivalPatternCumulativeDistribution_T(),
00029                              POSProbabilityMassFunction_T(),
00030                              ChannelProbabilityMassFunction_T(),
00031                              TripTypeProbabilityMassFunction_T(),
00032                              StayDurationProbabilityMassFunction_T(),
00033                              FrequentFlyerProbabilityMassFunction_T(),
00034                              0.5, 50, 0.5, 50,
00035                              PreferredDepartureTimeContinuousDistribution_T(),
00036                              0.0,
00037                              ValueOfTimeContinuousDistribution_T()),
00038       _posProMass (DEFAULT_POS_PROBALILITY_MASS),
00039       _firstDateTimeRequest (true) {
00040     assert (false);
00041   }
00042
00043     // //////////////////////////////////////
00044     DemandStream::DemandStream (const DemandStream&)
00045     : _key (stdair::DEFAULT_ORIGIN, stdair::DEFAULT_DESTINATION,
00046            stdair::DEFAULT_DEPARTURE_DATE, stdair::DEFAULT_CABIN_CODE),
00047       _parent (NULL),
00048       _demandCharacteristics (ArrivalPatternCumulativeDistribution_T(),
00049                              POSProbabilityMassFunction_T(),
00050                              ChannelProbabilityMassFunction_T(),
00051                              TripTypeProbabilityMassFunction_T(),
00052                              StayDurationProbabilityMassFunction_T(),
00053                              FrequentFlyerProbabilityMassFunction_T(),
00054                              0.5, 50, 0.5, 50,
00055                              PreferredDepartureTimeContinuousDistribution_T(),
00056                              0.0,
00057                              ValueOfTimeContinuousDistribution_T()),
00058       _posProMass (DEFAULT_POS_PROBALILITY_MASS),
00059       _firstDateTimeRequest (true) {
00060     assert (false);
00061   }
00062
00063     // //////////////////////////////////////
00064     DemandStream::DemandStream (const Key_T& iKey) :
00065     _key (iKey) {

```

```

00066     }
00067
00068     // ////////////////////////////////////////
00069     DemandStream::~DemandStream() {
00070     }
00071
00072     // ////////////////////////////////////////
00073     std::string DemandStream::toString() const {
00074         std::ostringstream ostr;
00075         ostr << _key.toString();
00076         return ostr.str();
00077     }
00078
00079     // ////////////////////////////////////////
00080     void DemandStream::
00081     setAll (const ArrivalPatternCumulativeDistribution_T& iArrivalPattern,
00082            const POSProbabilityMassFunction_T& iPOSProbMass,
00083            const ChannelProbabilityMassFunction_T& iChannelProbMass,
00084            const TripTypeProbabilityMassFunction_T& iTripTypeProbMass,
00085            const StayDurationProbabilityMassFunction_T& iStayDurationProbMass,
00086            const FrequentFlyerProbabilityMassFunction_T& iFrequentFlyerProbMass,
00087            const stdair::ChangeFeesRatio_T& iChangeFeeProb,
00088            const stdair::Disutility_T& iChangeFeeDisutility,
00089            const stdair::NonRefundableRatio_T& iNonRefundableProb,
00090            const stdair::Disutility_T& iNonRefundableDisutility,
00091            const PreferredDepartureTimeContinuousDistribution_T& iPreferredDepartu
reTimeContinuousDistribution,
00092            const stdair::WTP_T& iMinWTP,
00093            const ValueOfTimeContinuousDistribution_T& iValueOfTimeContinuousDistri
bution,
00094            const DemandDistribution& iDemandDistribution,
00095            stdair::BaseGenerator_T& ioSharedGenerator,
00096            const stdair::RandomSeed_T& iRequestDateTimeSeed,
00097            const stdair::RandomSeed_T& iDemandCharacteristicsSeed,
00098            const POSProbabilityMass_T& iDefaultPOSProbabilityMass) {
00099
00100         setDemandCharacteristics (iArrivalPattern, iPOSProbMass,
00101                                iChannelProbMass, iTripTypeProbMass,
00102                                iStayDurationProbMass, iFrequentFlyerProbMass,
00103                                iChangeFeeProb, iChangeFeeDisutility,
00104                                iNonRefundableProb, iNonRefundableDisutility,
00105                                iPreferredDepartureTimeContinuousDistribution,
00106                                iMinWTP, iValueOfTimeContinuousDistribution);
00107
00108         setDemandDistribution (iDemandDistribution);
00109         setTotalNumberOfRequestsToBeGenerated (0);
00110         setRequestDateTimeRandomGeneratorSeed (iRequestDateTimeSeed);
00111         setDemandCharacteristicsRandomGeneratorSeed (iDemandCharacteristicsSeed);
00112         setPOSProbabilityMass (iDefaultPOSProbabilityMass);
00113
00114         //
00115         init (ioSharedGenerator);
00116     }
00117
00118     // ////////////////////////////////////////
00119     std::string DemandStream::display() const {
00120         std::ostringstream ostr;
00121
00122         ostr << "Demand stream key: " << _key.toString() << std::endl;
00123
00124         //
00125         ostr << _demandCharacteristics.describe();
00126
00127         //
00128         ostr << _demandDistribution.describe() << " => "
00129              << _totalNumberOfRequestsToBeGenerated << " to be generated"
00130              << std::endl;

```

```

00131
00132     //
00133     oStr << "Random generation context: " << _randomGenerationContext
00134         << std::endl;
00135
00136     //
00137     oStr << "Random generator for date-time: "
00138         << _requestDateTimeRandomGenerator << std::endl;
00139     oStr << "Random generator for demand characteristics: "
00140         << _demandCharacteristicsRandomGenerator << std::endl;
00141
00142     //
00143     oStr << _posProMass.displayProbabilityMass() << std::endl;
00144
00145     return oStr.str();
00146 }
00147
00148 // //////////////////////////////////////
00149 void DemandStream::init (stdair::BaseGenerator_T& ioSharedGenerator) {
00150
00151     // Generate the number of requests
00152     const stdair::RealNumber_T lMu = _demandDistribution._meanNumberOfRequests;
00153     const stdair::RealNumber_T lSigma =
00154         _demandDistribution._stdDevNumberOfRequests;
00155
00156     stdair::NormalDistribution_T lDistrib (lMu, lSigma);
00157     stdair::NormalGenerator_T lNormalGen (ioSharedGenerator, lDistrib);
00158
00159     const stdair::RealNumber_T lRealNumberOfRequestsToBeGenerated = lNormalGen();
00160
00161     const stdair::NbOfRequests_T lIntegerNumberOfRequestsToBeGenerated =
00162         std::floor (lRealNumberOfRequestsToBeGenerated + 0.5);
00163
00164     _totalNumberOfRequestsToBeGenerated = lIntegerNumberOfRequestsToBeGenerated;
00165
00166     _stillHavingRequestsToBeGenerated = true;
00167     _firstDateTimeRequest = true;
00168 }
00169
00170 // //////////////////////////////////////
00171 const bool DemandStream::
00172 stillHavingRequestsToBeGenerated (const stdair::DemandGenerationMethod& iDemand
00173     GenerationMethod) const {
00174
00175     const stdair::DemandGenerationMethod::EN_DemandGenerationMethod& lENDemandGen
00176     erationMethod =
00177         iDemandGenerationMethod.getMethod();
00178     if (lENDemandGenerationMethod == stdair::DemandGenerationMethod::STA_ORD) {
00179         bool hasStillHavingRequestsToBeGenerated = true;
00180
00181         // Check whether enough requests have already been generated
00182         const stdair::Count_T lNbOfRequestsGeneratedSoFar =
00183             _randomGenerationContext.getNumberOfRequestsGeneratedSoFar();
00184
00185         const stdair::Count_T lRemainingNumberOfRequestsToBeGenerated =
00186             _totalNumberOfRequestsToBeGenerated - lNbOfRequestsGeneratedSoFar;
00187
00188         if (lRemainingNumberOfRequestsToBeGenerated <= 0) {
00189             hasStillHavingRequestsToBeGenerated = false;
00190         }
00191
00192         return hasStillHavingRequestsToBeGenerated;
00193     } else {
00194         return _stillHavingRequestsToBeGenerated;
00195     }
00196 }
00197
00198 }
00199
00200

```

```

00196 ///////////////////////////////////////////////////////////////////
00197 const stdair::DateTime_T DemandStream::generateTimeOfRequestPoissonProcess() {
00198
00199     // Prepare arrival pattern.
00200     const ContinuousFloatDuration_T& lArrivalPattern =
00201         _demandCharacteristics._arrivalPattern;
00202
00203     const stdair::Time_T lHardcodedReferenceDepartureTime =
00204         boost::posix_time::hours (8);
00205
00206     // Prepare departure date time.
00207     const stdair::DateTime_T lDepartureDateTime =
00208         boost::posix_time::ptime (_key.getPreferredDepartureDate(),
00209                                 lHardcodedReferenceDepartureTime);
00210
00211     // If no request has been generated so far...
00212     if (_firstDateTimeRequest) {
00213         const stdair::Probability_T lProbabilityFirstRequest = 0;
00214
00215         // Get the lower bound of the arrival pattern (corresponding
00216         // to a cumulative probability of 0).
00217         _dateTimeLastRequest =
00218             lArrivalPattern.getValue (lProbabilityFirstRequest);
00219
00220         _firstDateTimeRequest = false;
00221     }
00222
00223     // Sanity check.
00224     assert (_firstDateTimeRequest == false);
00225
00226     // If the date time of the last request is equal to the lower bound of
00227     // the last daily rate interval (default value is -1, meaning one day
00228     // before departure), we stopped generating request by returning a
00229     // request date time after departure date time.
00230     if (_dateTimeLastRequest == DEFAULT_LAST_LOWER_BOUND_ARRIVAL_PATTERN) {
00231         _stillHavingRequestsToBeGenerated = false;
00232
00233         // Get a positive number of days.
00234         const stdair::Duration_T lDifferenceBetweenDepartureAndThisLowerBound =
00235             convertFloatIntoDuration (-DEFAULT_LAST_LOWER_BOUND_ARRIVAL_PATTERN);
00236
00237         // Calculate a request date-time after the departure date time to end
00238         // the demand generation algorithm.
00239         const stdair::DateTime_T oDateTimeThisRequest =
00240             lDepartureDateTime + lDifferenceBetweenDepartureAndThisLowerBound;
00241
00242         return oDateTimeThisRequest;
00243     }
00244
00245     // Get the upper bound of the current daily rate interval.
00246     stdair::FloatDuration_T lUpperBound =
00247         lArrivalPattern.getUpperBound (_dateTimeLastRequest);
00248
00249     // Compute the daily rate demand.
00250     double lDailyRate = lArrivalPattern.getDerivativeValue(_dateTimeLastRequest);
00251     // Get the expected average number of requests.
00252     const double lDemandMean = _demandDistribution._meanNumberOfRequests;
00253     // Multiply the daily rate by the expected average number of requests.
00254     lDailyRate *= lDemandMean;
00255
00256     // Generate an exponential variable.
00257     const stdair::FloatDuration_T lExponentialVariable =
00258         _requestDateTimeRandomGenerator.generateExponential (lDailyRate);
00259
00260     // Compute the new date time request.
00261     const stdair::FloatDuration_T lDateTimeThisRequest =
00262         _dateTimeLastRequest + lExponentialVariable;

```

```

00263
00264     stdair::DateTime_T oDateTimeThisRequest;
00265
00266     // Verify if this request is in the given daily rate interval.
00267     if (lDateTimeThisRequest < lUpperBound) {
00268
00269         // Conversion.
00270         const stdair::Duration_T lDifferenceBetweenDepartureAndThisRequest =
00271             convertFloatIntoDuration (lDateTimeThisRequest);
00272
00273         // The request date-time is derived from departure date and arrival pattern
00274
00275         oDateTimeThisRequest = lDepartureDateTime
00276             + lDifferenceBetweenDepartureAndThisRequest;
00277
00278         // Remember this date time request.
00279         _dateTimeLastRequest = lDateTimeThisRequest;
00280
00281         // Update the counter of requests generated so far.
00282         incrementGeneratedRequestsCounter();
00283
00284         const double lRefDateTimeThisRequest = lDateTimeThisRequest + double(28800.
001/86400.0);
00285         STDAIR_LOG_NOTIFICATION (boost::gregorian::to_iso_string(_key.
00286             getPreferredDepartureDate()) << " " << std::setprecision(10) << lRefDateTimeThisR
00287             equest);
00288     } else {
00289
00290         // The current request is not in the given daily rate interval.
00291         // Change the daily rate.
00292         _dateTimeLastRequest = lUpperBound;
00293
00294         // Generate a date time request in the new daily rate interval.
00295         oDateTimeThisRequest = generateTimeOfRequestPoissonProcess ();
00296     }
00297
00298     return oDateTimeThisRequest;
00299 }
00300
00301 // //////////////////////////////////////
00302 const stdair::DateTime_T DemandStream::generateTimeOfRequestStatisticsOrder() {
00303
00304     //
00305     // Calculate the result of the formula above step by step.
00306     //
00307
00308     // 1) Get the number of requests generated so far.
00309     //     (equal to k - 1)
00310     const stdair::Count_T& lNbOfRequestsGeneratedSoFar =
00311         _randomGenerationContext.getNumberOfRequestsGeneratedSoFar();
00312
00313     // 2) Deduce the number of requests not generated yet.
00314     //     (equal to n - k + 1)
00315     const stdair::Count_T lRemainingNumberOfRequestsToBeGenerated =
00316         _totalNumberOfRequestsToBeGenerated - lNbOfRequestsGeneratedSoFar;
00317
00318     // Assert that there are still requests to be generated.
00319     assert (lRemainingNumberOfRequestsToBeGenerated > 0);
00320
00321     // 3) Inverse the number of requests not generated yet.
00322     //     1/(n - k + 1)
00323     const double lRemainingRate =
00324         1.0 / static_cast<double> (lRemainingNumberOfRequestsToBeGenerated);
00325
00326     // 4) Get the cumulative probability so far and take its complement.
00327     //     (equal to 1 - x(k-1))

```

```

00341     const stdair::Probability_T& lCumulativeProbabilitySoFar =
00342         _randomGenerationContext.getCumulativeProbabilitySoFar();
00343     const stdair::Probability_T lComplementOfCumulativeProbabilitySoFar =
00344         1.0 - lCumulativeProbabilitySoFar;
00345
00346     // 5) Draw a random variable y and calculate the factor equal to
00347     //     (1 - y)^(1/(n - k + 1)).
00348     const stdair::Probability_T& lVariate = _requestDateTimeRandomGenerator();
00349     double lFactor = std::pow(1.0 - lVariate, lRemainingRate);
00350     if (lFactor >= 1.0 - 1e-6){
00351         lFactor = 1.0 - 1e-6;
00352     }
00353
00354     // 6) Apply the whole formula above to calculate the cumulative probability
00355     //     of the new request.
00356     //     (equal to 1 - (1 - x(k-1)) (1 - y)^(1/(n - k + 1)))
00357     const stdair::Probability_T lCumulativeProbabilityThisRequest =
00358         1.0 - lComplementOfCumulativeProbabilitySoFar * lFactor;
00359
00360     // Now that the cumulative proportion of events generated has been
00361     // calculated, we deduce from the arrival pattern the arrival time of the
00362     // k-th event.
00363     const stdair::FloatDuration_T lNumberOfDaysBetweenDepartureAndThisRequest =
00364         _demandCharacteristics._arrivalPattern.getValue(lCumulativeProbabilityThis
Request);
00365
00366     const stdair::Duration_T lDifferenceBetweenDepartureAndThisRequest =
00367         convertFloatIntoDuration(lNumberOfDaysBetweenDepartureAndThisRequest);
00368
00369     const stdair::Time_T lHardcodedReferenceDepartureTime =
00370         boost::posix_time::hours(8);
00371
00372     const stdair::DateTime_T lDepartureDateTime =
00373         boost::posix_time::ptime(_key.getPreferredDepartureDate(),
lHardcodedReferenceDepartureTime);
00374
00375     // The request date-time is derived from departure date and arrival pattern.
00376     const stdair::DateTime_T oDateTimeThisRequest =
00377         lDepartureDateTime + lDifferenceBetweenDepartureAndThisRequest;
00378
00379     // Update random generation context
00380     _randomGenerationContext.setCumulativeProbabilitySoFar(lCumulativeProbabilit
yThisRequest);
00381
00382     // Update the counter of requests generated so far.
00383     incrementGeneratedRequestsCounter();
00384
00385     // DEBUG
00386     // STDAIR_LOG_DEBUG (lCumulativeProbabilityThisRequest << " "
00387     //                   << lNumberOfDaysBetweenDepartureAndThisRequest);
00388
00389     // NOTIFICATION
00390     double lRefNumberOfDaysBetweenDepartureAndThisRequest =
00391         lNumberOfDaysBetweenDepartureAndThisRequest + double(1.0/3.0);
00392     STDAIR_LOG_NOTIFICATION (boost::gregorian::to_iso_string(_key.
getPreferredDepartureDate()) << " " << std::setprecision(10) << lRefNumberOfDaysB
etweenDepartureAndThisRequest);
00393
00394     return oDateTimeThisRequest;
00395 }
00396
00397 // //////////////////////////////////////
00398
00399 const stdair::Duration_T DemandStream::
00400 convertFloatIntoDuration(const stdair::FloatDuration_T iNumberOfDays) {
00401
00402     // Convert the number of days in number of seconds + number of milliseconds

```



```

00404     const stdair::FloatDuration_T lNumberOfSeconds =
00405         lNumberOfDays * stdair::SECONDS_IN_ONE_DAY;
00406
00407     // Get the number of seconds.
00408     const stdair::IntDuration_T lIntNumberOfSeconds =
00409         std::floor (lNumberOfSeconds);
00410
00411     // Get the number of milliseconds.
00412     const stdair::FloatDuration_T lNumberOfMilliseconds =
00413         (lNumberOfSeconds - lIntNumberOfSeconds)
00414         * stdair::MILLISECONDS_IN_ONE_SECOND;
00415
00416     // +1 is a trick to ensure that the next Event is strictly later
00417     // than the current one
00418     const stdair::IntDuration_T lIntNumberOfMilliseconds =
00419         std::floor (lNumberOfMilliseconds) + 1;
00420
00421     // Convert the number of seconds and milliseconds into a duration.
00422     const stdair::Duration_T lDifferenceBetweenDepartureAndThisRequest =
00423         boost::posix_time::seconds (lIntNumberOfSeconds)
00424         + boost::posix_time::millisec (lIntNumberOfMilliseconds);
00425
00426     return lDifferenceBetweenDepartureAndThisRequest;
00427 }
00428
00429 // //////////////////////////////////////
00430 const stdair::AirportCode_T DemandStream::generatePOS() {
00431
00432     // Generate a random number between 0 and 1.
00433     const stdair::Probability_T& lVariate =
00434         _demandCharacteristicsRandomGenerator();
00435     const stdair::AirportCode_T& oPOS = _demandCharacteristics.getPOSValue (lVariate);
00436
00437     return oPOS;
00438 }
00439
00440 // //////////////////////////////////////
00441 const stdair::ChannelLabel_T DemandStream::generateChannel() {
00442     // Generate a random number between 0 and 1.
00443     const stdair::Probability_T lVariate =
00444         _demandCharacteristicsRandomGenerator();
00445
00446     return _demandCharacteristics._channelProbabilityMass.getValue (lVariate);
00447 }
00448
00449 // //////////////////////////////////////
00450 const stdair::TripType_T DemandStream::generateTripType() {
00451     // Generate a random number between 0 and 1.
00452     const stdair::Probability_T lVariate =
00453         _demandCharacteristicsRandomGenerator();
00454
00455     return _demandCharacteristics._tripTypeProbabilityMass.getValue (lVariate);
00456 }
00457
00458 // //////////////////////////////////////
00459 const stdair::DayDuration_T DemandStream::generateStayDuration() {
00460     // Generate a random number between 0 and 1.
00461     const stdair::Probability_T lVariate =
00462         _demandCharacteristicsRandomGenerator();
00463
00464     return _demandCharacteristics._stayDurationProbabilityMass.getValue (lVariate);
00465 }
00466
00467 // //////////////////////////////////////
00468 const stdair::FrequentFlyer_T DemandStream::generateFrequentFlyer() {

```

```

00468     // Generate a random number between 0 and 1.
00469     const stdair::Probability_T lVariate =
00470         _demandCharacteristicsRandomGenerator();
00471
00472     return _demandCharacteristics._frequentFlyerProbabilityMass.getValue (lVariate);
00473 }
00474
00475 // //////////////////////////////////////
00476 const stdair::ChangeFees_T DemandStream::generateChangeFees() {
00477     // Generate a random number between 0 and 1.
00478     const stdair::Probability_T lVariate =
00479         _demandCharacteristicsRandomGenerator();
00480     if (lVariate < _demandCharacteristics._changeFeeProb) {
00481         return true;
00482     }
00483     return false;
00484 }
00485
00486 // //////////////////////////////////////
00487 const stdair::NonRefundable_T DemandStream::generateNonRefundable() {
00488     // Generate a random number between 0 and 1.
00489     const stdair::Probability_T lVariate =
00490         _demandCharacteristicsRandomGenerator();
00491     if (lVariate < _demandCharacteristics._nonRefundableProb) {
00492         return true;
00493     }
00494     return false;
00495 }
00496
00497 // //////////////////////////////////////
00498 const stdair::Duration_T DemandStream::generatePreferredDepartureTime() {
00499     // Generate a random number between 0 and 1.
00500     const stdair::Probability_T lVariate =
00501         _demandCharacteristicsRandomGenerator();
00502     const stdair::IntDuration_T lNbOfSeconds = _demandCharacteristics.
00503         _preferredDepartureTimeCumulativeDistribution.getValue (lVariate);
00504
00505     const stdair::Duration_T oTime = boost::posix_time::seconds (lNbOfSeconds);
00506
00507     return oTime;
00508 }
00509
00510 // //////////////////////////////////////
00511 const stdair::WTP_T DemandStream::
00512 generateWTP (stdair::RandomGeneration& ioGenerator,
00513             const stdair::Date_T& iDepartureDate,
00514             const stdair::DateTime_T& iDateTimeThisRequest,
00515             const stdair::DayDuration_T& iDurationOfStay) {
00516     const stdair::Date_T lDateThisRequest = iDateTimeThisRequest.date();
00517     const stdair::DateOffset_T lAP = iDepartureDate - lDateThisRequest;
00518     const stdair::DayDuration_T lAPInDays = lAP.days();
00519
00520     stdair::RealNumber_T lProb = -lAPInDays;
00521     stdair::RealNumber_T lFrat5Coef =
00522         _demandCharacteristics._frat5Pattern.getValue (lProb);
00523
00524     const stdair::WTP_T lWTP = _demandCharacteristics._minWTP
00525         * (1.0 + (lFrat5Coef - 1.0) * log(ioGenerator()) / log(0.5));
00526
00527     return lWTP;
00528 }
00529
00530 // //////////////////////////////////////
00531 const stdair::PriceValue_T DemandStream::generateValueOfTime() {
00532     // Generate a random number between 0 and 1.
00533     const stdair::Probability_T lVariate =

```

```

00534     _demandCharacteristicsRandomGenerator();
00535
00536     return _demandCharacteristics._valueOfTimeCumulativeDistribution.getValue (lV
ariate);
00537 }
00538
00539 // //////////////////////////////////////
00540 stdair::BookingRequestPtr_T DemandStream::
00541 generateNextRequest (stdair::RandomGeneration& ioGenerator,
00542                     const stdair::DemandGenerationMethod& iDemandGenerationMet
hod) {
00543
00544     // Origin
00545     const stdair::AirportCode_T& lOrigin = _key.getOrigin();
00546     // Destination
00547     const stdair::AirportCode_T& lDestination = _key.getDestination();
00548     // Preferred departure date
00549     const stdair::Date_T& lPreferredDepartureDate =
00550     _key.getPreferredDepartureDate();
00551     // Preferred cabin
00552     const stdair::CabinCode_T& lPreferredCabin = _key.getPreferredCabin();
00553     // Party size
00554     const stdair::NbOfSeats_T lPartySize = stdair::DEFAULT_PARTY_SIZE;
00555     // POS
00556     const stdair::AirportCode_T lPOS = generatePOS();
00557
00558     // Compute the request date time with the correct algorithm.
00559     stdair::DateTime_T lDateTimeThisRequest;
00560     const stdair::DemandGenerationMethod::EN_DemandGenerationMethod& lENDemandGen
erationMethod =
00561     iDemandGenerationMethod.getMethod();
00562     switch(lENDemandGenerationMethod) {
00563     case stdair::DemandGenerationMethod::POI_PRO:
00564         lDateTimeThisRequest = generateTimeOfRequestPoissonProcess(); break;
00565     case stdair::DemandGenerationMethod::STA_ORD:
00566         lDateTimeThisRequest = generateTimeOfRequestStatisticsOrder(); break;
00567     default: assert (false); break;
00568     }
00569
00570     // Booking channel.
00571     const stdair::ChannelLabel_T lChannelLabel = generateChannel();
00572     // Trip type.
00573     const stdair::TripType_T lTripType = generateTripType();
00574     // Stay duration.
00575     const stdair::DayDuration_T lStayDuration = generateStayDuration();
00576     // Frequent flyer type.
00577     const stdair::FrequentFlyer_T lFrequentFlyer = generateFrequentFlyer();
00578     // Change fees
00579     const stdair::ChangeFees_T lChangeFees = generateChangeFees();
00580     // Change fee disutility
00581     const stdair::Disutility_T lChangeFeeDisutility =
00582     _demandCharacteristics._changeFeeDisutility;
00583     // Non refundable
00584     const stdair::NonRefundable_T lNonRefundable = generateNonRefundable();
00585     // Non refundable disutility
00586     const stdair::Disutility_T lNonRefundableDisutility =
00587     _demandCharacteristics._nonRefundableDisutility;
00588     // Preferred departure time.
00589     const stdair::Duration_T lPreferredDepartureTime =
00590     generatePreferredDepartureTime();
00591     // Value of time
00592     const stdair::PriceValue_T lValueOfTime = generateValueOfTime();
00593     // WTP
00594     const stdair::WTP_T lWTP = generateWTP (ioGenerator,lPreferredDepartureDate,
00595     lDateTimeThisRequest,lStayDuration);
00596
00597     // TODO: move the creation of the structure out of the BOM layer

```

```

00598     // (into the command layer, e.g., within the DemandManager command).
00599
00600     // Create the booking request
00601     stdair::BookingRequestStruct lBookingRequestStruct (describeKey(), lOrigin,
00602                                                         lDestination, lPOS,
00603                                                         lPreferredDepartureDate,
00604                                                         lDateTimeThisRequest,
00605                                                         lPreferredCabin, lPartySize,
00606                                                         lChannelLabel, lTripType,
00607                                                         lStayDuration, lFrequentFlyer,
00608                                                         lPreferredDepartureTime,
00609                                                         lWTP, lValueOfTime, lChangeFees,
00610                                                         lChangeFeeDisutility, lNonRefundable,
00611                                                         lNonRefundableDisutility);
00612
00613     stdair::BookingRequestPtr_T oBookingRequest_ptr =
00614         boost::make_shared<stdair::BookingRequestStruct> (lBookingRequestStruct);
00615
00616     // DEBUG
00617     // Be careful: this specific display is mandatory to retrieve the booking
00618     // requests when parsing the demand generation log with python scripts.
00619     STDAIR_LOG_NOTIFICATION ("\n[BKG] " << oBookingRequest_ptr->describe());
00620
00621     return oBookingRequest_ptr;
00622 }
00623
00624 // //////////////////////////////////////
00625 void DemandStream::reset (stdair::BaseGenerator_T& ioSharedGenerator) {
00626     _randomGenerationContext.reset();
00627     init (ioSharedGenerator);
00628 }
00629
00630 }

```

25.65 trademgen/bom/DemandStream.hpp File Reference

```
#include <stdair/bom/BomAbstract.hpp>
#include <stdair/bom/BookingRequestTypes.hpp>
#include <stdair/basic/RandomGeneration.hpp>
#include <stdair/basic/DemandGenerationMethod.hpp>
#include <trademgen/basic/DemandCharacteristics.hpp>
#include <trademgen/basic/DemandDistribution.hpp>
#include <trademgen/basic/RandomGenerationContext.hpp>
#include <trademgen/bom/DemandStreamKey.hpp>
#include <trademgen/bom/DemandStreamTypes.hpp>
```

Classes

- class [TRADEMGEN::DemandStream](#)
Class modeling a demand stream.

Namespaces

- namespace [stdair](#)
Forward declarations.
- namespace [TRADEMGEN](#)

25.66 DemandStream.hpp

```

00001 #ifndef __TRADEMGEN_BOM_DEMANDSTREAM_HPP
00002 #define __TRADEMGEN_BOM_DEMANDSTREAM_HPP
00003
00004 // //////////////////////////////////////
00005 // Import section
00006 // //////////////////////////////////////
00007 // StdAir
00008 #include <stdair/bom/BomAbstract.hpp>
00009 #include <stdair/bom/BookingRequestTypes.hpp>
00010 #include <stdair/basic/RandomGeneration.hpp>
00011 #include <stdair/basic/DemandGenerationMethod.hpp>
00012 // TraDemGen
00013 #include <trademgen/basic/DemandCharacteristics.hpp>
00014 #include <trademgen/basic/DemandDistribution.hpp>
00015 #include <trademgen/basic/RandomGenerationContext.hpp>
00016 #include <trademgen/bom/DemandStreamKey.hpp>
00017 #include <trademgen/bom/DemandStreamTypes.hpp>
00018
00020 namespace stdair {
00021     class FacBomManager;
00022     template <typename BOM> class FacBom;
00023 }
00024
00025 namespace TRADEMGEN {
00026
00030     class DemandStream : public stdair::BomAbstract {
00031     public:
00032         template <typename BOM> friend class stdair::FacBom;
00033         friend class stdair::FacBomManager;
00034
00035         // ////////////////////////////////// Type definitions //////////////////////////////////
00039         typedef DemandStreamKey Key_T;
00040
00042     public:
00043         // ////////////////////////////////// Getters //////////////////////////////////
00045         const Key_T& getKey() const {
00046             return _key;
00047         }
00048
00050         BomAbstract* const getParent() const {
00051             return _parent;
00052         }
00053
00055         const stdair::AirportCode_T& getOrigin() const {
00056             return _key.getOrigin();
00057         }
00058
00060         const stdair::AirportCode_T& getDestination() const {
00061             return _key.getDestination();
00062         }
00063
00065         const stdair::Date_T& getPreferredDepartureDate() const {
00066             return _key.getPreferredDepartureDate();
00067         }
00068
00070         const stdair::CabinCode_T& getPreferredCabin() const {
00071             return _key.getPreferredCabin();
00072         }
00073
00075         const stdair::HolderMap_T& getHolderMap() const {
00076             return _holderMap;
00077         }
00078
00080         const DemandCharacteristics& getDemandCharacteristics() const {

```

```

00081         return _demandCharacteristics;
00082     }
00083
00085     const DemandDistribution& getDemandDistribution() const {
00086         return _demandDistribution;
00087     }
00088
00090     const stdair::NbOfRequests_T& getTotalNumberOfRequestsToBeGenerated() const {
00091         return _totalNumberOfRequestsToBeGenerated;
00092     }
00093
00095     const stdair::NbOfRequests_T& getMeanNumberOfRequests() const {
00096         return _demandDistribution._meanNumberOfRequests;
00097     }
00098
00100     const stdair::StdDevValue_T& getStdDevNumberOfRequests() const {
00101         return _demandDistribution._stdDevNumberOfRequests;
00102     }
00103
00105     const stdair::Count_T& getNumberOfRequestsGeneratedSoFar() const {
00106         return _randomGenerationContext.getNumberOfRequestsGeneratedSoFar();
00107     }
00108
00110     const stdair::Disutility_T& getChangeFeeDisutility() const {
00111         return _demandCharacteristics._changeFeeDisutility;
00112     }
00113
00115     const stdair::Disutility_T& getNonRefundableDisutility() const {
00116         return _demandCharacteristics._nonRefundableDisutility;
00117     }
00118
00123     const POSProbabilityMass_T& getPOSProbabilityMass() const {
00124         return _posProMass;
00125     }
00126
00127
00128 public:
00129     // //////////// Setters ////////////
00131     void setNumberOfRequestsGeneratedSoFar (const stdair:: Count_T& iCount) {
00132         _randomGenerationContext.setNumberOfRequestsGeneratedSoFar (iCount);
00133     }
00134
00136     void setDemandDistribution (const DemandDistribution& iDemandDistribution) {
00137         _demandDistribution = iDemandDistribution;
00138     }
00139
00141     void
00142     setDemandCharacteristics (const ArrivalPatternCumulativeDistribution_T& iArri
valPattern,
00143                             const POSProbabilityMassFunction_T& iPOSProbMass,
00144                             const ChannelProbabilityMassFunction_T& iChannelPro
bMass,
00145                             const TripTypeProbabilityMassFunction_T& iTripTypeP
robMass,
00146                             const StayDurationProbabilityMassFunction_T& iStayD
urationProbMass,
00147                             const FrequentFlyerProbabilityMassFunction_T& iFreq
uentFlyerProbMass,
00148                             const stdair::ChangeFeesRatio_T& iChangeFeeProb,
00149                             const stdair::Disutility_T& iChangeFeeDisutility,
00150                             const stdair::NonRefundableRatio_T& iNonRefundableP
rob,
00151                             const stdair::Disutility_T& iNonRefundableDisutilit
y,
00152                             const
PreferredDepartureTimeContinuousDistribution_T& iPreferredDepartureTimeContinuous
Distribution,

```

```

00153             const stdair::WTP_T& iMinWTP,
00154             const ValueOfTimeContinuousDistribution_T& iValueOf
TimeContinuousDistribution) {
00155     _demandCharacteristics =
00156         DemandCharacteristics (iArrivalPattern, iPOSProbMass,
00157             iChannelProbMass, iTripTypeProbMass,
00158             iStayDurationProbMass, iFrequentFlyerProbMass,
00159             iChangeFeeProb, iChangeFeeDisutility,
00160             iNonRefundableProb, iNonRefundableDisutility,
00161             iPreferredDepartureTimeContinuousDistribution,
00162             iMinWTP, iValueOfTimeContinuousDistribution);
00163     }
00164
00166     void setTotalNumberOfRequestsToBeGenerated (const stdair::NbOfRequests_T& iNb
OfRequests) {
00167         _totalNumberOfRequestsToBeGenerated = iNbOfRequests;
00168     }
00169
00171     void setRequestDateTimeRandomGeneratorSeed (const stdair::RandomSeed_T& iSeed
) {
00172         _requestDateTimeRandomGenerator.init (iSeed);
00173     }
00174
00176     void setDemandCharacteristicsRandomGeneratorSeed (const stdair::RandomSeed_T&
iSeed) {
00177         _demandCharacteristicsRandomGenerator.init (iSeed);
00178     }
00179
00184     void setPOSProbabilityMass (const POSProbabilityMass_T& iProbMass) {
00185         _posProMass = iProbMass;
00186     }
00187
00191     void setAll (const ArrivalPatternCumulativeDistribution_T&,
00192         const POSProbabilityMassFunction_T&,
00193         const ChannelProbabilityMassFunction_T&,
00194         const TripTypeProbabilityMassFunction_T&,
00195         const StayDurationProbabilityMassFunction_T&,
00196         const FrequentFlyerProbabilityMassFunction_T&,
00197         const stdair::ChangeFeesRatio_T&,
00198         const stdair::Disutility_T&,
00199         const stdair::NonRefundableRatio_T&,
00200         const stdair::Disutility_T&,
00201         const PreferredDepartureTimeContinuousDistribution_T&,
00202         const stdair::WTP_T&,
00203         const ValueOfTimeContinuousDistribution_T&,
00204         const DemandDistribution&,
00205         stdair::BaseGenerator_T& ioSharedGenerator,
00206         const stdair::RandomSeed_T& iRequestDateTimeSeed,
00207         const stdair::RandomSeed_T& iDemandCharacteristicsSeed,
00208         const POSProbabilityMass_T&);
00209
00214     void setBoolFirstDateTimeRequest (const bool& iFirstDateTimeRequest) {
00215         _firstDateTimeRequest = iFirstDateTimeRequest;
00216     }
00217
00218
00219 public:
00220     // ////////////////////////////////// Business Methods //////////////////////////////////
00222     void incrementGeneratedRequestsCounter () {
00223         _randomGenerationContext.incrementGeneratedRequestsCounter();
00224     }
00225
00227     const bool stillHavingRequestsToBeGenerated (const stdair::DemandGenerationMe
thod& iDemandGenerationMethod) const;
00228
00230     const stdair::DateTime_T generateTimeOfRequestPoissonProcess();
00231

```



```

00233     const stdair::DateTime_T generateTimeOfRequestStatisticsOrder();
00234
00236     const stdair::AirportCode_T generatePOS();
00237
00239     const stdair::ChannelLabel_T generateChannel();
00240
00242     const stdair::TripType_T generateTripType();
00243
00245     const stdair::DayDuration_T generateStayDuration();
00246
00248     const stdair::FrequentFlyer_T generateFrequentFlyer();
00249
00251     const stdair::ChangeFees_T generateChangeFees();
00252
00254     const stdair::NonRefundable_T generateNonRefundable();
00255
00257     const stdair::Duration_T generatePreferredDepartureTime();
00258
00260     const stdair::WTP_T generateWTP (stdair::RandomGeneration&,
00261                                     const stdair::Date_T&,
00262                                     const stdair::DateTime_T&,
00263                                     const stdair::DayDuration_T&);
00264
00266     const stdair::PriceValue_T generateValueOfTime();
00267
00278     stdair::BookingRequestPtr_T
00279     generateNextRequest (stdair::RandomGeneration&,
00280                         const stdair::DemandGenerationMethod&);
00281
00283     void reset (stdair::BaseGenerator_T& ioSharedGenerator);
00284
00285
00286 public:
00287     // //////////// Display support methods ////////////
00292     void toStream (std::ostream& ioOut) const {
00293         ioOut << toString();
00294     }
00295
00300     void fromStream (std::istream& ioIn) {
00301     }
00302
00306     std::string toString() const;
00307
00311     const std::string describeKey() const {
00312         return _key.toString();
00313     }
00314
00318     std::string display() const;
00319     const stdair::Duration_T convertFloatIntoDuration (const stdair::FloatDuratio
n_T);
00320
00321 protected:
00322     // //////////// Constructors and destructors ////////////
00326     DemandStream (const Key_T&);
00330     virtual ~DemandStream();
00331
00332 private:
00334     DemandStream();
00336     DemandStream (const DemandStream&);
00338     void init (stdair::BaseGenerator_T& ioSharedGenerator);
00339
00340
00341 protected:
00342     // //////////// Attributes ////////////
00346     Key_T _key;
00347
00351     BomAbstract* _parent;

```

```
00352
00356     stdair::HolderMap_T _holderMap;
00357
00361     DemandCharacteristics _demandCharacteristics;
00362
00366     DemandDistribution _demandDistribution;
00367
00371     stdair::NbOfRequests_T _totalNumberOfRequestsToBeGenerated;
00372
00376     RandomGenerationContext _randomGenerationContext;
00377
00381     stdair::RandomGeneration _requestDateTimeRandomGenerator;
00382
00386     stdair::RandomGeneration _demandCharacteristicsRandomGenerator;
00387
00392     POSProbabilityMass_T _posProMass;
00393
00394     private:
00395         bool _stillHavingRequestsToBeGenerated;
00396         bool _firstDateTimeRequest;
00397         stdair::FloatDuration_T _dateTimeLastRequest;
00398     };
00399
00400 }
00401 #endif // __TRADEMGEN_BOM_DEMANDSTREAM_HPP
```

25.67 trademgen/bom/DemandStreamKey.cpp File Reference

```
#include <cassert>
#include <sstream>
#include <stdair/basic/BasConst_Inventory.hpp>
#include <trademgen/bom/DemandStreamKey.hpp>
```

Namespaces

- namespace [TRADEMGEN](#)

25.68 DemandStreamKey.cpp

```

00001 // //////////////////////////////////////
00002 // Import section
00003 // //////////////////////////////////////
00004 // STL
00005 #include <cassert>
00006 #include <sstream>
00007 // StdAir
00008 #include <stdair/basic/BasConst_Inventory.hpp>
00009 // TraDemGen
00010 #include <trademgen/bom/DemandStreamKey.hpp>
00011
00012 namespace TRADEMGEN {
00013
00014 // //////////////////////////////////////
00015 DemandStreamKey::DemandStreamKey()
00016 : _origin (stdair::DEFAULT_ORIGIN),
00017   _destination (stdair::DEFAULT_DESTINATION),
00018   _preferredDepartureDate (stdair::DEFAULT_DEPARTURE_DATE),
00019   _preferredCabin (stdair::DEFAULT_CABIN_CODE) {
00020     assert (false);
00021 }
00022
00023 // //////////////////////////////////////
00024 DemandStreamKey::
00025 DemandStreamKey (const stdair::AirportCode_T& iOrigin,
00026                 const stdair::AirportCode_T& iDestination,
00027                 const stdair::Date_T& iPreferredDepartureDate,
00028                 const stdair::CabinCode_T& iPreferredCabin)
00029 : _origin (iOrigin), _destination (iDestination),
00030   _preferredDepartureDate (iPreferredDepartureDate),
00031   _preferredCabin (iPreferredCabin) {
00032 }
00033
00034 // //////////////////////////////////////
00035 DemandStreamKey::DemandStreamKey (const DemandStreamKey& iKey)
00036 : _origin (iKey._origin), _destination (iKey._destination),
00037   _preferredDepartureDate (iKey._preferredDepartureDate),
00038   _preferredCabin (iKey._preferredCabin) {
00039 }
00040
00041 // //////////////////////////////////////
00042 DemandStreamKey::~DemandStreamKey () {
00043 }
00044
00045 // //////////////////////////////////////
00046 void DemandStreamKey::toStream (std::ostream& ioOut) const {
00047     ioOut << "DemandStreamKey: " << toString();
00048 }
00049
00050 // //////////////////////////////////////
00051 void DemandStreamKey::fromStream (std::istream& ioIn) {
00052 }
00053
00054 // //////////////////////////////////////
00055 const std::string DemandStreamKey::toString() const {
00056     std::ostringstream ostr;
00057     ostr << _origin << "-" << _destination << " " << _preferredDepartureDate
00058         << " " << _preferredCabin;
00059     return ostr.str();
00060 }
00061
00062 }

```

25.69 trademgen/bom/DemandStreamKey.hpp File Reference

```
#include <stdair/stdair_basic_types.hpp>
#include <stdair/stdair_date_time_types.hpp>
#include <stdair/bom/KeyAbstract.hpp>
```

Classes

- struct [TRADEMGEN::DemandStreamKey](#)

Namespaces

- namespace [TRADEMGEN](#)

25.70 DemandStreamKey.hpp

```

00001 #ifndef __TRADEMGEN_BOM_DEMANDSTREAMKEY_HPP
00002 #define __TRADEMGEN_BOM_DEMANDSTREAMKEY_HPP
00003
00004 // //////////////////////////////////////
00005 // Import section
00006 // //////////////////////////////////////
00007 // StdAir
00008 #include <stdair/stdair_basic_types.hpp>
00009 #include <stdair/stdair_date_time_types.hpp>
00010 #include <stdair/bom/KeyAbstract.hpp>
00011
00012 namespace TRADEMGEN {
00013
00020     struct DemandStreamKey : public stdair::KeyAbstract {
00021
00022         // ////////////////////////////////// Constructors and destructors //////////////////////////////////
00023     private:
00024         DemandStreamKey();
00025
00026     public:
00027         DemandStreamKey (const stdair::AirportCode_T& iOrigin,
00028                         const stdair::AirportCode_T& iDestination,
00029                         const stdair::Date_T& iPreferredDepartureDate,
00030                         const stdair::CabinCode_T& iPreferredCabin);
00031         DemandStreamKey (const DemandStreamKey&);
00032
00033         ~DemandStreamKey();
00034
00035     public:
00036         // ////////////////////////////////// Getters //////////////////////////////////
00037         const stdair::AirportCode_T& getOrigin() const {
00038             return _origin;
00039         }
00040
00041         const stdair::AirportCode_T& getDestination() const {
00042             return _destination;
00043         }
00044
00045         const stdair::Date_T& getPreferredDepartureDate () const {
00046             return _preferredDepartureDate;
00047         }
00048
00049         const stdair::CabinCode_T& getPreferredCabin() const {
00050             return _preferredCabin;
00051         }
00052
00053         // ////////////////////////////////// Display support methods //////////////////////////////////
00054         void toStream (std::ostream& ioOut) const;
00055
00056         void fromStream (std::istream& ioIn);
00057
00058         const std::string toString() const;
00059
00060     private:
00061         // ////////////////////////////////// Attributes //////////////////////////////////
00062         stdair::AirportCode_T _origin;
00063
00064         stdair::AirportCode_T _destination;
00065
00066         stdair::Date_T _preferredDepartureDate;
00067
00068         stdair::CabinCode_T _preferredCabin;

```

```
00093     };  
00094  
00095 }  
00096 #endif // __TRADEMGEN_BOM_DEMANDSTREAMKEY_HPP
```

25.71 trademgen/bom/DemandStreamTypes.hpp File Reference

```
#include <map>
#include <list>
#include <stdair/bom/key_types.hpp>
```

Namespaces

- namespace [TRADEMGEN](#)

Typedefs

- typedef std::list< DemandStream * > [TRADEMGEN::DemandStreamList_T](#)
- typedef std::map< const stdair::MapKey_T, DemandStream * >
[TRADEMGEN::DemandStreamMap_T](#)

25.72 DemandStreamTypes.hpp

```
00001 #ifndef __TRADEMGEN_BOM_DEMANDSTREAMTYPES_HPP
00002 #define __TRADEMGEN_BOM_DEMANDSTREAMTYPES_HPP
00003
00004 // //////////////////////////////////////
00005 // Import section
00006 // //////////////////////////////////////
00007 // STL
00008 #include <map>
00009 #include <list>
00010 // StdAir
00011 #include <stdair/bom/key_types.hpp>
00012
00013 namespace TRADEMGEN {
00014
00015     // Forward declarations.
00016     class DemandStream;
00017
00018     typedef std::list<DemandStream*> DemandStreamList_T;
00019
00020
00022     typedef std::map<const stdair::MapKey_T, DemandStream*> DemandStreamMap_T;
00023
00024 }
00025 #endif // __TRADEMGEN_BOM_DEMANDSTREAMTYPES_HPP
```

25.73 trademgen/bom/DemandStruct.cpp File Reference

```
#include <cassert>
#include <sstream>
#include <stdair/basic/BasConst_Inventory.hpp>
#include <stdair/basic/BasConst_Period_BOM.hpp>
#include <stdair/service/Logger.hpp>
#include <trademgen/TRADEMGEN_Types.hpp>
#include <trademgen/bom/DemandStruct.hpp>
```

Namespaces

- namespace [TRADEMGEN](#)

25.74 DemandStruct.cpp

```

00001 // //////////////////////////////////////
00002 // Import section
00003 // //////////////////////////////////////
00004 // STL
00005 #include <cassert>
00006 #include <sstream>
00007 // StdAir
00008 #include <stdair/basic/BasConst_Inventory.hpp>
00009 #include <stdair/basic/BasConst_Period_BOM.hpp>
00010 #include <stdair/service/Logger.hpp>
00011 // TRADEMGEN
00012 #include <trademgen/TRADEMGEN_Types.hpp>
00013 #include <trademgen/bom/DemandStruct.hpp>
00014
00015 namespace TRADEMGEN {
00016
00017 // //////////////////////////////////////
00018 DemandStruct::DemandStruct()
00019 : _dateRange (stdair::BOOST_DEFAULT_DATE_PERIOD),
00020   _dow (stdair::DEFAULT_DOW_STRING),
00021   _prefCabin (stdair::DEFAULT_CABIN_CODE),
00022   _itHours (0), _itMinutes (0), _itSeconds (0), _itFFCode ("") {
00023 }
00024
00025 // //////////////////////////////////////
00026 DemandStruct::~DemandStruct() {
00027 }
00028
00029 // //////////////////////////////////////
00030 stdair::Date_T DemandStruct::getDate() const {
00031     return stdair::Date_T (_itYear, _itMonth, _itDay);
00032 }
00033
00034 // //////////////////////////////////////
00035 stdair::Duration_T DemandStruct::getTime() const {
00036     return boost::posix_time::hours (_itHours)
00037        + boost::posix_time::minutes (_itMinutes)
00038        + boost::posix_time::seconds (_itSeconds);
00039 }
00040
00041 // //////////////////////////////////////
00042 const std::string DemandStruct::describe() const {
00043     std::ostringstream ostr;
00044     ostr << _dateRange << " - " << _dow
00045         << " " << _origin << "-" << _destination
00046         << " " << _prefCabin
00047         << ", N(" << _demandMean << ", " << _demandStdDev << ")";
00048
00049     unsigned short idx = 0;
00050     for (POSProbabilityMassFunction_T::const_iterator it = _posProbDist.begin();
00051          it != _posProbDist.end(); ++it, ++idx) {
00052         const stdair::AirportCode_T& lPosCode = it->first;
00053         const stdair::Probability_T& lPosProbMass = it->second;
00054         if (idx != 0) {
00055             ostr << ", ";
00056         }
00057         ostr << lPosCode << ":" << lPosProbMass;
00058     }
00059     ostr << "; ";
00060
00061     idx = 0;
00062     for (ChannelProbabilityMassFunction_T::const_iterator it =
00063          _channelProbDist.begin();
00064          it != _channelProbDist.end(); ++it, ++idx) {
00065         const stdair::ChannelLabel_T lChannelCode = it->first;

```

```

00066         const stdair::Probability_T& lChannelProbMass = it->second;
00067         if (idx != 0) {
00068             ostr << ", ";
00069         }
00070         ostr << lChannelCode << ":" << lChannelProbMass;
00071     }
00072     ostr << "; ";
00073
00074     idx = 0;
00075     for (TripTypeProbabilityMassFunction_T::const_iterator it =
00076         _tripProbDist.begin();
00077         it != _tripProbDist.end(); ++it, ++idx) {
00078         const stdair::TripType_T lTripCode = it->first;
00079         const stdair::Probability_T& lTripProbMass = it->second;
00080         if (idx != 0) {
00081             ostr << ", ";
00082         }
00083         ostr << lTripCode << ":" << lTripProbMass;
00084     }
00085     ostr << "; ";
00086
00087     idx = 0;
00088     for (StayDurationProbabilityMassFunction_T::const_iterator it =
00089         _stayProbDist.begin();
00090         it != _stayProbDist.end(); ++it, ++idx) {
00091         const stdair::DayDuration_T& lStayDuration = it->first;
00092         const stdair::Probability_T& lStayProbMass = it->second;
00093         if (idx != 0) {
00094             ostr << ", ";
00095         }
00096         ostr << lStayDuration << ":" << lStayProbMass;
00097     }
00098     ostr << "; ";
00099
00100     idx = 0;
00101     for (FrequentFlyerProbabilityMassFunction_T::const_iterator it =
00102         _ffProbDist.begin();
00103         it != _ffProbDist.end(); ++it, ++idx) {
00104         const stdair::FrequentFlyer_T lFFCode = it->first;
00105         const stdair::Probability_T& lFFProbMass = it->second;
00106         if (idx != 0) {
00107             ostr << ", ";
00108         }
00109         ostr << lFFCode << ":" << lFFProbMass;
00110     }
00111     ostr << "; ";
00112
00113     ostr << _changeFeeProb << "; ";
00114
00115     ostr << "; ";
00116
00117     ostr << _nonRefundableProb << "; ";
00118
00119     idx = 0;
00120     for (PreferredDepartureTimeContinuousDistribution_T::const_iterator it =
00121         _prefDepTimeProbDist.begin();
00122         it != _prefDepTimeProbDist.end(); ++it, ++idx) {
00123         const stdair::IntDuration_T& lPrefDepTime = it->first;
00124         const stdair::Probability_T& lPrefDepTimeProbMass = it->second;
00125         if (idx != 0) {
00126             ostr << ", ";
00127         }
00128         ostr << lPrefDepTime << ":" << lPrefDepTimeProbMass;
00129     }
00130     ostr << "; ";
00131
00132     ostr << _minWTP << "; ";

```

```
00133
00134     idx = 0;
00135     for (ValueOfTimeContinuousDistribution_T::const_iterator it =
00136         _timeValueProbDist.begin();
00137         it != _timeValueProbDist.end(); ++it, ++idx) {
00138         const stdair::PriceValue_T& lTimeValue = it->first;
00139         const stdair::Probability_T& lTimeValueProbMass = it->second;
00140         if (idx != 0) {
00141             ostr << ", ";
00142         }
00143         ostr << lTimeValue << ":" << lTimeValueProbMass;
00144     }
00145     ostr << "; ";
00146
00147     idx = 0;
00148     for (ArrivalPatternCumulativeDistribution_T::const_iterator it =
00149         _dtdProbDist.begin(); it != _dtdProbDist.end(); ++it, ++idx) {
00150         const stdair::FloatDuration_T& lDTD = it->first;
00151         const stdair::Probability_T& lDTDProbMass = it->second;
00152         if (idx != 0) {
00153             ostr << ", ";
00154         }
00155         ostr << lDTD << ":" << lDTDProbMass;
00156     }
00157     ostr << "; ";
00158
00159     return ostr.str();
00160 }
00161
00162 }
```

25.75 trademgen/bom/DemandStruct.hpp File Reference

```
#include <string>
#include <stdair/stdair_basic_types.hpp>
#include <stdair/stdair_maths_types.hpp>
#include <stdair/stdair_date_time_types.hpp>
#include <stdair/basic/StructAbstract.hpp>
#include <stdair/bom/DoWStruct.hpp>
#include <trademgen/basic/DemandCharacteristicsTypes.hpp>
```

Classes

- struct [TRADEMGEN::DemandStruct](#)

Namespaces

- namespace [TRADEMGEN](#)

25.76 DemandStruct.hpp

```

00001 #ifndef __TRADEMGEN_BOM_DEMANDSTRUCT_HPP
00002 #define __TRADEMGEN_BOM_DEMANDSTRUCT_HPP
00003
00004 // //////////////////////////////////////
00005 // Import section
00006 // //////////////////////////////////////
00007 // STL
00008 #include <string>
00009 // StdAir
00010 #include <stdair/stdair_basic_types.hpp>
00011 #include <stdair/stdair_maths_types.hpp>
00012 #include <stdair/stdair_date_time_types.hpp>
00013 #include <stdair/basic/StructAbstract.hpp>
00014 #include <stdair/bom/DoWStruct.hpp>
00015 // TraDemGen
00016 #include <trademgen/basic/DemandCharacteristicsTypes.hpp>
00017
00018 namespace TRADEMGEN {
00019
00021     struct DemandStruct : public stdair::StructAbstract {
00022
00023     public:
00024         // ////////////////////////////////////// Getters //////////////////////////////////////
00026         stdair::Date_T getDate() const;
00027
00029         stdair::Duration_T getTime() const;
00030
00031
00032     public:
00033         // ////////////////////////////////////// Display Support Methods //////////////////////////////////////
00035         const std::string describe() const;
00036
00037
00038     public:
00039         // ////////////////////////////////////// Constructors and destructors //////////////////////////////////////
00041         DemandStruct();
00043         ~DemandStruct();
00044     private:
00046         DemandStruct (const DemandStruct&);
00047
00048
00049     public:
00050         // ////////////////////////////////////// Attributes //////////////////////////////////////
00051         stdair::DatePeriod_T _dateRange;
00052         stdair::DoWStruct _dow;
00053         stdair::AirportCode_T _origin;
00054         stdair::AirportCode_T _destination;
00055         stdair::CabinCode_T _prefCabin;
00056         stdair::MeanValue_T _demandMean;
00057         stdair::StdDevValue_T _demandStdDev;
00058         stdair::ChangeFeesRatio_T _changeFeeProb;
00059         stdair::Disutility_T _changeFeeDisutility;
00060         stdair::NonRefundableRatio_T _nonRefundableProb;
00061         stdair::Disutility_T _nonRefundableDisutility;
00062         POSProbabilityMassFunction_T _posProbDist;
00063         ChannelProbabilityMassFunction_T _channelProbDist;
00064         TripTypeProbabilityMassFunction_T _tripProbDist;
00065         StayDurationProbabilityMassFunction_T _stayProbDist;
00066         FrequentFlyerProbabilityMassFunction_T _ffProbDist;
00067         PreferredDepartureTimeContinuousDistribution_T _prefDepTimeProbDist;
00068         stdair::WTP_T _minWTP;
00069         ValueOfTimeContinuousDistribution_T _timeValueProbDist;
00070         ArrivalPatternCumulativeDistribution_T _dtdProbDist;
00071
00072     public:

```

```
00073      // //////////// Staging ////////////
00075      stdair::Date_T _prefDepDateStart;
00076      stdair::Date_T _prefDepDateEnd;
00077      unsigned int _itYear;
00078      unsigned int _itMonth;
00079      unsigned int _itDay;
00080
00082      long _itHours;
00083      long _itMinutes;
00084      long _itSeconds;
00085
00087      stdair::AirportCode_T _itPosCode;
00088
00090      stdair::ChannelLabel_T _itChannelCode;
00091
00093      stdair::TripType_T _itTripCode;
00094
00096      stdair::DayDuration_T _itStayDuration;
00097
00099      stdair::FrequentFlyer_T _itFFCode;
00100
00102      stdair::Duration_T _itPrefDepTime;
00103
00105      stdair::PriceValue_T _itTimeValue;
00106
00108      stdair::DayDuration_T _itDTD;
00109  };
00110
00111 }
00112 #endif // __TRADEMGEN_BOM_DEMANDSTRUCT_HPP
```


25.77 trademgen/command/DBManager.cpp File Reference

```
#include <cassert>
#include <soci/soci.h>
#include <soci/mysql/soci-mysql.h>
#include <stdair/bom/AirlineStruct.hpp>
#include <stdair/service/Logger.hpp>
#include <trademgen/command/DBManager.hpp>
```

Namespaces

- namespace [TRADEMGEN](#)

25.78 DBManager.cpp

```

00001 // //////////////////////////////////////
00002 // Import section
00003 // //////////////////////////////////////
00004 // STL
00005 #include <cassert>
00006 // SOCI
00007 #if defined(SOCI_HEADERS_BURIED)
00008 #include <soci/core/soci.h>
00009 #include <soci/backends/mysql/soci-mysql.h>
00010 #else // SOCI_HEADERS_BURIED
00011 #include <soci/soci.h>
00012 #include <soci/mysql/soci-mysql.h>
00013 #endif // SOCI_HEADERS_BURIED
00014 // StdAir
00015 #include <stdair/bom/AirlineStruct.hpp>
00016 #include <stdair/service/Logger.hpp>
00017 // TraDemGen
00018 #include <trademgen/command/DBManager.hpp>
00019
00020 namespace TRADEMGEN {
00021
00022 // //////////////////////////////////////
00023 void DBManager::
00024 prepareSelectStatement (stdair::DBSession_T& ioSociSession,
00025                        stdair::DBRequestStatement_T& ioSelectStatement,
00026                        stdair::AirlineStruct& ioAirline) {
00027
00028     try {
00029
00030         // Instanciate a SQL statement (no request is performed at that stage)
00044     } catch (std::exception const& lException) {
00045         STDAIR_LOG_ERROR ("Error: " << lException.what());
00046         throw stdair::SQLDatabaseException (lException.what());
00047     }
00048 }
00049
00050 // //////////////////////////////////////
00051 void DBManager::
00052 prepareSelectOnAirlineCodeStatement (stdair::DBSession_T& ioSociSession,
00053                                     stdair::DBRequestStatement_T& ioSelectStat
00054                                     ement,
00055                                     const stdair::AirlineCode_T& iAirlineCode,
00056                                     stdair::AirlineStruct& ioAirline) {
00057
00058     try {
00059
00060         // Instanciate a SQL statement (no request is performed at that stage)
00090     } catch (std::exception const& lException) {
00091         STDAIR_LOG_ERROR ("Error: " << lException.what());
00092         throw stdair::SQLDatabaseException (lException.what());
00093     }
00094 }
00095
00096 // //////////////////////////////////////
00097 bool DBManager::iterateOnStatement (stdair::DBRequestStatement_T& ioStatement,
00098                                    stdair::AirlineStruct& ioAirline,
00099                                    const bool iShouldDoReset) {
00100     bool hasStillData = false;
00101
00102     try {
00103
00104         // Reset the list of names of the given Airline object
00105         if (iShouldDoReset == true) {
00106             // ioAirline.resetMatrix();

```

```

00107     }
00108
00109     // Retrieve the next row of Airline object
00110     hasStillData = ioStatement.fetch();
00111
00112     } catch (std::exception const& lException) {
00113         STDAIR_LOG_ERROR ("Error: " << lException.what());
00114         throw stdair::SQLDatabaseException (lException.what());
00115     }
00116
00117     return hasStillData;
00118 }
00119
00120 // //////////////////////////////////////
00121 void DBManager::updateAirlineInDB (stdair::DBSession_T& ioSociSession,
00122                                     const stdair::AirlineStruct& iAirline) {
00123
00124     try {
00125
00126         // Begin a transaction on the database
00127         ioSociSession.begin();
00128
00129         // Instanciate a SQL statement (no request is performed at that stage)
00130         std::string lAirlineCode;
00131         /*
00132         stdair::DBRequestStatement_T lUpdateStatement =
00133             (ioSociSession.prepare
00134              << "update ref_airline_details "
00135              << "set xapian_docid = :xapian_docid "
00136              << "where code = :code", soci::use (lDocID), soci::use (lAirlineCode));
00137
00138         // Execute the SQL query
00139         lDocID = iAirline.getDocID();
00140         lAirlineCode = iAirline.getAirlineCode();
00141         lUpdateStatement.execute (true);
00142         */
00143
00144         // Commit the transaction on the database
00145         ioSociSession.commit();
00146
00147         // Debug
00148         // TRADEMGEN_LOG_DEBUG ("[" << lDocID << "]" << iAirline);
00149
00150     } catch (std::exception const& lException) {
00151         STDAIR_LOG_ERROR ("Error: " << lException.what());
00152         throw stdair::SQLDatabaseException (lException.what());
00153     }
00154 }
00155
00156 // //////////////////////////////////////
00157 bool DBManager::retrieveAirline (stdair::DBSession_T& ioSociSession,
00158                                   const stdair::AirlineCode_T& iAirlineCode,
00159                                   stdair::AirlineStruct& ioAirline) {
00160     bool oHasRetrievedAirline = false;
00161
00162     try {
00163
00164         // Prepare the SQL request corresponding to the select statement
00165         stdair::DBRequestStatement_T lSelectStatement (ioSociSession);
00166         DBManager::prepareSelectOnAirlineCodeStatement (ioSociSession,
00167                                                         lSelectStatement,
00168                                                         iAirlineCode, ioAirline);
00169         const bool shouldDoReset = true;
00170         bool hasStillData = iterateOnStatement (lSelectStatement, ioAirline,
00171                                                 shouldDoReset);
00172         if (hasStillData == true) {
00173             oHasRetrievedAirline = true;

```

```
00174     }
00175
00176     // Sanity check
00177     const bool shouldNotDoReset = false;
00178     hasStillData = iterateOnStatement (lSelectStatement, ioAirline,
00179                                     shouldNotDoReset);
00180     // Debug
00181     // STDAIR_LOG_DEBUG ("[" << iDocID << "]" " << ioAirline);
00182
00183 } catch (std::exception const& lException) {
00184     STDAIR_LOG_ERROR ("Error: " << lException.what());
00185     throw stdair::SQLDatabaseException (lException.what());
00186 }
00187
00188 return oHasRetrievedAirline;
00189 }
00190
00191 }
```

25.79 trademgen/command/DBManager.hpp File Reference

```
#include <stdair/stdair_basic_types.hpp>
#include <stdair/stdair_db.hpp>
#include <trademgen/TRADEMGEN_Types.hpp>
```

Classes

- class [TRADEMGEN::DBManager](#)

Namespaces

- namespace [TRADEMGEN](#)

25.80 DBManager.hpp

```

00001 #ifndef __TRADEMGEN_CMD_DBMANAGER_HPP
00002 #define __TRADEMGEN_CMD_DBMANAGER_HPP
00003
00004 // //////////////////////////////////////
00005 // Import section
00006 // //////////////////////////////////////
00007 // StdAir
00008 #include <stdair/stdair_basic_types.hpp>
00009 #include <stdair/stdair_db.hpp>
00010 // Trademgen
00011 #include <trademgen/TRADEMGEN_Types.hpp>
00012
00013 namespace TRADEMGEN {
00014
00015     // Forward declarations
00016     struct AirlineStruct;
00017
00018     class DBManager {
00019     public:
00020         static void updateAirlineInDB (stdair::DBSession_T&,
00021                                     const stdair::AirlineStruct&);
00022
00023         static bool retrieveAirline (stdair::DBSession_T&,
00024                                    const stdair::AirlineCode_T&,
00025                                    stdair::AirlineStruct&);
00026
00027     public:
00028         static void prepareSelectStatement (stdair::DBSession_T&,
00029                                            stdair::DBRequestStatement_T&,
00030                                            stdair::AirlineStruct&);
00031
00032         static bool iterateOnStatement (stdair::DBRequestStatement_T&,
00033                                        stdair::AirlineStruct&,
00034                                        const bool iShouldDoReset);
00035
00036     private:
00037         static void prepareSelectOnAirlineCodeStatement (stdair::DBSession_T&,
00038                                                         stdair::DBRequestStatement_T
00039                                                         &,
00040                                                         const stdair::AirlineCode_T&
00041                                                         ,
00042                                                         stdair::AirlineStruct&);
00043
00044     private:
00045         DBManager() {}
00046         DBManager(const DBManager&) {}
00047         ~DBManager() {}
00048     };
00049
00050 }
00051 #endif // __TRADEMGEN_CMD_DBMANAGER_HPP

```

25.81 trademgen/command/DemandManager.cpp File Reference

```
#include <cassert>
#include <boost/make_shared.hpp>
#include <stdair/basic/ProgressStatusSet.hpp>
#include <stdair/basic/BasConst_Request.hpp>
#include <stdair/bom/BomManager.hpp>
#include <stdair/bom/EventStruct.hpp>
#include <stdair/bom/BookingRequestStruct.hpp>
#include <stdair/bom/TravelSolutionStruct.hpp>
#include <stdair/bom/CancellationStruct.hpp>
#include <stdair/factory/FacBom.hpp>
#include <stdair/factory/FacBomManager.hpp>
#include <stdair/service/Logger.hpp>
#include <sevmgr/SEVMGR_Service.hpp>
#include <trademgen/basic/DemandCharacteristics.hpp>
#include <trademgen/basic/DemandDistribution.hpp>
#include <trademgen/bom/DemandStruct.hpp>
#include <trademgen/bom/DemandStream.hpp>
#include <trademgen/command/DemandManager.hpp>
```

Namespaces

- namespace [TRADEMGEN](#)

25.82 DemandManager.cpp

```

00001 // //////////////////////////////////////
00002 // Import section
00003 // //////////////////////////////////////
00004 // STL
00005 #include <cassert>
00006 // Boost
00007 #include <boost/make_shared.hpp>
00008 // StdAir
00009 #include <stdair/basic/ProgressStatusSet.hpp>
00010 #include <stdair/basic/BasConst_Request.hpp>
00011 #include <stdair/bom/BomManager.hpp>
00012 #include <stdair/bom/EventStruct.hpp>
00013 #include <stdair/bom/BookingRequestStruct.hpp>
00014 #include <stdair/bom/TravelSolutionStruct.hpp>
00015 #include <stdair/bom/CancellationStruct.hpp>
00016 #include <stdair/factory/FacBom.hpp>
00017 #include <stdair/factory/FacBomManager.hpp>
00018 #include <stdair/service/Logger.hpp>
00019 // SEvMgr
00020 #include <sevmgr/SEVMGR_Service.hpp>
00021 // TraDemGen
00022 #include <trademgen/basic/DemandCharacteristics.hpp>
00023 #include <trademgen/basic/DemandDistribution.hpp>
00024 #include <trademgen/bom/DemandStruct.hpp>
00025 #include <trademgen/bom/DemandStream.hpp>
00026 #include <trademgen/command/DemandManager.hpp>
00027
00028 namespace TRADEMGEN {
00029
00030 // //////////////////////////////////////
00031 void DemandManager::
00032     buildSampleBomStd (SEVMGR::SEVMGR_ServicePtr_T ioSEVMGR_ServicePtr,
00033                       stdair::RandomGeneration& ioSharedGenerator,
00034                       const POSProbabilityMass_T& iPOSProbMass) {
00035     // Sanity check
00036     assert (ioSEVMGR_ServicePtr != NULL);
00037
00038     // Key of the demand stream
00039     const stdair::AirportCode_T lOrigin ("SIN");
00040     const stdair::AirportCode_T lDestination ("BKK");
00041     const stdair::Date_T lDepDate (2011, 2, 14);
00042     const stdair::CabinCode_T lCabin ("Y");
00043
00044     //
00045     const DemandStreamKey lDemandStreamKey (lOrigin, lDestination, lDepDate,
00046                                             lCabin);
00047
00048     // DEBUG
00049     // STDAIR_LOG_DEBUG ("Demand stream key: " << lDemandStreamKey.describe());
00050
00051     // Distribution for the number of requests
00052     const stdair::MeanValue_T lDemandMean (10.0);
00053     const stdair::StdDevValue_T lDemandStdDev (1.0);
00054     const DemandDistribution lDemandDistribution (lDemandMean, lDemandStdDev);
00055
00056     // Seed
00057     const stdair::RandomSeed_T& lRequestDateTimeSeed =
00058         generateSeed (ioSharedGenerator);
00059     const stdair::RandomSeed_T& lDemandCharacteristicsSeed =
00060         generateSeed (ioSharedGenerator);
00061
00062     //
00063     ArrivalPatternCumulativeDistribution_T lDTPProbDist;
00064     lDTPProbDist.insert (ArrivalPatternCumulativeDistribution_T::value_type(-330,
00065                                     0));

```



```

00066     lDTPProbDist.insert(ArrivalPatternCumulativeDistribution_T::value_type(-40,
00067                                                                    0.2));

00068     lDTPProbDist.insert(ArrivalPatternCumulativeDistribution_T::value_type(-20,
00069                                                                    0.6));

00070     lDTPProbDist.insert(ArrivalPatternCumulativeDistribution_T::value_type(-1,
00071                                                                    1.0));

00072     //
00073     POSProbabilityMassFunction_T lPOSProbDist;
00074     lPOSProbDist.insert(POSProbabilityMassFunction_T::value_type("BKK", 0.3));
00075     lPOSProbDist.insert(POSProbabilityMassFunction_T::value_type("SIN", 0.7));
00076     //
00077     ChannelProbabilityMassFunction_T lChannelProbDist;
00078     lChannelProbDist.insert(ChannelProbabilityMassFunction_T::value_type("DF",
00079                                                                    0.1));

00080     lChannelProbDist.insert(ChannelProbabilityMassFunction_T::value_type("DN",
00081                                                                    0.3));

00082     lChannelProbDist.insert(ChannelProbabilityMassFunction_T::value_type("IF",
00083                                                                    0.4));

00084     lChannelProbDist.insert(ChannelProbabilityMassFunction_T::value_type("IN",
00085                                                                    0.2));

00086     //
00087     TripTypeProbabilityMassFunction_T lTripProbDist;
00088     lTripProbDist.insert(TripTypeProbabilityMassFunction_T::value_type("RO",
00089                                                                    0.6));
00090     lTripProbDist.insert(TripTypeProbabilityMassFunction_T::value_type("RI",
00091                                                                    0.2));
00092     lTripProbDist.insert(TripTypeProbabilityMassFunction_T::value_type("OW",
00093                                                                    0.2));
00094     //
00095     StayDurationProbabilityMassFunction_T lStayProbDist;
00096     lStayProbDist.insert(StayDurationProbabilityMassFunction_T::value_type(0,
00097                                                                    0.1));

00098     lStayProbDist.insert(StayDurationProbabilityMassFunction_T::value_type(1,
00099                                                                    0.1));

00100     lStayProbDist.insert(StayDurationProbabilityMassFunction_T::value_type(2,
00101                                                                    .15));

00102     lStayProbDist.insert(StayDurationProbabilityMassFunction_T::value_type(3,
00103                                                                    .15));

00104     lStayProbDist.insert(StayDurationProbabilityMassFunction_T::value_type(4,
00105                                                                    .15));

00106     lStayProbDist.insert(StayDurationProbabilityMassFunction_T::value_type(5,
00107                                                                    .35));

00108     //
00109     FrequentFlyerProbabilityMassFunction_T lFFProbDist;
00110     lFFProbDist.insert(FrequentFlyerProbabilityMassFunction_T::value_type("P",
00111                                                                    0.01));

00112     lFFProbDist.insert(FrequentFlyerProbabilityMassFunction_T::value_type("G",
00113                                                                    0.05));

00114     lFFProbDist.insert(FrequentFlyerProbabilityMassFunction_T::value_type("S",
00115                                                                    0.15));

00116     lFFProbDist.insert(FrequentFlyerProbabilityMassFunction_T::value_type("M",

```

```

00117                                                                 0.3));
00118     lFFProbDist.insert(FrequentFlyerProbabilityMassFunction_T::value_type("N",
00119                                                                 0.49));

00120     //
00121     PreferredDepartureTimeContinuousDistribution_T lPrefDepTimeProbDist;
00122     lPrefDepTimeProbDist.
00123         insert (PreferredDepartureTimeContinuousDistribution_T::value_type (6 * std
air::HOURL_CONVERTED_IN_SECONDS, 0));
00124     lPrefDepTimeProbDist.
00125         insert (PreferredDepartureTimeContinuousDistribution_T::value_type (7 * std
air::HOURL_CONVERTED_IN_SECONDS,
00126                                                                 0.1));
00127     lPrefDepTimeProbDist.
00128         insert (PreferredDepartureTimeContinuousDistribution_T::value_type (9 * std
air::HOURL_CONVERTED_IN_SECONDS,
00129                                                                 0.3));
00130     lPrefDepTimeProbDist.
00131         insert (PreferredDepartureTimeContinuousDistribution_T::value_type (17 * st
dair::HOURL_CONVERTED_IN_SECONDS,
00132                                                                 0.4));
00133     lPrefDepTimeProbDist.
00134         insert (PreferredDepartureTimeContinuousDistribution_T::value_type (19 * st
dair::HOURL_CONVERTED_IN_SECONDS,
00135                                                                 0.80));

00136     lPrefDepTimeProbDist.
00137         insert (PreferredDepartureTimeContinuousDistribution_T::value_type (20 * st
dair::HOURL_CONVERTED_IN_SECONDS,
00138                                                                 0.95));

00139     lPrefDepTimeProbDist.
00140         insert (PreferredDepartureTimeContinuousDistribution_T::value_type (22 * st
dair::HOURL_CONVERTED_IN_SECONDS,
00141                                                                 1));
00142     //
00143     ValueOfTimeContinuousDistribution_T lTimeValueProbDist;
00144     lTimeValueProbDist.insert (ValueOfTimeContinuousDistribution_T::value_type(15,

00145                                                                 0))
00146     ;
00147     lTimeValueProbDist.insert (ValueOfTimeContinuousDistribution_T::value_type(60,

00148                                                                 1))
00149     ;

00150     //
00151     const stdair::WTP_T lWTP (1000.0);
00152     const stdair::ChangeFeesRatio_T lChangeFees (0.5);
00153     const stdair::Disutility_T lChangeFeeDisutility (50);
00154     const stdair::NonRefundableRatio_T lNonRefundable (0.5);
00155     const stdair::Disutility_T lNonRefundableDisutility (50);
00156
00157     // Delegate the call to the dedicated command
00158     DemandStream& lDemandStream =
00159         createDemandStream (ioSEVMGR_ServicePtr, lDemandStreamKey, lDTPProbDist,
00160                             lPOSProbDist, lChannelProbDist, lTripProbDist,
00161                             lStayProbDist, lFFProbDist,
00162                             lChangeFees, lChangeFeeDisutility,
00163                             lNonRefundable, lNonRefundableDisutility,
00164                             lPrefDepTimeProbDist,
00165                             lWTP, lTimeValueProbDist, lDemandDistribution,
00166                             ioSharedGenerator.getBaseGenerator(),
00167                             lRequestDateTimeSeed,
00168                             lDemandCharacteristicsSeed, iPOSProbMass);
00169

```

```

00170     // Calculate the expected total number of events for the current
00171     // demand stream
00172     const stdair::NbOfRequests_T& lExpectedTotalNbOfEvents =
00173         lDemandStream.getMeanNumberOfRequests();
00174
00175     ioSEVMGR_ServicePtr->addStatus (stdair::EventType::BKG_REQ,
00176                                     lExpectedTotalNbOfEvents);
00177 }
00178
00179 // //////////////////////////////////////
00180 DemandStream& DemandManager::createDemandStream
00181 (SEVMGR::SEVMGR_ServicePtr_T ioSEVMGR_ServicePtr,
00182  const DemandStreamKey& iKey,
00183  const ArrivalPatternCumulativeDistribution_T& iArrivalPattern,
00184  const POSProbabilityMassFunction_T& iPOSProbMass,
00185  const ChannelProbabilityMassFunction_T& iChannelProbMass,
00186  const TripTypeProbabilityMassFunction_T& iTripTypeProbMass,
00187  const StayDurationProbabilityMassFunction_T& iStayDurationProbMass,
00188  const FrequentFlyerProbabilityMassFunction_T& iFrequentFlyerProbMass,
00189  const stdair::ChangeFeesRatio_T& iChangeFeeProb,
00190  const stdair::Disutility_T& iChangeFeeDisutility,
00191  const stdair::NonRefundableRatio_T& iNonRefundableProb,
00192  const stdair::Disutility_T& iNonRefundableDisutility,
00193  const PreferredDepartureTimeContinuousDistribution_T& iPreferredDepartureTimeC
00194  ontinuousDistribution,
00195  const stdair::WTP_T& iMinWTP,
00196  const ValueOfTimeContinuousDistribution_T& iValueOfTimeContinuousDistribution,
00197
00198  const DemandDistribution& iDemandDistribution,
00199  stdair::BaseGenerator_T& ioSharedGenerator,
00200  const stdair::RandomSeed_T& iRequestDateTimeSeed,
00201  const stdair::RandomSeed_T& iDemandCharacteristicsSeed,
00202  const POSProbabilityMass_T& iDefaultPOSProbabilityMass) {
00203
00204     // Sanity check
00205     assert (ioSEVMGR_ServicePtr != NULL);
00206
00207     //
00208     DemandStream& oDemandStream =
00209         stdair::FacBom<DemandStream>::instance().create (iKey);
00210
00211     oDemandStream.setAll (iArrivalPattern, iPOSProbMass,
00212                          iChannelProbMass, iTripTypeProbMass,
00213                          iStayDurationProbMass, iFrequentFlyerProbMass,
00214                          iChangeFeeProb, iChangeFeeDisutility,
00215                          iNonRefundableProb, iNonRefundableDisutility,
00216                          iPreferredDepartureTimeContinuousDistribution,
00217                          iMinWTP, iValueOfTimeContinuousDistribution,
00218                          iDemandDistribution, ioSharedGenerator,
00219                          iRequestDateTimeSeed, iDemandCharacteristicsSeed,
00220                          iDefaultPOSProbabilityMass);
00221
00222     ioSEVMGR_ServicePtr->addEventGenerator (oDemandStream);
00223
00224     return oDemandStream;
00225 }
00226
00227 // //////////////////////////////////////
00228 void DemandManager::
00229 createDemandCharacteristics (SEVMGR::SEVMGR_ServicePtr_T ioSEVMGR_ServicePtr,
00230                             stdair::RandomGeneration& ioSharedGenerator,
00231                             const POSProbabilityMass_T& iPOSProbMass,
00232                             const DemandStruct& iDemand) {
00233
00234     // Sanity check
00235     assert (ioSEVMGR_ServicePtr != NULL);
00236
00237     //

```

```

00238     stdair::BaseGenerator_T& lSharedGenerator =
00239         ioSharedGenerator.getBaseGenerator();
00240
00241     // Parse the date period and DoW and generate demand characteristics.
00242     const stdair::DatePeriod_T lDateRange = iDemand._dateRange;
00243     for (boost::gregorian::day_iterator itDate = lDateRange.begin();
00244         itDate != lDateRange.end(); ++itDate) {
00245         const stdair::Date_T& currentDate = *itDate;
00246
00247         // Retrieve, for the current day, the Day-Of-the-Week (thanks to Boost)
00248         const unsigned short currentDoW = currentDate.day_of_week().as_number();
00249
00250         // The demand structure stores which Days (-Of-the-Week) are
00251         // active within the week. For each day (Mon., Tue., etc.), a boolean
00252         // states whether the Flight is active for that day.
00253         const stdair::DoWStruct& lDoWList = iDemand._dow;
00254         const bool isDoWActive = lDoWList.getStandardDayOfWeek (currentDoW);
00255
00256         if (isDoWActive == true) {
00257             const DemandStreamKey lDemandStreamKey (iDemand._origin,
00258                                                     iDemand._destination,
00259                                                     currentDate,
00260                                                     iDemand._prefCabin);
00261             // DEBUG
00262             // STDAIR_LOG_DEBUG ("Demand stream key: " << lDemandStreamKey.describe()
00263         );
00264
00265             //
00266             const DemandDistribution lDemandDistribution (iDemand._demandMean,
00267                                                         iDemand._demandStdDev);
00268
00269             // Seed
00270             const stdair::RandomSeed_T& lRequestDateTimeSeed =
00271                 generateSeed (ioSharedGenerator);
00272             const stdair::RandomSeed_T& lDemandCharacteristicsSeed =
00273                 generateSeed (ioSharedGenerator);
00274
00275             // Delegate the call to the dedicated command
00276             DemandStream& lDemandStream =
00277                 createDemandStream (ioSEVMGR_ServicePtr, lDemandStreamKey,
00278                                     iDemand._dtdProbDist, iDemand._posProbDist,
00279                                     iDemand._channelProbDist,
00280                                     iDemand._tripProbDist,
00281                                     iDemand._stayProbDist, iDemand._ffProbDist,
00282                                     iDemand._changeFeeProb,
00283                                     iDemand._changeFeeDisutility,
00284                                     iDemand._nonRefundableProb,
00285                                     iDemand._nonRefundableDisutility,
00286                                     iDemand._prefDepTimeProbDist,
00287                                     iDemand._minWTP,
00288                                     iDemand._timeValueProbDist,
00289                                     lDemandDistribution, lSharedGenerator,
00290                                     lRequestDateTimeSeed,
00291                                     lDemandCharacteristicsSeed,
00292                                     iPOSProbMass);
00293
00294             // Calculate the expected total number of events for the current
00295             // demand stream
00296             const stdair::NbOfRequests_T& lExpectedTotalNbOfEvents =
00297                 lDemandStream.getMeanNumberOfRequests();
00298
00299             ioSEVMGR_ServicePtr->addStatus (stdair::EventType::BKG_REQ,
00300                                             lExpectedTotalNbOfEvents);
00301         }
00302     }
00303 }
00304 }
00305 }
00306 }
00307

```

```

00308 // //////////////////////////////////////
00309 stdair::RandomSeed_T DemandManager::
00310 generateSeed (stdair::RandomGeneration& ioSharedGenerator) {
00311     stdair::RealNumber_T lVariateUnif = ioSharedGenerator() * 1e9;
00312     stdair::RandomSeed_T oSeed = static_cast<stdair::RandomSeed_T>(lVariateUnif);

00313     return oSeed;
00314 }
00315
00316 // //////////////////////////////////////
00317 const bool DemandManager::
00318 stillHavingRequestsToBeGenerated (SEVMGR::SEVMGR_ServicePtr_T ioSEVMGR_ServiceP
tr,
00319                                     const stdair::DemandStreamKeyStr_T& iKey,
00320                                     stdair::ProgressStatusSet& ioPSS,
00321                                     const stdair::DemandGenerationMethod& iDemand
GenerationMethod) {
00322     // Sanity check
00323     assert (ioSEVMGR_ServicePtr != NULL);
00324
00325     // Retrieve the DemandStream which corresponds to the given key.
00326     const DemandStream& lDemandStream =
00327         ioSEVMGR_ServicePtr->getEventGenerator<DemandStream, stdair::DemandStreamKey
Str_T>(iKey);
00328
00329     // Retrieve the progress status of the demand stream.
00330     stdair::ProgressStatus
00331         lProgressStatus (lDemandStream.getNumberOfRequestsGeneratedSoFar(),
00332                         lDemandStream.getMeanNumberOfRequests(),
00333                         lDemandStream.getTotalNumberOfRequestsToBeGenerated());
00334     ioPSS.setSpecificGeneratorStatus (lProgressStatus, iKey);
00335
00336     return lDemandStream.stillHavingRequestsToBeGenerated (iDemandGenerationMetho
d);
00337 }
00338
00339 // //////////////////////////////////////
00340 stdair::BookingRequestPtr_T DemandManager::
00341 generateNextRequest (SEVMGR::SEVMGR_ServicePtr_T ioSEVMGR_ServicePtr,
00342                     stdair::RandomGeneration& ioGenerator,
00343                     const stdair::DemandStreamKeyStr_T& iKey,
00344                     const stdair::DemandGenerationMethod& iDemandGenerationMet
hod) {
00345     // Sanity check
00346     assert (ioSEVMGR_ServicePtr != NULL);
00347
00348     // Retrieve the DemandStream which corresponds to the given key.
00349     DemandStream& lDemandStream =
00350         ioSEVMGR_ServicePtr->getEventGenerator<DemandStream, stdair::DemandStreamKey
Str_T>(iKey);
00351
00352     // Generate the next booking request
00353     stdair::BookingRequestPtr_T lBookingRequest =
00354         lDemandStream.generateNextRequest (ioGenerator,
00355                                             iDemandGenerationMethod);
00356
00357     const stdair::DateTime_T& lBookingRequestDateTime =
00358         lBookingRequest->getRequestDateTime();
00359     const stdair::Date_T& lBookingRequestDate =
00360         lBookingRequestDateTime.date();
00361     const stdair::Duration_T& lBookingRequestTime =
00362         lBookingRequestDateTime.time_of_day();
00363     const stdair::Date_T& lPreferredDepartureDate =
00364         lBookingRequest->getPreferredDepartureDate();
00365     const stdair::Duration_T& lPreferredDepartureTime =
00366         lBookingRequest->getPreferredDepartureTime();
00367

```

```

00368     if ((lPreferredDepartureDate > lBookingRequestDate) ||
00369         (lPreferredDepartureDate == lBookingRequestDate &&
00370          lPreferredDepartureTime > lBookingRequestTime)) {
00371
00372         // Create an event structure
00373         stdair::EventStruct lEventStruct (stdair::EventType::BKG_REQ,
00374                                           lBookingRequest);
00375
00383         ioSEVMGR_ServicePtr->addEvent (lEventStruct);
00384
00385     } else {
00386
00387         // Update the expected number of eventss for the given event type (i.e.,
00388         // booking request)
00389         stdair::Count_T lCurrentBRNumber =
00390             ioSEVMGR_ServicePtr->getActualTotalNumberOfEventsToBeGenerated (stdair::E
ventType::BKG_REQ);
00391         lCurrentBRNumber--;
00392         ioSEVMGR_ServicePtr->updateStatus (stdair::EventType::BKG_REQ, lCurrentBRNu
mber);
00393
00394     }
00395
00396     return lBookingRequest;
00397 }
00398
00399 // //////////////////////////////////////
00400 stdair::Count_T DemandManager::
00401 generateFirstRequests (SEVMGR::SEVMGR_ServicePtr_T ioSEVMGR_ServicePtr,
00402                       stdair::RandomGeneration& ioGenerator,
00403                       const stdair::DemandGenerationMethod& iDemandGenerationM
ethod) {
00404     // Sanity check
00405     assert (ioSEVMGR_ServicePtr != NULL);
00406
00407     // Actual total number of events to be generated
00408     stdair::NbOfRequests_T lActualTotalNbOfEvents = 0.0;
00409
00410     // Retrieve the DemandStream list
00411     const DemandStreamList_T& lDemandStreamList =
00412         ioSEVMGR_ServicePtr->getEventGeneratorList<DemandStream>();
00413
00414     for (DemandStreamList_T::const_iterator itDemandStream =
00415          lDemandStreamList.begin();
00416          itDemandStream != lDemandStreamList.end(); ++itDemandStream) {
00417         DemandStream* lDemandStream_ptr = *itDemandStream;
00418         assert (lDemandStream_ptr != NULL);
00419
00420         lDemandStream_ptr->setBoolFirstDateTimeRequest (true);
00421
00422         // Calculate the expected total number of events for the current
00423         // demand stream
00424         const stdair::NbOfRequests_T& lActualNbOfEvents =
00425             lDemandStream_ptr->getTotalNumberOfRequestsToBeGenerated();
00426         lActualTotalNbOfEvents += lActualNbOfEvents;
00427
00428         // Retrieve the key of the demand stream
00429         const DemandStreamKey& lKey = lDemandStream_ptr->getKey();
00430
00431         // Check whether there are still booking requests to be generated
00432         const bool stillHavingRequestsToBeGenerated =
00433             lDemandStream_ptr->stillHavingRequestsToBeGenerated (iDemandGenerationMet
hod);
00434
00435         if (stillHavingRequestsToBeGenerated) {
00436             // Generate the next event (booking request), and insert it
00437             // into the event queue

```

```

00438         generateNextRequest (ioSEVMGR_ServicePtr, ioGenerator,
00439                               lKey.toString(),
00440                               iDemandGenerationMethod);
00441     }
00442 }
00443
00444 // Update the progress status for the given event type (i.e.,
00445 // booking request)
00446 ioSEVMGR_ServicePtr->updateStatus (stdair::EventType::BKG_REQ,
00447                                     lActualTotalNbOfEvents);
00448
00449 // Retrieve the actual total number of events to be generated
00450 const stdair::Count_T oTotalNbOfEvents = std::floor (lActualTotalNbOfEvents);
00451
00452 //
00453 return oTotalNbOfEvents;
00454 }
00455
00456 // //////////////////////////////////////
00457 void DemandManager::reset (SEVMGR::SEVMGR_ServicePtr_T ioSEVMGR_ServicePtr,
00458                             stdair::BaseGenerator_T& ioShareGenerator) {
00459     // Sanity check
00460     assert (ioSEVMGR_ServicePtr != NULL);
00461
00462     // TODO: check whether it is really necessary to destroy the
00463     // objects manually. Indeed, FacSupervisor::cleanAll() should
00464     // destroy any BOM object.
00465
00466     // Reset all the DemandStream objects
00467     const DemandStreamList_T& lDemandStreamList =
00468         ioSEVMGR_ServicePtr->getEventGeneratorList<DemandStream>();
00469     for (DemandStreamList_T::const_iterator itDS = lDemandStreamList.begin();
00470          itDS != lDemandStreamList.end(); ++itDS) {
00471         DemandStream* lCurrentDS_ptr = *itDS;
00472         assert (lCurrentDS_ptr != NULL);
00473
00474         lCurrentDS_ptr->reset (ioShareGenerator);
00475     }
00476
00477     ioSEVMGR_ServicePtr->reset();
00478 }
00479
00480 // //////////////////////////////////////
00481 bool DemandManager::
00482 generateCancellation (stdair::RandomGeneration& ioGenerator,
00483                       const stdair::TravelSolutionStruct& iTravelSolution,
00484                       const stdair::PartySize_T& iPartySize,
00485                       const stdair::DateTime_T& iRequestTime,
00486                       const stdair::Date_T& iDepartureDate,
00487                       stdair::EventStruct& ioEventStruct) {
00488
00489     // Draw a random number to decide if we generate a
00490     // cancellation. For instance, the probability will be hardcoded.
00491     // The cancellation time will be generated uniformly.
00492     double lRandomNumber = ioGenerator();
00493
00494     if (lRandomNumber >= 0.05) {
00495         return false;
00496     }
00497     lRandomNumber /= 0.05;
00498
00499     // Hardcode the latest cancellation time.
00500     const stdair::Time_T lMidNight =
00501         boost::posix_time::hours (0);
00502     const stdair::DateTime_T lDepartureDateTime =
00503         boost::posix_time::ptime (iDepartureDate, lMidNight);

```

```

00511
00512     // Time to departure.
00513     const stdair::Duration_T lTimeToDeparture = lDepartureDateTime-iRequestTime;
00514
00515     // Cancellation time to departure
00516     const long lTimeToDepartureInSeconds = lTimeToDeparture.total_seconds();
00517     const long lCancellationTimeToDepartureInSeconds =
00518         static_cast<long> (lTimeToDepartureInSeconds * lRandomNumber);
00519     const stdair::Duration_T lCancellationTimeToDeparture (0, 0, lCancellationTim
eToDepartureInSeconds);
00520
00521     // Cancellation time
00522     const stdair::DateTime_T lCancellationTime =
00523         lDepartureDateTime - lCancellationTimeToDeparture;
00524     const stdair::Duration_T lTimeBetweenCancellationAndTheRequest =
00525         lCancellationTime - iRequestTime;
00526
00527     if (lTimeBetweenCancellationAndTheRequest.is_negative() == true) {
00528         return false;
00529     }
00530
00531     // Build the list of Class ID's.
00532     stdair::BookingClassIDList_T lClassIDList;
00533
00534     const stdair::ClassObjectIDMapHolder_T& lClassObjectIDMapHolder =
00535         iTravelSolution.getClassObjectIDMapHolder();
00536     const stdair::FareOptionStruct& lChosenFareOption =
00537         iTravelSolution.getChosenFareOption ();
00538     const stdair::ClassList_StringList_T& lClassPath =
00539         lChosenFareOption.getClassPath();
00540     const stdair::SegmentPath_T& lSegmentPath =
00541         iTravelSolution.getSegmentPath();
00542     stdair::ClassList_StringList_T::const_iterator itClassKeyList =
00543         lClassPath.begin();
00544     for (stdair::ClassObjectIDMapHolder_T::const_iterator itClassObjectIDMap =
00545         lClassObjectIDMapHolder.begin();
00546         itClassObjectIDMap != lClassObjectIDMapHolder.end();
00547         ++itClassObjectIDMap, ++itClassKeyList) {
00548         const stdair::ClassObjectIDMap_T& lClassObjectIDMap = *itClassObjectIDMap;
00549
00550         // TODO: Remove this hard-coded part.
00551         std::ostringstream ostr;
00552         const stdair::ClassList_String_T& lClassList = *itClassKeyList;
00553         assert (lClassList.empty() == false);
00554
00555         ostr << lClassList.at(0);
00556         const stdair::ClassCode_T lClassCode (ostr.str());
00557
00558         stdair::ClassObjectIDMap_T::const_iterator itClassID =
00559             lClassObjectIDMap.find (lClassCode);
00560         assert (itClassID != lClassObjectIDMap.end());
00561         const stdair::BookingClassID_T& lClassID = itClassID->second;
00562
00563         lClassIDList.push_back (lClassID);
00564
00565     }
00566 }
00567
00568 // Create the cancellation.
00569 stdair::CancellationStruct lCancellationStruct (lSegmentPath,
00570         lClassIDList,
00571         iPartySize,
00572         lCancellationTime);
00573
00574 stdair::CancellationPtr_T lCancellation_ptr =
00575     boost::make_shared<stdair::CancellationStruct> (lCancellationStruct);
00576

```



```

00577     // Create an event structure
00578     stdair::EventStruct lEventStruct (stdair::EventType::CX, lCancellation_ptr);
00579     ioEventStruct = lEventStruct;
00580
00581     return true;
00582 }
00583
00584 // //////////////////////////////////////
00585 void DemandManager::
00586 buildSampleBom (SEVMGR::SEVMGR_ServicePtr_T ioSEVMGR_ServicePtr,
00587                stdair::RandomGeneration& ioSharedGenerator,
00588                const POSProbabilityMass_T& iPOSProbMass) {
00589     // Sanity check
00590     assert (ioSEVMGR_ServicePtr != NULL);
00591
00592     //
00593     ArrivalPatternCumulativeDistribution_T lDTPProbDist;
00594     lDTPProbDist.insert (ArrivalPatternCumulativeDistribution_T::value_type(-330,
00595                                     0));
00596     lDTPProbDist.insert (ArrivalPatternCumulativeDistribution_T::value_type(-150,
00597                                     0.1));
00598
00599     lDTPProbDist.insert (ArrivalPatternCumulativeDistribution_T::value_type(-92,
00600                                     0.2));
00601
00602     lDTPProbDist.insert (ArrivalPatternCumulativeDistribution_T::value_type(-55,
00603                                     0.3));
00604
00605     lDTPProbDist.insert (ArrivalPatternCumulativeDistribution_T::value_type(-34,
00606                                     0.4));
00607
00608     lDTPProbDist.insert (ArrivalPatternCumulativeDistribution_T::value_type(-21,
00609                                     0.5));
00610
00611     lDTPProbDist.insert (ArrivalPatternCumulativeDistribution_T::value_type(-12,
00612                                     0.6));
00613
00614     lDTPProbDist.insert (ArrivalPatternCumulativeDistribution_T::value_type(-6,
00615                                     0.7));
00616
00617     lDTPProbDist.insert (ArrivalPatternCumulativeDistribution_T::value_type(-3,
00618                                     0.8));
00619
00620     lDTPProbDist.insert (ArrivalPatternCumulativeDistribution_T::value_type(-1,
00621                                     0.9));
00622
00623     lDTPProbDist.insert (ArrivalPatternCumulativeDistribution_T::value_type(0,
00624                                     1.0));
00625
00626     //
00627     ChannelProbabilityMassFunction_T lChannelProbDist;
00628     lChannelProbDist.insert (ChannelProbabilityMassFunction_T::value_type ("DF",
00629                                     0.0));
00630
00631     lChannelProbDist.insert (ChannelProbabilityMassFunction_T::value_type ("DN",
00632                                     0.0));
00633
00634     lChannelProbDist.insert (ChannelProbabilityMassFunction_T::value_type ("IF",
00635                                     0.0));
00636
00637     lChannelProbDist.insert (ChannelProbabilityMassFunction_T::value_type ("IN",
00638                                     1.0));
00639
00640     //
00641     TripTypeProbabilityMassFunction_T lTripProbDist;

```

```

00630     lTripProbDist.insert (TripTypeProbabilityMassFunction_T::value_type ("RO",
00631                                                                    0.0));
00632     lTripProbDist.insert (TripTypeProbabilityMassFunction_T::value_type ("RI",
00633                                                                    0.0));
00634     lTripProbDist.insert (TripTypeProbabilityMassFunction_T::value_type ("OW",
00635                                                                    1.0));
00636
00637     //
00638     StayDurationProbabilityMassFunction_T lStayProbDist;
00639     lStayProbDist.insert (StayDurationProbabilityMassFunction_T::value_type(0,
00640                                                                    0.1));
00641
00642     lStayProbDist.insert (StayDurationProbabilityMassFunction_T::value_type(1,
00643                                                                    0.1));
00644
00645     lStayProbDist.insert (StayDurationProbabilityMassFunction_T::value_type(2,
00646                                                                    .15));
00647
00648     lStayProbDist.insert (StayDurationProbabilityMassFunction_T::value_type(3,
00649                                                                    .15));
00650
00651     lStayProbDist.insert (StayDurationProbabilityMassFunction_T::value_type(4,
00652                                                                    .15));
00653
00654     lStayProbDist.insert (StayDurationProbabilityMassFunction_T::value_type(5,
00655                                                                    .35));
00656
00657     //
00658     FrequentFlyerProbabilityMassFunction_T lFFProbDist;
00659     lFFProbDist.insert (FrequentFlyerProbabilityMassFunction_T::value_type("P",
00660                                                                    0.1));
00661     lFFProbDist.insert (FrequentFlyerProbabilityMassFunction_T::value_type("G",
00662                                                                    0.01));
00663
00664     lFFProbDist.insert (FrequentFlyerProbabilityMassFunction_T::value_type("S",
00665                                                                    0.09));
00666
00667     lFFProbDist.insert (FrequentFlyerProbabilityMassFunction_T::value_type("M",
00668                                                                    0.4));
00669     lFFProbDist.insert (FrequentFlyerProbabilityMassFunction_T::value_type("N",
00670                                                                    0.4));
00671
00672     //
00673     ValueOfTimeContinuousDistribution_T lTimeValueProbDist;
00674     lTimeValueProbDist.insert (ValueOfTimeContinuousDistribution_T::value_type(15,
00675                                                                    0))
00676     ;
00677     lTimeValueProbDist.insert (ValueOfTimeContinuousDistribution_T::value_type(60,
00678                                                                    1))
00679     ;
00680
00681     /*=====*/
00682     // Key of the demand stream
00683     const stdair::AirportCode_T lSINOrigin ("SIN");
00684     const stdair::AirportCode_T lBKKDestination ("BKK");
00685     const stdair::Date_T lDepDate (2010, 2, 8);
00686     const stdair::CabinCode_T lCabin ("Y");
00687
00688     //
00689     const DemandStreamKey lSINBKKDemandStreamKey (lSINOrigin, lBKKDestination, lDepDate,
00690                                                                    lCabin);

```

```

00683
00684     // DEBUG
00685     // STDAIR_LOG_DEBUG ("Demand stream key: " << lDemandStreamKey.describe());
00686
00687     // Distribution for the number of requests
00688     const stdair::MeanValue_T lSINBKKDemandMean (60.0);
00689     const stdair::StdDevValue_T lSINBKKDemandStdDev (4.0);
00690     const DemandDistribution lSINBKKDemandDistribution (lSINBKKDemandMean, lSINBK
KDemandStdDev);
00691
00692     // Seed
00693     const stdair::RandomSeed_T& lSINBKKRequestDateTimeSeed =
00694         generateSeed (ioSharedGenerator);
00695     const stdair::RandomSeed_T& lSINBKKDemandCharacteristicsSeed =
00696         generateSeed (ioSharedGenerator);
00697
00698
00699     //
00700     POSProbabilityMassFunction_T lSINBKKPOSProbDist;
00701     lSINBKKPOSProbDist.insert (POSProbabilityMassFunction_T::value_type ("SIN", 1
.0));
00702     lSINBKKPOSProbDist.insert (POSProbabilityMassFunction_T::value_type ("BKK", 0
.0));
00703
00704     //
00705     PreferredDepartureTimeContinuousDistribution_T lSINPrefDepTimeProbDist;
00706     lSINPrefDepTimeProbDist.
00707         insert (PreferredDepartureTimeContinuousDistribution_T::value_type (6 * std
air::HOURL_CONVERTED_IN_SECONDS, 0));
00708     lSINPrefDepTimeProbDist.
00709         insert (PreferredDepartureTimeContinuousDistribution_T::value_type (8 * std
air::HOURL_CONVERTED_IN_SECONDS,
0.7));
00710
00711     lSINPrefDepTimeProbDist.
00712         insert (PreferredDepartureTimeContinuousDistribution_T::value_type (10 * st
dair::HOURL_CONVERTED_IN_SECONDS,
0.8));
00713
00714     lSINPrefDepTimeProbDist.
00715         insert (PreferredDepartureTimeContinuousDistribution_T::value_type (12 * st
dair::HOURL_CONVERTED_IN_SECONDS,
0.9));
00716
00717     lSINPrefDepTimeProbDist.
00718         insert (PreferredDepartureTimeContinuousDistribution_T::value_type (14 * st
dair::HOURL_CONVERTED_IN_SECONDS,
1.0));
00719
00720
00721     //
00722     const stdair::WTP_T lSINBKKWTP (400.0);
00723     const stdair::ChangeFeesRatio_T lChangeFees (0.5);
00724     const stdair::Disutility_T lChangeFeeDisutility (50);
00725     const stdair::NonRefundableRatio_T lNonRefundable (0.5);
00726     const stdair::Disutility_T lNonRefundableDisutility (50);
00727
00728
00729     // Delegate the call to the dedicated command
00730     DemandStream& lSINBKKDemandStream =
00731         createDemandStream (ioSEVMGR_ServicePtr, lSINBKKDemandStreamKey, lDTPProbDi
st,
00732                             lSINBKKPOSProbDist, lChannelProbDist, lTripProbDist,
00733                             lStayProbDist, lFFProbDist,
00734                             lChangeFees, lChangeFeeDisutility,
00735                             lNonRefundable, lNonRefundableDisutility,
00736                             lSINPrefDepTimeProbDist,
00737                             lSINBKKWTP, lTimeValueProbDist,
00738                             lSINBKKDemandDistribution,
00739                             ioSharedGenerator.getBaseGenerator(),

```

```

00740             lSINBKKRequestDateTimeSeed,
00741             lSINBKKDemandCharacteristicsSeed, iPOSProbMass);
00742
00743         // Calculate the expected total number of events for the current
00744         // demand stream
00745         const stdair::NbOfRequests_T& lSINBKKExpectedNbOfEvents =
00746             lSINBKKDemandStream.getMeanNumberOfRequests();
00747
00748         /*=====
=====*/
00749
00750         // Key of the demand stream
00751         const stdair::AirportCode_T lBKKOrigin ("BKK");
00752         const stdair::AirportCode_T lHKGDestination ("HKG");
00753
00754         //
00755         const DemandStreamKey lBKKHKGDemandStreamKey (lBKKOrigin, lHKGDestination, lD
epDate,
00756                                                         lCabin);
00757
00758         // DEBUG
00759         // STDAIR_LOG_DEBUG ("Demand stream key: " << lDemandStreamKey.describe());
00760
00761         // Distribution for the number of requests
00762         const stdair::MeanValue_T lBKKHKGDemandMean (60.0);
00763         const stdair::StdDevValue_T lBKKHKGDemandStdDev (4.0);
00764         const DemandDistribution lBKKHKGDemandDistribution (lBKKHKGDemandMean, lBKKHK
GDemandStdDev);
00765
00766         // Seed
00767         const stdair::RandomSeed_T& lBKKHKGRequestDateTimeSeed =
00768             generateSeed (ioSharedGenerator);
00769         const stdair::RandomSeed_T& lBKKHKGDemandCharacteristicsSeed =
00770             generateSeed (ioSharedGenerator);
00771
00772
00773         //
00774         POSProbabilityMassFunction_T lBKKHKGPOSProbDist;
00775         lBKKHKGPOSProbDist.insert (POSProbabilityMassFunction_T::value_type ("BKK", 1
.0));
00776         lBKKHKGPOSProbDist.insert (POSProbabilityMassFunction_T::value_type ("HKG", 0
.0));
00777
00778         //
00779         PreferredDepartureTimeContinuousDistribution_T lBKKPrefDepTimeProbDist;
00780         lBKKPrefDepTimeProbDist.
00781             insert (PreferredDepartureTimeContinuousDistribution_T::value_type (8 * std
air::HOURL_CONVERTED_IN_SECONDS, 0));
00782         lBKKPrefDepTimeProbDist.
00783             insert (PreferredDepartureTimeContinuousDistribution_T::value_type (10 * st
dair::HOURL_CONVERTED_IN_SECONDS,
00784                                                         0.2));
00785         lBKKPrefDepTimeProbDist.
00786             insert (PreferredDepartureTimeContinuousDistribution_T::value_type (1 * std
air::HOURL_CONVERTED_IN_SECONDS,
00787                                                         0.6));
00788         lBKKPrefDepTimeProbDist.
00789             insert (PreferredDepartureTimeContinuousDistribution_T::value_type (14 * st
dair::HOURL_CONVERTED_IN_SECONDS,
00790                                                         0.8));
00791         lBKKPrefDepTimeProbDist.
00792             insert (PreferredDepartureTimeContinuousDistribution_T::value_type (16 * st
dair::HOURL_CONVERTED_IN_SECONDS,
00793                                                         1.0));
00794
00795         //

```

```

00796     const stdair::WTP_T lBKKHKGWTP (400.0);
00797
00798
00799     // Delegate the call to the dedicated command
00800     DemandStream& lBKKHKGDemandStream =
00801         createDemandStream (ioSEVMGR_ServicePtr, lBKKHKGDemandStreamKey, lDTPProbDi
00802 st,
00803                             lBKKHKGPOSProbDist, lChannelProbDist, lTripProbDist,
00804                             lStayProbDist, lFFProbDist,
00805                             lChangeFees, lChangeFeeDisutility,
00806                             lNonRefundable, lNonRefundableDisutility,
00807                             lBKKHPrefDepTimeProbDist,
00808                             lBKKHKGWTP, lTimeValueProbDist,
00809                             lBKKHKGDemandDistribution,
00810                             ioSharedGenerator.getBaseGenerator(),
00811                             lBKKHKGRequestDateTimeSeed,
00812                             lBKKHKGDemandCharacteristicsSeed, iPOSProbMass);
00813
00814     // Calculate the expected total number of events for the current
00815     // demand stream
00816     const stdair::NbOfRequests_T& lBKKHKGExpectedNbOfEvents =
00817         lBKKHKGDemandStream.getMeanNumberOfRequests();
00818
00819     /*=====*/
00820     // Key of the demand stream
00821
00822     //
00823     const DemandStreamKey lSINHKGDemandStreamKey (lSINOrigin, lHKGDestination, lD
00824 epDate,
00825                                                  lCabin);
00826
00827     // DEBUG
00828     // STDAIR_LOG_DEBUG ("Demand stream key: " << lDemandStreamKey.describe());
00829
00830     // Distribution for the number of requests
00831     const stdair::MeanValue_T lSINHKGDemandMean (60.0);
00832     const stdair::StdDevValue_T lSINHKGDemandStdDev (4.0);
00833     const DemandDistribution lSINHKGDemandDistribution (lSINHKGDemandMean, lSINHKG
00834 DemandStdDev);
00835
00836     // Seed
00837     const stdair::RandomSeed_T& lSINHKGRequestDateTimeSeed =
00838         generateSeed (ioSharedGenerator);
00839     const stdair::RandomSeed_T& lSINHKGDemandCharacteristicsSeed =
00840         generateSeed (ioSharedGenerator);
00841
00842     //
00843     POSProbabilityMassFunction_T lSINHKGPOSProbDist;
00844     lSINHKGPOSProbDist.insert (POSProbabilityMassFunction_T::value_type ("SIN", 1
00845 .0));
00846     lSINHKGPOSProbDist.insert (POSProbabilityMassFunction_T::value_type ("HKG", 0
00847 .0));
00848
00849     //
00850     const stdair::WTP_T lSINHKGWTP (750.0);
00851
00852     // Delegate the call to the dedicated command
00853     DemandStream& lSINHKGDemandStream =
00854         createDemandStream (ioSEVMGR_ServicePtr, lSINHKGDemandStreamKey, lDTPProbDi
00855 st,
00856                             lSINHKGPOSProbDist, lChannelProbDist, lTripProbDist,
00857                             lStayProbDist, lFFProbDist,
00858                             lChangeFees, lChangeFeeDisutility,

```

```
00856             lNonRefundable, lNonRefundableDisutility,
00857             lSINPrefDepTimeProbDist,
00858             lSINHKGWTP, lTimeValueProbDist, lSINHKGDemandDistributi
on,
00859             ioSharedGenerator.getBaseGenerator(),
00860             lSINHKGRequestDateTimeSeed,
00861             lSINHKGDemandCharacteristicsSeed, iPOSProbMass);
00862
00863     // Calculate the expected total number of events for the current
00864     // demand stream
00865     const stdair::NbOfRequests_T& lSINHKGExpectedNbOfEvents =
00866         lSINHKGDemandStream.getMeanNumberOfRequests();
00867
00868     /*=====
=====*/
00869
00873     const stdair::NbOfRequests_T lExpectedTotalNbOfEvents =
00874         lSINBKKExpectedNbOfEvents + lBKKHKExpectedNbOfEvents + lSINHKGExpectedNbOf
Events;
00875     ioSEVMGR_ServicePtr->addStatus (stdair::EventType::BKG_REQ,
00876                                     lExpectedTotalNbOfEvents);
00877 }
00878
00879 }
00880
```

25.83 trademgen/command/DemandManager.hpp File Reference

```
#include <stdair/stdair_basic_types.hpp>
#include <stdair/basic/RandomGeneration.hpp>
#include <stdair/basic/DemandGenerationMethod.hpp>
#include <stdair/bom/BookingRequestTypes.hpp>
#include <stdair/command/CmdAbstract.hpp>
#include <sevmgr/SEVMGR_Types.hpp>
#include <trademgen/TRADEMGEN_Types.hpp>
#include <trademgen/basic/DemandCharacteristicsTypes.hpp>
#include <trademgen/bom/DemandStreamKey.hpp>
```

Classes

- class [TRADEMGEN::DemandManager](#)
Utility class for Demand and [DemandStream](#) objects.

Namespaces

- namespace [stdair](#)
Forward declarations.
- namespace [TRADEMGEN](#)
- namespace [TRADEMGEN::DemandParserHelper](#)

25.84 DemandManager.hpp

```

00001 #ifndef __TRADEMGEN_CMD_DEMANDMANAGER_HPP
00002 #define __TRADEMGEN_CMD_DEMANDMANAGER_HPP
00003
00004 // //////////////////////////////////////
00005 // Import section
00006 // //////////////////////////////////////
00007 // StdAir
00008 #include <stdair/stdair_basic_types.hpp>
00009 #include <stdair/basic/RandomGeneration.hpp>
00010 #include <stdair/basic/DemandGenerationMethod.hpp>
00011 #include <stdair/bom/BookingRequestTypes.hpp>
00012 #include <stdair/command/CmdAbstract.hpp>
00013 // SEVMgr
00014 #include <sevmgr/SEVMGR_Types.hpp>
00015 // TraDemGen
00016 #include <trademgen/TRADEMGEN_Types.hpp>
00017 #include <trademgen/basic/DemandCharacteristicsTypes.hpp>
00018 #include <trademgen/bom/DemandStreamKey.hpp>
00019
00020 // Forward declarations
00021 namespace stdair {
00022     class EventStruct;
00023     struct ProgressStatusSet;
00024     struct TravelSolutionStruct;
00025 }
00026
00027 namespace TRADEMGEN {
00028
00029     // Forward declarations
00030     struct DemandDistribution;
00031     struct DemandStruct;
00032     class DemandStream;
00033     namespace DemandParserHelper {
00034         struct doEndDemand;
00035     }
00036
00037     class DemandManager : public stdair::CmdAbstract {
00038     friend struct DemandParserHelper::doEndDemand;
00039     friend class TRADEMGEN_Service;
00040
00041     private:
00042         // ////////// Business methodes //////////
00043         static void buildSampleBomStd (SEVMGR::SEVMGR_ServicePtr_T,
00044                                     stdair::RandomGeneration&,
00045                                     const POSProbabilityMass_T&);
00046
00047         // Demand sample bom for partnerships study.
00048         static void buildSampleBom (SEVMGR::SEVMGR_ServicePtr_T,
00049                                     stdair::RandomGeneration&,
00050                                     const POSProbabilityMass_T&);
00051
00052         static void createDemandCharacteristics (SEVMGR::SEVMGR_ServicePtr_T,
00053                                                 stdair::RandomGeneration&,
00054                                                 const POSProbabilityMass_T&,
00055                                                 const DemandStruct&);
00056
00057         static stdair::RandomSeed_T generateSeed (stdair::RandomGeneration&);
00058
00059         static DemandStream&
00060         createDemandStream (SEVMGR::SEVMGR_ServicePtr_T,
00061                             const DemandStreamKey&,
00062                             const ArrivalPatternCumulativeDistribution_T&,
00063                             const POSProbabilityMassFunction_T&,
00064                             const ChannelProbabilityMassFunction_T&,
00065                             const TripTypeProbabilityMassFunction_T&,

```



```

00184         const StayDurationProbabilityMassFunction_T&,
00185         const FrequentFlyerProbabilityMassFunction_T&,
00186         const stdair::ChangeFeesRatio_T&,
00187         const stdair::Disutility_T&,
00188         const stdair::NonRefundableRatio_T&,
00189         const stdair::Disutility_T&,
00190         const PreferredDepartureTimeContinuousDistribution_T&,
00191         const stdair::WTP_T&,
00192         const ValueOfTimeContinuousDistribution_T&,
00193         const DemandDistribution&, stdair::BaseGenerator_T&,
00194         const stdair::RandomSeed_T&,
00195         const stdair::RandomSeed_T&,
00196         const POSProbabilityMass_T&);
00197
00215     static const bool
00216     stillHavingRequestsToBeGenerated (SEVMGR::SEVMGR_ServicePtr_T,
00217         const stdair::DemandStreamKeyStr_T&,
00218         stdair::ProgressStatusSet&,
00219         const stdair::DemandGenerationMethod&);
00220
00236     static stdair::Count_T generateFirstRequests (SEVMGR::SEVMGR_ServicePtr_T,
00237         stdair::RandomGeneration&,
00238         const stdair::DemandGenerationM
00239         ethod&);
00239
00268     static stdair::BookingRequestPtr_T
00269     generateNextRequest (SEVMGR::SEVMGR_ServicePtr_T, stdair::RandomGeneration&,
00270         const stdair::DemandStreamKeyStr_T&,
00271         const stdair::DemandGenerationMethod&);
00272
00282     static void reset (SEVMGR::SEVMGR_ServicePtr_T, stdair::BaseGenerator_T&);
00283
00287     static bool generateCancellation (stdair::RandomGeneration&,
00288         const stdair::TravelSolutionStruct&,
00289         const stdair::PartySize_T&,
00290         const stdair::DateTime_T&,
00291         const stdair::Date_T&,
00292         stdair::EventStruct& ioEventStruct);
00293 };
00294
00295 }
00296 #endif // __TRADEMGEN_CMD_DEMANDMANAGER_HPP

```

25.85 trademgen/command/DemandParser.cpp File Reference

```
#include <cassert>
#include <stdair/basic/BasFileMgr.hpp>
#include <stdair/basic/RandomGeneration.hpp>
#include <stdair/bom/Inventory.hpp>
#include <trademgen/command/DemandParserHelper.hpp>
#include <trademgen/command/DemandParser.hpp>
```

Namespaces

- namespace [TRADEMGEN](#)

25.86 DemandParser.cpp

```

00001 // //////////////////////////////////////
00002 // Import section
00003 // //////////////////////////////////////
00004 // STL
00005 #include <cassert>
00006 // StdAir
00007 #include <stdair/basic/BasFileMgr.hpp>
00008 #include <stdair/basic/RandomGeneration.hpp>
00009 #include <stdair/bom/Inventory.hpp>
00010 // TraDemGen
00011 #include <trademgen/command/DemandParserHelper.hpp>
00012 #include <trademgen/command/DemandParser.hpp>
00013
00014 namespace TRADEMGEN {
00015
00016 // //////////////////////////////////////
00017 void DemandParser::
00018 generateDemand (const DemandFilePath& iDemandFilename,
00019                SEVMGR::SEVMGR_ServicePtr_T ioSEVMGR_ServicePtr,
00020                stdair::RandomGeneration& ioSharedGenerator,
00021                const POSProbabilityMass_T& iDefaultPOSProbabilityMass) {
00022
00023     const stdair::Filename_T lFilename = iDemandFilename.name();
00024
00025     // Check that the file path given as input corresponds to an actual file
00026     const bool doesExistAndIsReadable =
00027         stdair::BasFileMgr::doesExistAndIsReadable (lFilename);
00028     if (doesExistAndIsReadable == false) {
00029         STDAIR_LOG_ERROR ("The demand input file '" << lFilename
00030                          << "' does not exist or can not be read");
00031
00032         throw DemandInputFileNotFoundException ("The demand file '" + lFilename
00033                                                 + "' does not exist or can not "
00034                                                 "be read");
00035     }
00036
00037     // Initialise the demand file parser.
00038     DemandFileParser lDemandParser (ioSEVMGR_ServicePtr, ioSharedGenerator,
00039                                     iDefaultPOSProbabilityMass, lFilename);
00040
00041     // Parse the CSV-formatted demand input file, and generate the
00042     // corresponding DemandCharacteristic objects.
00043     lDemandParser.generateDemand();
00044 }
00045
00046 }

```

25.87 trademgen/command/DemandParser.hpp File Reference

```
#include <string>
#include <stdair/stdair_basic_types.hpp>
#include <stdair/command/CmdAbstract.hpp>
#include <sevmgr/SEVMGR_Types.hpp>
#include <trademgen/TRADEMGEN_Types.hpp>
#include <trademgen/basic/DemandCharacteristicsTypes.hpp>
```

Classes

- class [TRADEMGEN::DemandParser](#)
Class wrapping the parser entry point.

Namespaces

- namespace [stdair](#)
Forward declarations.
- namespace [TRADEMGEN](#)

25.88 DemandParser.hpp

```

00001 #ifndef __TRADEMGEN_CMD_DEMANDPARSER_HPP
00002 #define __TRADEMGEN_CMD_DEMANDPARSER_HPP
00003
00004 // //////////////////////////////////////
00005 // Import section
00006 // //////////////////////////////////////
00007 // STL
00008 #include <string>
00009 // StdAir
00010 #include <stdair/stdair_basic_types.hpp>
00011 #include <stdair/command/CmdAbstract.hpp>
00012 // SEvMgr
00013 #include <sevmgr/SEVMGR_Types.hpp>
00014 // TraDemGen
00015 #include <trademgen/TRADEMGEN_Types.hpp>
00016 #include <trademgen/basic/DemandCharacteristicsTypes.hpp>
00017
00019 namespace stdair {
00020     struct RandomGeneration;
00021 }
00022
00023 namespace TRADEMGEN {
00024
00028     class DemandParser : public stdair::CmdAbstract {
00029     public:
00043         static void generateDemand (const DemandFilePath&,
00044                                     SEVMGR::SEVMGR_ServicePtr_T,
00045                                     stdair::RandomGeneration&,
00046                                     const POSProbabilityMass_T&);
00047     };
00048 }
00049 #endif // __TRADEMGEN_CMD_DEMANDPARSER_HPP

```

25.89 trademgen/command/DemandParserHelper.cpp File Reference

```
#include <cassert>
#include <stdair/basic/RandomGeneration.hpp>
#include <stdair/basic/BasFileMgr.hpp>
#include <stdair/service/Logger.hpp>
#include <trademgen/basic/DemandCharacteristicsTypes.hpp>
#include <trademgen/command/DemandParserHelper.hpp>
#include <trademgen/command/DemandManager.hpp>
```

Namespaces

- namespace [TRADEMGEN](#)
- namespace [TRADEMGEN::DemandParserHelper](#)

Functions

- repeat_p_t [TRADEMGEN::DemandParserHelper::airline_code_p](#) (chset_t("0-9A-Z").derived(), 2, 3)
- bounded1_4_p_t [TRADEMGEN::DemandParserHelper::flight_number_p](#) (uint1_4_p.derived(), 0u, 9999u)
- bounded4_p_t [TRADEMGEN::DemandParserHelper::year_p](#) (uint4_p.derived(), 2000u, 2099u)
- bounded2_p_t [TRADEMGEN::DemandParserHelper::month_p](#) (uint2_p.derived(), 1u, 12u)
- bounded2_p_t [TRADEMGEN::DemandParserHelper::day_p](#) (uint2_p.derived(), 1u, 31u)
- repeat_p_t [TRADEMGEN::DemandParserHelper::dow_p](#) (chset_t("0-1").derived().derived(), 7, 7)
- repeat_p_t [TRADEMGEN::DemandParserHelper::airport_p](#) (chset_t("0-9A-Z").derived(), 3, 3)
- bounded1_2_p_t [TRADEMGEN::DemandParserHelper::hours_p](#) (uint1_2_p.derived(), 0u, 23u)
- bounded2_p_t [TRADEMGEN::DemandParserHelper::minutes_p](#) (uint2_p.derived(), 0u, 59u)
- bounded2_p_t [TRADEMGEN::DemandParserHelper::seconds_p](#) (uint2_p.derived(), 0u, 59u)
- chset_t [TRADEMGEN::DemandParserHelper::cabin_code_p](#) ("A-Z")
- chset_t [TRADEMGEN::DemandParserHelper::passenger_type_p](#) ("A-Z")
- chset_t [TRADEMGEN::DemandParserHelper::ff_type_p](#) ("A-Z")
- repeat_p_t [TRADEMGEN::DemandParserHelper::class_code_list_p](#) (chset_t("A-Z").derived(), 1, 26)
- bounded1_3_p_t [TRADEMGEN::DemandParserHelper::stay_duration_p](#) (uint1_3_p.derived(), 0u, 999u)

Variables

- int1_p_t [TRADEMGEN::DemandParserHelper::int1_p](#)
- uint2_p_t [TRADEMGEN::DemandParserHelper::uint2_p](#)
- uint1_2_p_t [TRADEMGEN::DemandParserHelper::uint1_2_p](#)
- uint1_3_p_t [TRADEMGEN::DemandParserHelper::uint1_3_p](#)
- uint4_p_t [TRADEMGEN::DemandParserHelper::uint4_p](#)
- uint1_4_p_t [TRADEMGEN::DemandParserHelper::uint1_4_p](#)
- int1_p_t [TRADEMGEN::DemandParserHelper::family_code_p](#)

25.90 DemandParserHelper.cpp

```

00001 // //////////////////////////////////////
00002 // Import section
00003 // //////////////////////////////////////
00004 // STL
00005 #include <cassert>
00006 // StdAir
00007 #include <stdair/basic/RandomGeneration.hpp>
00008 #include <stdair/basic/BasFileMgr.hpp>
00009 #include <stdair/service/Logger.hpp>
00010 // TraDemGen
00011 #include <trademgen/basic/DemandCharacteristicsTypes.hpp>
00012 //#define BOOST_SPIRIT_DEBUG
00013 #include <trademgen/command/DemandParserHelper.hpp>
00014 #include <trademgen/command/DemandManager.hpp>
00015
00016 namespace bsc = boost::spirit::classic;
00017
00018 namespace TRADEMGEN {
00019
00020     namespace DemandParserHelper {
00021
00022         // //////////////////////////////////////
00023         // Semantic actions
00024         // //////////////////////////////////////
00025
00026         ParserSemanticAction::ParserSemanticAction (DemandStruct& ioDemand)
00027             : _demand (ioDemand) {
00028         }
00029
00030         // //////////////////////////////////////
00031         storePrefDepDateRangeStart::
00032         storePrefDepDateRangeStart (DemandStruct& ioDemand)
00033             : ParserSemanticAction (ioDemand) {
00034         }
00035
00036         // //////////////////////////////////////
00037         void storePrefDepDateRangeStart::operator() (iterator_t iStr,
00038             iterator_t iStrEnd) const {
00039             _demand._prefDepDateStart = _demand.getDate();
00040
00041             // Reset the number of seconds
00042             _demand._itSeconds = 0;
00043         }
00044
00045         // //////////////////////////////////////
00046         storePrefDepDateRangeEnd::
00047         storePrefDepDateRangeEnd (DemandStruct& ioDemand)
00048             : ParserSemanticAction (ioDemand) {
00049         }
00050
00051         // //////////////////////////////////////
00052         void storePrefDepDateRangeEnd::operator() (iterator_t iStr,
00053             iterator_t iStrEnd) const {
00054             // As a Boost date period (DatePeriod_T) defines the last day of
00055             // the period to be end-date - one day, we have to add one day to that
00056             // end date before.
00057             const stdair::DateOffset_T oneDay (1);
00058             _demand._prefDepDateEnd = _demand.getDate() + oneDay;
00059
00060             // Transform the date pair (i.e., the date range) into a date period
00061             _demand._dateRange =
00062                 stdair::DatePeriod_T (_demand._prefDepDateStart,
00063                     _demand._prefDepDateEnd);
00064
00065             // Reset the number of seconds

```

```

00066     _demand._itSeconds = 0;
00067 }
00068
00069 // //////////////////////////////////////
00070 storeDow::storeDow (DemandStruct& ioDemand)
00071 : ParserSemanticAction (ioDemand) {
00072 }
00073
00074 // //////////////////////////////////////
00075 void storeDow::operator() (iterator_t iStr, iterator_t iStrEnd) const {
00076     stdair::DOW_String_T lDow (iStr, iStrEnd);
00077     _demand._dow = lDow;
00078 }
00079
00080 // //////////////////////////////////////
00081 storeOrigin::storeOrigin (DemandStruct& ioDemand)
00082 : ParserSemanticAction (ioDemand) {
00083 }
00084
00085 // //////////////////////////////////////
00086 void storeOrigin::operator() (iterator_t iStr, iterator_t iStrEnd) const {
00087     stdair::AirportCode_T lOrigin (iStr, iStrEnd);
00088     _demand._origin = lOrigin;
00089 }
00090
00091 // //////////////////////////////////////
00092 storeDestination::storeDestination (DemandStruct& ioDemand)
00093 : ParserSemanticAction (ioDemand) {
00094 }
00095
00096 // //////////////////////////////////////
00097 void storeDestination::operator() (iterator_t iStr,
00098                                     iterator_t iStrEnd) const {
00099     stdair::AirportCode_T lDestination (iStr, iStrEnd);
00100     _demand._destination = lDestination;
00101 }
00102
00103 // //////////////////////////////////////
00104 storePrefCabin::storePrefCabin (DemandStruct& ioDemand)
00105 : ParserSemanticAction (ioDemand) {
00106 }
00107
00108 // //////////////////////////////////////
00109 void storePrefCabin::operator() (iterator_t iStr,
00110                                     iterator_t iStrEnd) const {
00111     stdair::CabinCode_T lPrefCabin (iStr, iStrEnd);
00112     _demand._prefCabin = lPrefCabin;
00113     //STDAIR_LOG_DEBUG ("Preferred cabin: " << lPrefCabin);
00114 }
00115
00116 // //////////////////////////////////////
00117 storeDemandMean::storeDemandMean (DemandStruct& ioDemand)
00118 : ParserSemanticAction (ioDemand) {
00119 }
00120
00121 // //////////////////////////////////////
00122 void storeDemandMean::operator() (double iReal) const {
00123     _demand._demandMean = iReal;
00124     //STDAIR_LOG_DEBUG ("Demand mean: " << iReal);
00125 }
00126
00127 // //////////////////////////////////////
00128 storeDemandStdDev::storeDemandStdDev (DemandStruct& ioDemand)
00129 : ParserSemanticAction (ioDemand) {
00130 }
00131
00132 // //////////////////////////////////////

```



```

00133     void storeDemandStdDev::operator() (double iReal) const {
00134         _demand._demandStdDev = iReal;
00135         //STDAIR_LOG_DEBUG ("Demand stddev: " << iReal);
00136     }
00137
00138     // //////////////////////////////////////
00139     storeDemandChangeFeeProb::storeDemandChangeFeeProb (DemandStruct& ioDemand)
00140         : ParserSemanticAction (ioDemand) {
00141     }
00142
00143     // //////////////////////////////////////
00144     void storeDemandChangeFeeProb::operator() (double iReal) const {
00145         _demand._changeFeeProb = iReal;
00146         //STDAIR_LOG_DEBUG ("Demand change fee prob: " << iReal);
00147     }
00148
00149     // //////////////////////////////////////
00150     storeDemandChangeFeeDisutility::storeDemandChangeFeeDisutility (DemandStruct&
ioDemand)
00151         : ParserSemanticAction (ioDemand) {
00152     }
00153
00154     // //////////////////////////////////////
00155     void storeDemandChangeFeeDisutility::operator() (double iReal) const {
00156         _demand._changeFeeDisutility = iReal;
00157         //STDAIR_LOG_DEBUG ("Demand change fee disutility: " << iReal);
00158     }
00159
00160     // //////////////////////////////////////
00161     storeDemandNonRefundableProb::
00162     storeDemandNonRefundableProb (DemandStruct& ioDemand)
00163         : ParserSemanticAction (ioDemand) {
00164     }
00165
00166     // //////////////////////////////////////
00167     void storeDemandNonRefundableProb::operator() (double iReal) const {
00168         _demand._nonRefundableProb = iReal;
00169         //STDAIR_LOG_DEBUG ("Demand non refundable prob: " << iReal);
00170     }
00171
00172     // //////////////////////////////////////
00173     storeDemandNonRefundableDisutility::
00174     storeDemandNonRefundableDisutility (DemandStruct& ioDemand)
00175         : ParserSemanticAction (ioDemand) {
00176     }
00177
00178     // //////////////////////////////////////
00179     void storeDemandNonRefundableDisutility::operator() (double iReal) const {
00180         _demand._nonRefundableDisutility = iReal;
00181         //STDAIR_LOG_DEBUG ("Demand non refundable disutility: " << iReal);
00182     }
00183
00184     // //////////////////////////////////////
00185     storePosCode::storePosCode (DemandStruct& ioDemand)
00186         : ParserSemanticAction (ioDemand) {
00187     }
00188
00189     // //////////////////////////////////////
00190     void storePosCode::operator() (iterator_t iStr, iterator_t iStrEnd) const {
00191         const stdair::AirportCode_T lPosCode (iStr, iStrEnd);
00192         _demand._itPosCode = lPosCode;
00193         //STDAIR_LOG_DEBUG ("Pos code: " << lPosCode);
00194     }
00195
00196     // //////////////////////////////////////
00197     storePosProbMass::storePosProbMass (DemandStruct& ioDemand)
00198         : ParserSemanticAction (ioDemand) {

```

```

00199     }
00200
00201     // ////////////////////////////////////////
00202     void storePosProbMass::operator() (double iReal) const {
00203         const bool hasInsertBeenSuccessfull =
00204             _demand._posProbDist.
00205             insert (POSProbabilityMassFunction_T::
00206                 value_type (_demand._itPosCode, iReal)).second;
00207         if (hasInsertBeenSuccessfull == false) {
00208             STDAIR_LOG_ERROR ("The same POS code ('" << _demand._itPosCode
00209                 << "') has probably been given twice");
00210             throw stdair::CodeDuplicationException ("The same POS code ('"
00211                 + _demand._itPosCode
00212                 + "') has probably been given twi
00213 ce");
00214         }
00215         //STDAIR_LOG_DEBUG ("PosProbMass: " << iReal);
00216     }
00217
00218     // ////////////////////////////////////////
00219     storeChannelCode::storeChannelCode (DemandStruct& ioDemand)
00220         : ParserSemanticAction (ioDemand) {
00221     }
00222
00223     // ////////////////////////////////////////
00224     void storeChannelCode::operator() (iterator_t iStr,
00225         iterator_t iStrEnd) const {
00226         _demand._itChannelCode = std::string (iStr, iStrEnd);
00227         //STDAIR_LOG_DEBUG ("Channel code: " << _demand._itChannelCode);
00228     }
00229
00230     // ////////////////////////////////////////
00231     storeChannelProbMass::storeChannelProbMass (DemandStruct& ioDemand)
00232         : ParserSemanticAction (ioDemand) {
00233     }
00234
00235     // ////////////////////////////////////////
00236     void storeChannelProbMass::operator() (double iReal) const {
00237         const bool hasInsertBeenSuccessfull =
00238             _demand._channelProbDist.
00239             insert (ChannelProbabilityMassFunction_T::
00240                 value_type (_demand._itChannelCode, iReal)).second;
00241         if (hasInsertBeenSuccessfull == false) {
00242             STDAIR_LOG_ERROR ("The same channel type code ('"
00243                 << _demand._itChannelCode
00244                 << "') has probably been given twice");
00245             throw stdair::CodeDuplicationException ("The same channel type code ('"
00246                 + _demand._itChannelCode
00247                 + "') has probably been given twi
00248 ce");
00249         }
00250         //STDAIR_LOG_DEBUG ("ChannelProbMass: " << iReal);
00251     }
00252
00253     // ////////////////////////////////////////
00254     storeTripCode::storeTripCode (DemandStruct& ioDemand)
00255         : ParserSemanticAction (ioDemand) {
00256     }
00257
00258     // ////////////////////////////////////////
00259     void storeTripCode::operator() (iterator_t iStr,
00260         iterator_t iStrEnd) const {
00261         _demand._itTripCode = std::string (iStr, iStrEnd);
00262         //STDAIR_LOG_DEBUG ("Trip code: " << _demand._itTripCode);
00263     }

```

```

00264
00265 // //////////////////////////////////////
00266 storeTripProbMass::storeTripProbMass (DemandStruct& ioDemand)
00267 : ParserSemanticAction (ioDemand) {
00268 }
00269
00270 // //////////////////////////////////////
00271 void storeTripProbMass::operator() (double iReal) const {
00272     const bool hasInsertBeenSuccessfull =
00273         _demand._tripProbDist.
00274         insert (TripTypeProbabilityMassFunction_T::
00275             value_type (_demand._itTripCode, iReal)).second;
00276     if (hasInsertBeenSuccessfull == false) {
00277         STDAIR_LOG_ERROR ("The same trip type code ('"
00278             << _demand._itTripCode
00279             << "') has probably been given twice");
00280         throw stdair::CodeDuplicationException ("The same trip type code ('"
00281             + _demand._itTripCode
00282             + "') has probably been given twi
00283 ce");
00284     }
00285     //STDAIR_LOG_DEBUG ("TripProbMass: " << iReal);
00286 }
00287
00288 // //////////////////////////////////////
00289 storeStayCode::storeStayCode (DemandStruct& ioDemand)
00290 : ParserSemanticAction (ioDemand) {
00291 }
00292
00293 // //////////////////////////////////////
00294 void storeStayCode::operator() (unsigned int iInteger) const {
00295     const stdair::DayDuration_T lStayDuration (iInteger);
00296     _demand._itStayDuration = lStayDuration;
00297     // STDAIR_LOG_DEBUG ("Stay duration: " << lStayDuration);
00298 }
00299
00300 // //////////////////////////////////////
00301 storeStayProbMass::storeStayProbMass (DemandStruct& ioDemand)
00302 : ParserSemanticAction (ioDemand) {
00303 }
00304
00305 // //////////////////////////////////////
00306 void storeStayProbMass::operator() (double iReal) const {
00307     const bool hasInsertBeenSuccessfull =
00308         _demand._stayProbDist.
00309         insert (StayDurationProbabilityMassFunction_T::
00310             value_type (_demand._itStayDuration, iReal)).second;
00311     if (hasInsertBeenSuccessfull == false) {
00312         std::ostringstream oStr;
00313         oStr << "The same stay duration ('" << _demand._itStayDuration
00314             << "') has probably been given twice";
00315         STDAIR_LOG_ERROR (oStr.str());
00316         throw stdair::CodeDuplicationException (oStr.str());
00317     }
00318
00319     // STDAIR_LOG_DEBUG ("StayProbMass: " << iReal);
00320 }
00321
00322 // //////////////////////////////////////
00323 storeFFCode::storeFFCode (DemandStruct& ioDemand)
00324 : ParserSemanticAction (ioDemand) {
00325 }
00326
00327 // //////////////////////////////////////
00328 void storeFFCode::operator() (iterator_t iStr, iterator_t iStrEnd) const {
00329     _demand._itFFCode = std::string (iStr, iStrEnd);

```

```

00330         //STDAIR_LOG_DEBUG ("FF code: " << _demand._itFFCode);
00331     }
00332
00333     // //////////////////////////////////////
00334     storeFFProbMass::storeFFProbMass (DemandStruct& ioDemand)
00335         : ParserSemanticAction (ioDemand) {
00336     }
00337
00338     // //////////////////////////////////////
00339     void storeFFProbMass::operator() (double iReal) const {
00340         const bool hasInsertBeenSuccessfull =
00341             _demand._ffProbDist.
00342             insert (FrequentFlyerProbabilityMassFunction_T::
00343                 value_type (_demand._itFFCode, iReal)).second;
00344         if (hasInsertBeenSuccessfull == false) {
00345             STDAIR_LOG_ERROR ("The same Frequent Flyer code ('"
00346                 << _demand._itFFCode
00347                 << "') has probably been given twice");
00348             throw stdair::CodeDuplicationException("The same Frequent Flyer code ('"
00349                 + _demand._itFFCode
00350                 + "') has probably been given twic
00351 e");
00352         }
00353         //STDAIR_LOG_DEBUG ("FfProbMass: " << iReal);
00354     }
00355
00356     // //////////////////////////////////////
00357     storePrefDepTime::storePrefDepTime (DemandStruct& ioDemand)
00358         : ParserSemanticAction (ioDemand) {
00359     }
00360
00361     // //////////////////////////////////////
00362     void storePrefDepTime::operator() (iterator_t iStr,
00363         iterator_t iStrEnd) const {
00364         _demand._itPrefDepTime = _demand.getTime();
00365
00366         // DEBUG
00367         // STDAIR_LOG_DEBUG ("Pref dep time: " << _demand._itHours << ":"
00368             << _demand._itMinutes << ":" << _demand._itSeconds
00369             << " ==> " << _demand._itPrefDepTime);
00370
00371         // Reset the number of minutes and seconds
00372         _demand._itMinutes = 0;
00373         _demand._itSeconds = 0;
00374     }
00375
00376     // //////////////////////////////////////
00377     storePrefDepTimeProbMass::storePrefDepTimeProbMass (DemandStruct& ioDemand)
00378         : ParserSemanticAction (ioDemand) {
00379     }
00380
00381     // //////////////////////////////////////
00382     void storePrefDepTimeProbMass::operator() (double iReal) const {
00383         const stdair::IntDuration_T lIntDuration =
00384             _demand._itPrefDepTime.total_seconds();
00385
00386         _demand._prefDepTimeProbDist.
00387             insert (PreferredDepartureTimeContinuousDistribution_T::
00388                 value_type (lIntDuration, iReal));
00389         //STDAIR_LOG_DEBUG ("PrefDepTimeProbMass: " << iReal);
00390     }
00391
00392     // //////////////////////////////////////
00393     storeWTP::storeWTP (DemandStruct& ioDemand)
00394         : ParserSemanticAction (ioDemand) {
00395     }

```

```

00396
00397 ///////////////////////////////////////////////////////////////////
00398 void storeWTP::operator() (double iReal) const {
00399     _demand._minWTP = iReal;
00400     //STDAIR_LOG_DEBUG ("WTP: " << iReal);
00401 }
00402
00403 ///////////////////////////////////////////////////////////////////
00404 storeTimeValue::storeTimeValue (DemandStruct& ioDemand)
00405     : ParserSemanticAction (ioDemand) {
00406 }
00407
00408 ///////////////////////////////////////////////////////////////////
00409 void storeTimeValue::operator() (double iReal) const {
00410     _demand._itTimeValue = iReal;
00411     //STDAIR_LOG_DEBUG ("Time value: " << iReal);
00412 }
00413
00414 ///////////////////////////////////////////////////////////////////
00415 storeTimeValueProbMass::storeTimeValueProbMass (DemandStruct& ioDemand)
00416     : ParserSemanticAction (ioDemand) {
00417 }
00418
00419 ///////////////////////////////////////////////////////////////////
00420 void storeTimeValueProbMass::operator() (double iReal) const {
00421     _demand._timeValueProbDist.
00422         insert (ValueOfTimeContinuousDistribution_T::
00423             value_type (_demand._itTimeValue, iReal));
00424     //STDAIR_LOG_DEBUG ("TimeValueProbMass: " << iReal);
00425 }
00426
00427 ///////////////////////////////////////////////////////////////////
00428 storeDTD::storeDTD (DemandStruct& ioDemand)
00429     : ParserSemanticAction (ioDemand) {
00430 }
00431
00432 ///////////////////////////////////////////////////////////////////
00433 void storeDTD::operator() (unsigned int iInteger) const {
00434     const stdair::DayDuration_T lDTD (iInteger);
00435     _demand._itDTD = lDTD;
00436     //STDAIR_LOG_DEBUG ("DTD: " << lDTD);
00437 }
00438
00439 ///////////////////////////////////////////////////////////////////
00440 storeDTDProbMass::storeDTDProbMass (DemandStruct& ioDemand)
00441     : ParserSemanticAction (ioDemand) {
00442 }
00443
00444 ///////////////////////////////////////////////////////////////////
00445 void storeDTDProbMass::operator() (double iReal) const {
00446     const stdair::FloatDuration_T lZeroDTDFloat = 0.0;
00447     stdair::FloatDuration_T lDTDFloat =
00448         static_cast<stdair::FloatDuration_T> (_demand._itDTD);
00449     lDTDFloat = lZeroDTDFloat - lDTDFloat;
00450
00451     _demand._dtdProbDist.insert (ArrivalPatternCumulativeDistribution_T::
00452         value_type (lDTDFloat, iReal));
00453     //STDAIR_LOG_DEBUG ("DTDProbMass: " << iReal);
00454 }
00455
00456 ///////////////////////////////////////////////////////////////////
00457 doEndDemand::doEndDemand (SEVMGR::SEVMGR_ServicePtr_T ioSEVMGR_ServicePtr,
00458     stdair::RandomGeneration& ioSharedGenerator,
00459     const POSProbabilityMass_T& iPOSProbMass,
00460     DemandStruct& ioDemand)
00461     : ParserSemanticAction (ioDemand),
00462     _sevmgrServicePtr (ioSEVMGR_ServicePtr),

```

```

00463         _uniformGenerator (ioSharedGenerator),
00464         _posProbabilityMass (iPOSProbMass) {
00465     }
00466
00467     // //////////////////////////////////////
00468     // void doEndDemand::operator() (char iChar) const {
00469     void doEndDemand::operator() (iterator_t iStr, iterator_t iStrEnd) const {
00470
00471         // DEBUG: Display the result
00472         // STDAIR_LOG_DEBUG ("Demand: " << _demand.describe());
00473
00474         // Create the Demand BOM objects
00475         DemandManager::createDemandCharacteristics (_sevmgrServicePtr,
00476                                                     _uniformGenerator,
00477                                                     _posProbabilityMass, _demand);
00478
00479         // Clean the lists
00480         _demand._posProbDist.clear();
00481         _demand._channelProbDist.clear();
00482         _demand._tripProbDist.clear();
00483         _demand._stayProbDist.clear();
00484         _demand._ffProbDist.clear();
00485         _demand._prefDepTimeProbDist.clear();
00486         _demand._timeValueProbDist.clear();
00487         _demand._dtdProbDist.clear();
00488     }
00489
00490
00491     // //////////////////////////////////////
00492     //
00493     // Utility Parsers
00494     //
00495     // //////////////////////////////////////
00497     int1_p_t int1_p;
00498
00500     uint2_p_t uint2_p;
00501
00503     uint1_2_p_t uint1_2_p;
00504
00506     uint1_3_p_t uint1_3_p;
00507
00509     uint4_p_t uint4_p;
00510
00512     uint1_4_p_t uint1_4_p;
00513
00515     repeat_p_t airline_code_p (chset_t("0-9A-Z").derived(), 2, 3);
00516
00518     bounded1_4_p_t flight_number_p (uint1_4_p.derived(), 0u, 9999u);
00519
00521     bounded4_p_t year_p (uint4_p.derived(), 2000u, 2099u);
00522
00524     bounded2_p_t month_p (uint2_p.derived(), 1u, 12u);
00525
00527     bounded2_p_t day_p (uint2_p.derived(), 1u, 31u);
00528
00530     repeat_p_t dow_p (chset_t("0-1").derived().derived(), 7, 7);
00531
00533     repeat_p_t airport_p (chset_t("0-9A-Z").derived(), 3, 3);
00534
00536     bounded1_2_p_t hours_p (uint1_2_p.derived(), 0u, 23u);
00537
00539     bounded2_p_t minutes_p (uint2_p.derived(), 0u, 59u);
00540
00542     bounded2_p_t seconds_p (uint2_p.derived(), 0u, 59u);
00543
00545     chset_t cabin_code_p ("A-Z");
00546

```

```

00548     chset_t passenger_type_p ("A-Z");
00549
00551     chset_t ff_type_p ("A-Z");
00552
00554     intl_p_t family_code_p;
00555
00557     repeat_p_t class_code_list_p (chset_t("A-Z").derived(), 1, 26);
00558
00560     boundedl_3_p_t stay_duration_p (uintl_3_p.derived(), 0u, 999u);
00561
00562
00563     // //////////////////////////////////////
00564     // (Boost Spirit) Grammar Definition
00565     // //////////////////////////////////////
00566
00567     // //////////////////////////////////////
00568     DemandParser::DemandParser (SEVMGR::SEVMGR_ServicePtr_T ioSEVMGR_ServicePtr,
00569                                 stdair::RandomGeneration& ioSharedGenerator,
00570                                 const POSProbabilityMass_T& iPOSProbMass,
00571                                 DemandStruct& ioDemand)
00572     : _sevmgrServicePtr (ioSEVMGR_ServicePtr),
00573       _uniformGenerator (ioSharedGenerator),
00574       _posProbabilityMass (iPOSProbMass), _demand (ioDemand) {
00575 }
00576
00577     // //////////////////////////////////////
00578     template<typename ScannerT>
00579     DemandParser::definition<ScannerT>::
00580     definition (DemandParser const& self) {
00581
00582         demand_list = *( not_to_be_parsed |
00583                        demand)
00584
00585         ;
00586
00587         not_to_be_parsed = bsc::
00588         lexeme_d[bsc::comment_p("//")
00589                 | bsc::comment_p("/*", "*/")
00590                 | bsc::eol_p]
00591
00592         ;
00593
00594         demand =
00595         pref_dep_date_range
00596         >> ';' >> origin >> ';' >> destination
00597         >> ';' >> pref_cabin[storePrefCabin(self._demand)]
00598         >> ';' >> pos_dist
00599         >> ';' >> channel_dist
00600         >> ';' >> trip_dist
00601         >> ';' >> stay_dist
00602         >> ';' >> ff_dist
00603         >> ';' >> change_fees
00604         >> ';' >> non_refundable
00605         >> ';' >> pref_dep_time_dist
00606         >> ';' >> wtp
00607         >> ';' >> time_value_dist
00608         >> ';' >> dtd_dist
00609         >> ';' >> demand_params
00610         >> demand_end[doEndDemand (self._sevmgrServicePtr,
00611                                   self._uniformGenerator,
00612                                   self._posProbabilityMass, self._demand)]
00613
00614         ;
00615
00616         demand_end = bsc::ch_p(';')
00617
00618         ;
00619
00620         pref_dep_date_range = date[storePrefDepDateRangeStart (self._demand)]
00621         >> ';' >> date[storePrefDepDateRangeEnd (self._demand)]
00622         >> ';' >> dow[storeDow (self._demand)]

```

```

00619         ;
00620
00621     date =
00622         bsc::lexeme_d[ (year_p) [bsc::assign_a(self._demand._itYear)]
00623             >> '-' >> (month_p) [bsc::assign_a(self._demand._itMonth)]
00624             >> '-' >> (day_p) [bsc::assign_a(self._demand._itDay)]
00625     ]
00626     ;
00627
00628     dow = bsc::lexeme_d[ dow_p ]
00629     ;
00630
00631     origin =
00632         (airport_p) [storeOrigin(self._demand)]
00633     ;
00634
00635     destination =
00636         (airport_p) [storeDestination(self._demand)]
00637     ;
00638
00639     pref_cabin = cabin_code_p;
00640
00641     pos_dist =
00642         pos_pair >> * ( ',' >> pos_pair )
00643     ;
00644
00645     pos_pair =
00646         pos_code[storePosCode(self._demand)]
00647         >> ':' >> pos_share
00648     ;
00649
00650     pos_code =
00651         airport_p
00652         | bsc::chseq_p("row")
00653     ;
00654
00655     pos_share =
00656         (bsc::ureal_p) [storePosProbMass(self._demand)]
00657     ;
00658
00659     channel_dist =
00660         channel_pair >> * ( ',' >> channel_pair )
00661     ;
00662
00663     channel_pair =
00664         channel_code[storeChannelCode(self._demand)]
00665         >> ':' >> channel_share
00666     ;
00667
00668     channel_code =
00669         bsc::chseq_p("DF") | bsc::chseq_p("DN")
00670         | bsc::chseq_p("IF") | bsc::chseq_p("IN")
00671     ;
00672
00673     channel_share =
00674         (bsc::ureal_p) [storeChannelProbMass(self._demand)]
00675     ;
00676
00677     trip_dist =
00678         trip_pair >> * ( ',' >> trip_pair )
00679     ;
00680
00681     trip_pair =
00682         trip_code[storeTripCode(self._demand)]
00683         >> ':' >> trip_share
00684     ;
00685

```



```

00686     trip_code =
00687         bsc::chseq_p("R0") | bsc::chseq_p("RI") | bsc::chseq_p("OW")
00688     ;
00689
00690     trip_share =
00691         (bsc::ureal_p) [storeTripProbMass(self._demand)]
00692     ;
00693
00694     stay_dist =
00695         stay_pair >> *( ',' >> stay_pair )
00696     ;
00697
00698     stay_pair =
00699         (stay_duration_p) [storeStayCode(self._demand)]
00700         >> ':' >> stay_share
00701     ;
00702
00703     stay_share =
00704         (bsc::ureal_p) [storeStayProbMass(self._demand)]
00705     ;
00706
00707     ff_dist =
00708         ff_pair >> *( ',' >> ff_pair )
00709     ;
00710
00711     ff_pair =
00712         ff_code [storeFFCode(self._demand)]
00713         >> ':' >> ff_share
00714     ;
00715
00716     ff_code = ff_type_p;
00717
00718     ff_share =
00719         (bsc::ureal_p) [storeFFProbMass(self._demand)]
00720     ;
00721
00722     change_fees =
00723         (bsc::ureal_p) [storeDemandChangeFeeProb(self._demand)]
00724         >> ';' >> (bsc::ureal_p) [storeDemandChangeFeeDisutility(self._demand)]
00725     ;
00726
00727     non_refundable =
00728         (bsc::ureal_p) [storeDemandNonRefundableProb(self._demand)]
00729         >> ';' >> (bsc::ureal_p) [storeDemandNonRefundableDisutility(self._demand)]
00730     ]
00731     ;
00732
00733     pref_dep_time_dist =
00734         pref_dep_time_pair >> *( ',' >> pref_dep_time_pair )
00735     ;
00736
00737     pref_dep_time_pair =
00738         (time) [storePrefDepTime(self._demand)]
00739         >> ':' >> pref_dep_time_share
00740     ;
00741
00742     pref_dep_time_share =
00743         (bsc::ureal_p) [storePrefDepTimeProbMass(self._demand)]
00744     ;
00745
00746     time =
00747         bsc::lexeme_d[
00748             (hours_p) [bsc::assign_a(self._demand._itHours)]
00749             >> !('.' >> (minutes_p) [bsc::assign_a(self._demand._itMinutes)])
00750             >> !('.' >> (seconds_p) [bsc::assign_a(self._demand._itSeconds)])
00751         ]
00752     ;

```

```

00752
00753     wtp =
00754         (bsc::ureal_p) [storeWTP(self._demand)]
00755     ;
00756
00757     time_value_dist =
00758         time_value_pair >> *( ',' >> time_value_pair )
00759     ;
00760
00761     time_value_pair =
00762         (bsc::ureal_p) [storeTimeValue(self._demand)]
00763     >> ':' >> time_value_share
00764     ;
00765
00766     time_value_share =
00767         (bsc::ureal_p) [storeTimeValueProbMass(self._demand)]
00768     ;
00769
00770     dtd_dist =
00771         dtd_pair >> *( ',' >> dtd_pair )
00772     ;
00773
00774     dtd_pair =
00775         (bsc::ureal_p) [storeDTD(self._demand)]
00776     >> ':' >> dtd_share
00777     ;
00778
00779     dtd_share =
00780         (bsc::ureal_p) [storeDTDProbMass(self._demand)]
00781     ;
00782
00783     demand_params =
00784         bsc::ch_p('N')
00785     >> ','
00786     >> (bsc::ureal_p) [storeDemandMean(self._demand)]
00787     >> ','
00788     >> (bsc::ureal_p) [storeDemandStdDev(self._demand)]
00789     ;
00790
00791     // BOOST_SPIRIT_DEBUG_NODE (DemandParser);
00792     BOOST_SPIRIT_DEBUG_NODE (demand_list);
00793     BOOST_SPIRIT_DEBUG_NODE (not_to_be_parsed);
00794     BOOST_SPIRIT_DEBUG_NODE (demand);
00795     BOOST_SPIRIT_DEBUG_NODE (demand_end);
00796     BOOST_SPIRIT_DEBUG_NODE (pref_dep_date);
00797     BOOST_SPIRIT_DEBUG_NODE (date);
00798     BOOST_SPIRIT_DEBUG_NODE (origin);
00799     BOOST_SPIRIT_DEBUG_NODE (destination);
00800     BOOST_SPIRIT_DEBUG_NODE (pref_cabin);
00801     BOOST_SPIRIT_DEBUG_NODE (pos_dist);
00802     BOOST_SPIRIT_DEBUG_NODE (pos_pair);
00803     BOOST_SPIRIT_DEBUG_NODE (pos_code);
00804     BOOST_SPIRIT_DEBUG_NODE (pos_share);
00805     BOOST_SPIRIT_DEBUG_NODE (channel_dist);
00806     BOOST_SPIRIT_DEBUG_NODE (channel_pair);
00807     BOOST_SPIRIT_DEBUG_NODE (channel_code);
00808     BOOST_SPIRIT_DEBUG_NODE (channel_share);
00809     BOOST_SPIRIT_DEBUG_NODE (trip_dist);
00810     BOOST_SPIRIT_DEBUG_NODE (trip_pair);
00811     BOOST_SPIRIT_DEBUG_NODE (trip_code);
00812     BOOST_SPIRIT_DEBUG_NODE (trip_share);
00813     BOOST_SPIRIT_DEBUG_NODE (stay_dist);
00814     BOOST_SPIRIT_DEBUG_NODE (stay_pair);
00815     BOOST_SPIRIT_DEBUG_NODE (stay_share);
00816     BOOST_SPIRIT_DEBUG_NODE (ff_dist);
00817     BOOST_SPIRIT_DEBUG_NODE (ff_pair);
00818     BOOST_SPIRIT_DEBUG_NODE (ff_code);

```

```

00819     BOOST_SPIRIT_DEBUG_NODE (ff_share);
00820     BOOST_SPIRIT_DEBUG_NODE (change_fees);
00821     BOOST_SPIRIT_DEBUG_NODE (non_refundable);
00822     BOOST_SPIRIT_DEBUG_NODE (pref_dep_time_dist);
00823     BOOST_SPIRIT_DEBUG_NODE (pref_dep_time_pair);
00824     BOOST_SPIRIT_DEBUG_NODE (pref_dep_time_share);
00825     BOOST_SPIRIT_DEBUG_NODE (time);
00826     BOOST_SPIRIT_DEBUG_NODE (wtp);
00827     BOOST_SPIRIT_DEBUG_NODE (time_value_dist);
00828     BOOST_SPIRIT_DEBUG_NODE (time_value_pair);
00829     BOOST_SPIRIT_DEBUG_NODE (time_value_share);
00830     BOOST_SPIRIT_DEBUG_NODE (dtd_dist);
00831     BOOST_SPIRIT_DEBUG_NODE (dtd_pair);
00832     BOOST_SPIRIT_DEBUG_NODE (dtd_share);
00833     BOOST_SPIRIT_DEBUG_NODE (demand_params);
00834 }
00835
00836 // //////////////////////////////////////
00837 template<typename ScannerT>
00838 bsc::rule<ScannerT> const&
00839 DemandParser::definition<ScannerT>::start() const {
00840     return demand_list;
00841 }
00842
00843 }
00844
00845 //
00846 // Entry class for the file parser
00847 //
00848 // //////////////////////////////////////
00849 DemandFileParser::
00850 DemandFileParser (SEVMGR::SEVMGR_ServicePtr_T ioSEVMGR_ServicePtr,
00851                  stdair::RandomGeneration& ioSharedGenerator,
00852                  const POSProbabilityMass_T& iPOSProbMass,
00853                  const std::string& iFilename)
00854 : _filename (iFilename),
00855   _sevmgrServicePtr (ioSEVMGR_ServicePtr),
00856   _uniformGenerator (ioSharedGenerator),
00857   _posProbabilityMass (iPOSProbMass) {
00858     init();
00859 }
00860
00861 // //////////////////////////////////////
00862 void DemandFileParser::init() {
00863     // Check that the file exists and is readable
00864     const bool doesExistAndIsReadable =
00865         stdair::BasFileMgr::doesExistAndIsReadable (_filename);
00866
00867     if (doesExistAndIsReadable == false) {
00868         STDAIR_LOG_ERROR ("The demand file " << _filename
00869             << " does not exist or can not be read.");
00870
00871         throw DemandInputFileNotFoundException ("The demand file " + _filename
00872             + " does not exist or can not "
00873             + "be read");
00874     }
00875
00876     // Open the file
00877     _startIterator = iterator_t (_filename);
00878
00879     // Check the filename exists and can be open
00880     if (!_startIterator) {
00881         STDAIR_LOG_ERROR ("The demand file " << _filename << " can not be open.");
00882
00883         throw DemandInputFileNotFoundException ("The demand file " + _filename

```

```

00888                                     + " does not exist or can not "
00889                                     + "be read");
00890     }
00891
00892     // Create an EOF iterator
00893     _endIterator = _startIterator.make_end();
00894 }
00895
00896 // //////////////////////////////////////
00897 bool DemandFileParser::generateDemand () {
00898     bool oResult = false;
00899
00900     STDAIR_LOG_DEBUG ("Parsing demand input file: " << _filename);
00901
00902     // Initialise the parser (grammar) with the helper/staging structure.
00903     DemandParserHelper::DemandParser lDemandParser (_sevmgrServicePtr,
00904                                                     _uniformGenerator,
00905                                                     _posProbabilityMass,
00906                                                     _demand);
00907
00908     // Launch the parsing of the file and, thanks to the doEndDemand
00909     // call-back structure, the building of the whole EventQueue BOM
00910     // (i.e., including Inventory, FlightDate, LegDate, SegmentDate, etc.)
00911     bsc::parse_info<iterator_t> info =
00912         bsc::parse (_startIterator, _endIterator, lDemandParser,
00913                   bsc::space_p - bsc::eol_p);
00914
00915     // Retrieves whether or not the parsing was successful
00916     oResult = info.hit;
00917
00918     const std::string hasBeenFullyReadStr = (info.full == true)?"":"not ";
00919     if (oResult == true) {
00920         STDAIR_LOG_DEBUG ("Parsing of demand input file: " << _filename
00921                           << " succeeded: read " << info.length
00922                           << " characters. The input file has "
00923                           << hasBeenFullyReadStr
00924                           << "been fully read. Stop point: " << info.stop);
00925     } else {
00926         std::ostringstream oStr;
00927         oStr << "Parsing of demand input file: " << _filename << " failed: read "
00928             << info.length << " characters. The input file has "
00929             << hasBeenFullyReadStr << "been fully read. Stop point: "
00930             << info.stop;
00931         STDAIR_LOG_ERROR (oStr.str());
00932         throw stdair::ParserException (oStr.str());
00933     }
00934
00935     return oResult;
00936 }
00937
00938
00939 }

```

25.91 trademgen/command/DemandParserHelper.hpp File Reference

```
#include <string>
#include <stdair/command/CmdAbstract.hpp>
#include <sevmgr/SEVMGR_Types.hpp>
#include <trademgen/TRADEMGEN_Types.hpp>
#include <trademgen/basic/BasParserTypes.hpp>
#include <trademgen/bom/DemandStruct.hpp>
```

Classes

- struct [TRADEMGEN::DemandParserHelper::ParserSemanticAction](#)
- struct [TRADEMGEN::DemandParserHelper::storePrefDepDateRangeStart](#)
- struct [TRADEMGEN::DemandParserHelper::storePrefDepDateRangeEnd](#)
- struct [TRADEMGEN::DemandParserHelper::storeDow](#)
- struct [TRADEMGEN::DemandParserHelper::storeOrigin](#)
- struct [TRADEMGEN::DemandParserHelper::storeDestination](#)
- struct [TRADEMGEN::DemandParserHelper::storePrefCabin](#)
- struct [TRADEMGEN::DemandParserHelper::storeDemandMean](#)
- struct [TRADEMGEN::DemandParserHelper::storeDemandStdDev](#)
- struct [TRADEMGEN::DemandParserHelper::storeDemandChangeFeeProb](#)
- struct [TRADEMGEN::DemandParserHelper::storeDemandChangeFeeDisutility](#)
- struct [TRADEMGEN::DemandParserHelper::storeDemandNonRefundableProb](#)
- struct [TRADEMGEN::DemandParserHelper::storeDemandNonRefundableDisutility](#)
- struct [TRADEMGEN::DemandParserHelper::storePosCode](#)
- struct [TRADEMGEN::DemandParserHelper::storePosProbMass](#)
- struct [TRADEMGEN::DemandParserHelper::storeChannelCode](#)
- struct [TRADEMGEN::DemandParserHelper::storeChannelProbMass](#)
- struct [TRADEMGEN::DemandParserHelper::storeTripCode](#)
- struct [TRADEMGEN::DemandParserHelper::storeTripProbMass](#)
- struct [TRADEMGEN::DemandParserHelper::storeStayCode](#)
- struct [TRADEMGEN::DemandParserHelper::storeStayProbMass](#)
- struct [TRADEMGEN::DemandParserHelper::storeFFCode](#)
- struct [TRADEMGEN::DemandParserHelper::storeFFProbMass](#)
- struct [TRADEMGEN::DemandParserHelper::storePrefDepTime](#)
- struct [TRADEMGEN::DemandParserHelper::storePrefDepTimeProbMass](#)
- struct [TRADEMGEN::DemandParserHelper::storeWTP](#)
- struct [TRADEMGEN::DemandParserHelper::storeTimeValue](#)
- struct [TRADEMGEN::DemandParserHelper::storeTimeValueProbMass](#)
- struct [TRADEMGEN::DemandParserHelper::storeDTD](#)
- struct [TRADEMGEN::DemandParserHelper::storeDTDProbMass](#)
- struct [TRADEMGEN::DemandParserHelper::doEndDemand](#)
- struct [TRADEMGEN::DemandParserHelper::DemandParser](#)
- struct [TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT >](#)
- class [TRADEMGEN::DemandFileParser](#)

Namespaces

- namespace [stdair](#)
Forward declarations.
- namespace [TRADEMGEN](#)
- namespace [TRADEMGEN::DemandParserHelper](#)

25.92 DemandParserHelper.hpp

```

00001 #ifndef __TRADEMGEN_CMD_DEMANDPARSERHELPER_HPP
00002 #define __TRADEMGEN_CMD_DEMANDPARSERHELPER_HPP
00003
00004 // //////////////////////////////////////
00005 // Import section
00006 // //////////////////////////////////////
00007 // STL
00008 #include <string>
00009 // STDAIR
00010 #include <stdair/command/CmdAbstract.hpp>
00011 // SEvMgr
00012 #include <sevmgr/SEVMGR_Types.hpp>
00013 // TRADEMGEN
00014 #include <trademgen/TRADEMGEN_Types.hpp>
00015 #include <trademgen/basic/BasParserTypes.hpp>
00016 #include <trademgen/bom/DemandStruct.hpp>
00017
00018 // Forward declarations
00019 namespace stdair {
00020     struct RandomGeneration;
00021 }
00022
00023 namespace TRADEMGEN {
00024
00025     namespace DemandParserHelper {
00026
00027         // //////////////////////////////////////
00028         // Semantic actions
00029         // //////////////////////////////////////
00031         struct ParserSemanticAction {
00033             ParserSemanticAction (DemandStruct&);
00035             DemandStruct& _demand;
00036         };
00037
00039         struct storePrefDepDateRangeStart : public ParserSemanticAction {
00041             storePrefDepDateRangeStart (DemandStruct&);
00043             void operator() (iterator_t iStr, iterator_t iStrEnd) const;
00044         };
00045
00047         struct storePrefDepDateRangeEnd : public ParserSemanticAction {
00049             storePrefDepDateRangeEnd (DemandStruct&);
00051             void operator() (iterator_t iStr, iterator_t iStrEnd) const;
00052         };
00053
00055         struct storeDow : public ParserSemanticAction {
00057             storeDow (DemandStruct&);
00059             void operator() (iterator_t iStr, iterator_t iStrEnd) const;
00060         };
00061
00063         struct storeOrigin : public ParserSemanticAction {
00065             storeOrigin (DemandStruct&);
00067             void operator() (iterator_t iStr, iterator_t iStrEnd) const;
00068         };
00069
00071         struct storeDestination : public ParserSemanticAction {
00073             storeDestination (DemandStruct&);
00075             void operator() (iterator_t iStr, iterator_t iStrEnd) const;
00076         };
00077
00079         struct storePrefCabin : public ParserSemanticAction {
00081             storePrefCabin (DemandStruct&);
00083             void operator() (iterator_t iStr, iterator_t iStrEnd) const;
00084         };
00085
00087         struct storeDemandMean : public ParserSemanticAction {

```

```
00089     storeDemandMean (DemandStruct&);
00091     void operator() (double iReal) const;
00092 };
00093
00095 struct storeDemandStdDev : public ParserSemanticAction {
00097     storeDemandStdDev (DemandStruct&);
00099     void operator() (double iReal) const;
00100 };
00101
00103 struct storeDemandChangeFeeProb : public ParserSemanticAction {
00105     storeDemandChangeFeeProb (DemandStruct&);
00107     void operator() (double iReal) const;
00108 };
00109
00111 struct storeDemandChangeFeeDisutility : public ParserSemanticAction {
00113     storeDemandChangeFeeDisutility (DemandStruct&);
00115     void operator() (double iReal) const;
00116 };
00117
00119 struct storeDemandNonRefundableProb : public ParserSemanticAction {
00121     storeDemandNonRefundableProb (DemandStruct&);
00123     void operator() (double iReal) const;
00124 };
00125
00127 struct storeDemandNonRefundableDisutility : public ParserSemanticAction {
00129     storeDemandNonRefundableDisutility (DemandStruct&);
00131     void operator() (double iReal) const;
00132 };
00133
00135 struct storePosCode : public ParserSemanticAction {
00137     storePosCode (DemandStruct&);
00139     void operator() (iterator_t iStr, iterator_t iStrEnd) const;
00140 };
00141
00143 struct storePosProbMass : public ParserSemanticAction {
00145     storePosProbMass (DemandStruct&);
00147     void operator() (double iReal) const;
00148 };
00149
00151 struct storeChannelCode : public ParserSemanticAction {
00153     storeChannelCode (DemandStruct&);
00155     void operator() (iterator_t iStr, iterator_t iStrEnd) const;
00156 };
00157
00159 struct storeChannelProbMass : public ParserSemanticAction {
00161     storeChannelProbMass (DemandStruct&);
00163     void operator() (double iReal) const;
00164 };
00165
00167 struct storeTripCode : public ParserSemanticAction {
00169     storeTripCode (DemandStruct&);
00171     void operator() (iterator_t iStr, iterator_t iStrEnd) const;
00172 };
00173
00175 struct storeTripProbMass : public ParserSemanticAction {
00177     storeTripProbMass (DemandStruct&);
00179     void operator() (double iReal) const;
00180 };
00181
00183 struct storeStayCode : public ParserSemanticAction {
00185     storeStayCode (DemandStruct&);
00187     void operator() (unsigned int iInteger) const;
00188 };
00189
00191 struct storeStayProbMass : public ParserSemanticAction {
00193     storeStayProbMass (DemandStruct&);
00195     void operator() (double iReal) const;
```



```

00196     };
00197
00199     struct storeFFCode : public ParserSemanticAction {
00201         storeFFCode (DemandStruct&);
00203         void operator() (iterator_t iStr, iterator_t iStrEnd) const;
00204     };
00205
00207     struct storeFFProbMass : public ParserSemanticAction {
00209         storeFFProbMass (DemandStruct&);
00211         void operator() (double iReal) const;
00212     };
00213
00216     struct storePrefDepTime : public ParserSemanticAction {
00218         storePrefDepTime (DemandStruct&);
00220         void operator() (iterator_t iStr, iterator_t iStrEnd) const;
00221     };
00222
00225     struct storePrefDepTimeProbMass : public ParserSemanticAction {
00227         storePrefDepTimeProbMass (DemandStruct&);
00229         void operator() (double iReal) const;
00230     };
00231
00233     struct storeWTP : public ParserSemanticAction {
00235         storeWTP (DemandStruct&);
00237         void operator() (double iReal) const;
00238     };
00239
00241     struct storeTimeValue : public ParserSemanticAction {
00243         storeTimeValue (DemandStruct&);
00245         void operator() (double iReal) const;
00246     };
00247
00249     struct storeTimeValueProbMass : public ParserSemanticAction {
00251         storeTimeValueProbMass (DemandStruct&);
00253         void operator() (double iReal) const;
00254     };
00255
00258     struct storeDTD : public ParserSemanticAction {
00260         storeDTD (DemandStruct&);
00262         void operator() (unsigned int iInteger) const;
00263     };
00264
00267     struct storeDTDProbMass : public ParserSemanticAction {
00269         storeDTDProbMass (DemandStruct&);
00271         void operator() (double iReal) const;
00272     };
00273
00275     struct doEndDemand : public ParserSemanticAction {
00277         doEndDemand (SEVMGR::SEVMGR_ServicePtr_T, stdair::RandomGeneration&,
00278                     const POSProbabilityMass_T&, DemandStruct&);
00280         void operator() (iterator_t iStr, iterator_t iStrEnd) const;
00282         SEVMGR::SEVMGR_ServicePtr_T _sevmgrServicePtr;
00283         stdair::RandomGeneration& _uniformGenerator;
00284         const POSProbabilityMass_T& _posProbabilityMass;
00285     };
00286
00287
00289     //
00290     // (Boost Spirit) Grammar Definition
00291     //
00293
00384     struct DemandParser :
00385         public boost::spirit::classic::grammar<DemandParser> {
00386
00387         DemandParser (SEVMGR::SEVMGR_ServicePtr_T, stdair::RandomGeneration&,
00388                     const POSProbabilityMass_T&, DemandStruct&);
00389

```

```

00390     template <typename ScannerT>
00391     struct definition {
00392         definition (DemandParser const& self);
00393
00394         // Instantiation of rules
00395         boost::spirit::classic::rule<ScannerT> demand_list,
00396             not_to_be_parsed, demand, demand_end, pref_dep_date_range,
00397             date, dow, origin, destination, pref_cabin, demand_params,
00398             pos_dist, pos_pair, pos_code, pos_share,
00399             channel_dist, channel_pair, channel_code, channel_share,
00400             trip_dist, trip_pair, trip_code, trip_share,
00401             stay_dist, stay_pair, stay_share,
00402             ff_dist, ff_pair, ff_code, ff_share,
00403             change_fees,
00404             non_refundable,
00405             pref_dep_time_dist, pref_dep_time_pair, pref_dep_time_share, time,
00406             wtp,
00407             time_value_dist, time_value_pair, time_value_share,
00408             dtd_dist, dtd_pair, dtd_share;
00409
00410         boost::spirit::classic::rule<ScannerT> const& start() const;
00411     };
00412
00413     // Parser Context
00414     SEVMGR::SEVMGR_ServicePtr_T _sevmgrServicePtr;
00415     stdair::RandomGeneration& _uniformGenerator;
00416     const POSProbabilityMass_T& _posProbabilityMass;
00417     DemandStruct& _demand;
00418 };
00419
00420 }
00421
00422 //
00423 // Entry class for the file parser
00424 //
00425
00426 class DemandFileParser : public stdair::CmdAbstract {
00427 public:
00428     DemandFileParser (SEVMGR::SEVMGR_ServicePtr_T, stdair::RandomGeneration&,
00429         const POSProbabilityMass_T&,
00430         const stdair::Filename_T& iDemandInputFilename);
00431
00432     bool generateDemand ();
00433
00434 private:
00435     void init();
00436
00437 private:
00438     // Attributes
00439     stdair::Filename_T _filename;
00440
00441     iterator_t _startIterator;
00442
00443     iterator_t _endIterator;
00444
00445     SEVMGR::SEVMGR_ServicePtr_T _sevmgrServicePtr;
00446
00447     stdair::RandomGeneration& _uniformGenerator;
00448
00449     const POSProbabilityMass_T& _posProbabilityMass;
00450
00451     DemandStruct _demand;
00452 };
00453
00454 }
00455 #endif // __TRADEMGEN_CMD_DEMANDPARSERHELPER_HPP

```

25.93 trademgen/DBParams.hpp File Reference

```
#include <iosfwd>
#include <string>
#include <trademgen/TRADEMGEN_Types.hpp>
#include <trademgen/TRADEMGEN_Abstract.hpp>
```

Classes

- struct [TRADEMGEN::DBParams](#)

Namespaces

- namespace [TRADEMGEN](#)

Typedefs

- typedef std::list< std::string > [TRADEMGEN::DBParamsNameList_T](#)

25.94 DBParams.hpp

```

00001 #ifndef __TRADEMGEN_DBPARAMS_HPP
00002 #define __TRADEMGEN_DBPARAMS_HPP
00003
00004 // //////////////////////////////////////
00005 // Import section
00006 // //////////////////////////////////////
00007 // STL
00008 #include <iosfwd>
00009 #include <string>
00010 // Trademgen
00011 #include <trademgen/TRADEMGEN_Types.hpp>
00012 #include <trademgen/TRADEMGEN_Abstract.hpp>
00013
00014 namespace TRADEMGEN {
00015
00016     typedef std::list<std::string> DBParamsNameList_T;
00017
00018
00019
00020     struct DBParams : public TRADEMGEN_Abstract {
00021     public:
00022         // ////////// Getters //////////
00023         std::string getUser() const {
00024             return _user;
00025         }
00026
00027         std::string getPassword() const {
00028             return _passwd;
00029         }
00030
00031         std::string getHost() const {
00032             return _host;
00033         }
00034
00035         std::string getPort() const {
00036             return _port;
00037         }
00038
00039         std::string getDBName() const {
00040             return _dbname;
00041         }
00042
00043         // ////////// Setters //////////
00044         void setUser (const std::string& iUser) {
00045             _user = iUser;
00046         }
00047
00048         void setPassword (const std::string& iPasswd) {
00049             _passwd = iPasswd;
00050         }
00051
00052         void setHost (const std::string& iHost) {
00053             _host = iHost;
00054         }
00055
00056         void setPort (const std::string& iPort) {
00057             _port = iPort;
00058         }
00059
00060         void setDBName (const std::string& iDBName) {
00061             _dbname = iDBName;
00062         }
00063
00064     public:

```

```

00078     // /////////// Busines methods ///////////
00080     bool check () const {
00081         if (_user.empty() == true || _passwd.empty() == true
00082             || _host.empty() == true || _port.empty()
00083             || _dbname.empty() == true) {
00084             return false;
00085         }
00086         return true;
00087     }
00088
00089 public:
00090     // /////////// Display methods ///////////
00093     void toStream (std::ostream& ioOut) const {
00094         ioOut << toString();
00095     }
00096
00099     void fromStream (std::istream&) {
00100     }
00101
00103     std::string toShortString() const {
00104         std::ostringstream oStr;
00105         oStr << _dbname << "." << _user << "@" << _host << ":" << _port;
00106         return oStr.str();
00107     }
00108
00110     std::string toString() const {
00111         std::ostringstream oStr;
00112         oStr << _dbname << "." << _user << "@" << _host << ":" << _port;
00113         return oStr.str();
00114     }
00115
00116
00117 public:
00119     DBParams (const std::string& iDBUser, const std::string& iDBPasswd,
00120              const std::string& iDBHost, const std::string& iDBPort,
00121              const std::string& iDBName)
00122         : _user (iDBUser), _passwd (iDBPasswd), _host (iDBHost), _port (iDBPort),
00123           _dbname (iDBName) {
00124     }
00125
00127     // DBParams ();
00129     // DBParams (const DBParams&);
00130
00132     virtual ~DBParams() {}
00133
00134
00135 private:
00136     // /////////// Attributes ///////////
00138     std::string _user;
00140     std::string _passwd;
00142     std::string _host;
00144     std::string _port;
00146     std::string _dbname;
00147 };
00148
00149 }
00150 #endif // __TRADEMGEN_DBPARAMS_HPP

```

25.95 trademgen/factory/FacTRADEMGENSEerviceContext.cpp File Reference

```
#include <cassert>
#include <stdair/service/FacSupervisor.hpp>
#include <trademgen/factory/FacTRADEMGENSEerviceContext.hpp>
#include <trademgen/service/TRADEMGEN_ServiceContext.hpp>
```

Namespaces

- namespace [TRADEMGEN](#)

25.96 FacTRADEMGENServiceContext.cpp

```

00001 // //////////////////////////////////////
00002 // Import section
00003 // //////////////////////////////////////
00004 // STL
00005 #include <cassert>
00006 // StdAir
00007 #include <stdair/service/FacSupervisor.hpp>
00008 // TraDemGen
00009 #include <trademgen/factory/FacTRADEMGENServiceContext.hpp>
00010 #include <trademgen/service/TRADEMGEN_ServiceContext.hpp>
00011
00012 namespace TRADEMGEN {
00013
00014     FacTRADEMGENServiceContext* FacTRADEMGENServiceContext::_instance = NULL;
00015
00016     // //////////////////////////////////////
00017     FacTRADEMGENServiceContext::~FacTRADEMGENServiceContext() {
00018         _instance = NULL;
00019     }
00020
00021     // //////////////////////////////////////
00022     FacTRADEMGENServiceContext& FacTRADEMGENServiceContext::instance() {
00023
00024         if (_instance == NULL) {
00025             _instance = new FacTRADEMGENServiceContext();
00026             assert (_instance != NULL);
00027
00028             stdair::FacSupervisor::instance().registerServiceFactory (_instance);
00029         }
00030         return *_instance;
00031     }
00032
00033     // //////////////////////////////////////
00034     TRADEMGEN_ServiceContext& FacTRADEMGENServiceContext::
00035     create (const stdair::RandomSeed_T& iRandomSeed) {
00036         TRADEMGEN_ServiceContext* aServiceContext_ptr = NULL;
00037
00038         aServiceContext_ptr = new TRADEMGEN_ServiceContext (iRandomSeed);
00039         assert (aServiceContext_ptr != NULL);
00040
00041         // The new object is added to the Bom pool
00042         _pool.push_back (aServiceContext_ptr);
00043
00044         return *aServiceContext_ptr;
00045     }
00046
00047 }

```

25.97 trademgen/factory/FacTRADEMGENSEerviceContext.hpp File Reference

```
#include <stdair/stdair_maths_types.hpp>
#include <stdair/service/FacServiceAbstract.hpp>
#include <trademgen/TRADEMGEN_Types.hpp>
```

Classes

- class [TRADEMGEN::FacTRADEMGENSEerviceContext](#)
Factory for creating the TraDemGen service context instance.

Namespaces

- namespace [TRADEMGEN](#)

25.98 FacTRADEMGENServiceContext.hpp

```
00001 #ifndef __TRADEMGEN_FAC_FACTRADEMGENSEVICECONTEXT_HPP
00002 #define __TRADEMGEN_FAC_FACTRADEMGENSEVICECONTEXT_HPP
00003
00004 // //////////////////////////////////////
00005 // Import section
00006 // //////////////////////////////////////
00007 // StdAir
00008 #include <stdair/stdair_maths_types.hpp>
00009 #include <stdair/service/FacServiceAbstract.hpp>
00010 // TraDemGen
00011 #include <trademgen/TRADEMGEN_Types.hpp>
00012
00013 namespace TRADEMGEN {
00014
00016     class TRADEMGEN_ServiceContext;
00017
00021     class FacTRADEMGENServiceContext : public stdair::FacServiceAbstract {
00022     public:
00030         static FacTRADEMGENServiceContext& instance();
00031
00038         ~FacTRADEMGENServiceContext();
00039
00048         TRADEMGEN_ServiceContext& create (const stdair::RandomSeed_T&);
00049
00050
00051     protected:
00057         FacTRADEMGENServiceContext () {}
00058
00059     private:
00063         static FacTRADEMGENServiceContext* _instance;
00064     };
00065
00066 }
00067 #endif // __TRADEMGEN_FAC_FACTRADEMGENSEVICECONTEXT_HPP
```

25.99 trademgen/python/pytrademgen.cpp File Reference

```
#include <cassert>
#include <stdexcept>
#include <fstream>
#include <sstream>
#include <string>
#include <list>
#include <vector>
#include <boost/python.hpp>
#include <boost/accumulators/accumulators.hpp>
#include <boost/accumulators/statistics.hpp>
#include <stdair/stdair_basic_types.hpp>
#include <stdair/basic/BasConst_General.hpp>
#include <stdair/basic/ProgressStatusSet.hpp>
#include <stdair/basic/DemandGenerationMethod.hpp>
#include <stdair/bom/EventStruct.hpp>
#include <stdair/bom/BookingRequestStruct.hpp>
#include <stdair/bom/BomDisplay.hpp>
#include <stdair/service/Logger.hpp>
#include <trademgen/TRADEMGEN_Service.hpp>
#include <trademgen/config/trademgen-paths.hpp>
```

Classes

- struct [TRADEMGEN::Trademgener](#)
Wrapper structure around the C++ API, so as to expose a Python API.

Namespaces

- namespace [TRADEMGEN](#)

Typedefs

- typedef unsigned int [NbOfRuns_T](#)
- typedef `ba::accumulator_set< double, ba::stats< ba::tag::min, ba::tag::max, ba::tag::mean(ba::immediate), ba::tag::sum, ba::tag::variance > > stat_acc_type`

Functions

- void [TRADEMGEN::stat_display](#) (std::ostream &oStream, const [stat_acc_type](#) &iStatAcc)

- [BOOST_PYTHON_MODULE](#) (libpytrademgen)

25.99.1 Typedef Documentation

25.99.1.1 typedef unsigned int NbOfRuns_T

Definition at line 31 of file [pytrademgen.cpp](#).

25.99.1.2 typedef ba::accumulator_set<double, ba::stats<ba::tag::min, ba::tag::max, ba::tag::mean (ba::immediate), ba::tag::sum, ba::tag::variance> > stat_acc_type

Type definition to gather statistics.

Definition at line 40 of file [pytrademgen.cpp](#).

25.99.2 Function Documentation

25.99.2.1 BOOST_PYTHON_MODULE (libpytrademgen)

Definition at line 355 of file [pytrademgen.cpp](#).

References [TRADEMGEN::Trademgener::init\(\)](#), and [TRADEMGEN::Trademgener::trademgen\(\)](#).

25.100 pytrademgen.cpp

```

00001 // STL
00002 #include <cassert>
00003 #include <stdexcept>
00004 #include <fstream>
00005 #include <sstream>
00006 #include <string>
00007 #include <list>
00008 #include <vector>
00009 // Boost Python
00010 #include <boost/python.hpp>
00011 // Boost Accumulators
00012 #include <boost/accumulators/accumulators.hpp>
00013 #include <boost/accumulators/statistics.hpp>
00014 // StdAir
00015 #include <stdair/stdair_basic_types.hpp>
00016 #include <stdair/basic/BasConst_General.hpp>
00017 #include <stdair/basic/ProgressStatusSet.hpp>
00018 #include <stdair/basic/DemandGenerationMethod.hpp>
00019 #include <stdair/bom/EventStruct.hpp>
00020 #include <stdair/bom/BookingRequestStruct.hpp>
00021 #include <stdair/bom/BomDisplay.hpp>
00022 #include <stdair/service/Logger.hpp>
00023 // TraDemGen
00024 #include <trademgen/TRADEMGEN_Service.hpp>
00025 #include <trademgen/config/trademgen-paths.hpp>
00026
00027 // Aliases for namespaces
00028 namespace ba = boost::accumulators;
00029
00030 // ////////// Specific type definitions //////////
00031 typedef unsigned int NbOfRuns_T;
00032
00033 typedef ba::accumulator_set<double,
00034                             ba::stats<ba::tag::min, ba::tag::max,
00035                                     ba::tag::mean (ba::immediate),
00036                                     ba::tag::sum,
00037                                     ba::tag::variance> > stat_acc_type;
00038
00039 namespace TRADEMGEN {
00040
00041     void stat_display (std::ostream& oStream, const stat_acc_type& iStatAcc) {
00042
00043         // Store current formatting flags of the output stream
00044         std::ios::fmtflags oldFlags = oStream.flags();
00045
00046         //
00047         oStream.setf (std::ios::fixed);
00048
00049         //
00050         oStream << "Statistics for the demand generation runs: " << std::endl;
00051         oStream << " minimum   = " << ba::min (iStatAcc) << std::endl;
00052         oStream << " mean      = " << ba::mean (iStatAcc) << std::endl;
00053         oStream << " maximum  = " << ba::max (iStatAcc) << std::endl;
00054         oStream << " count    = " << ba::count (iStatAcc) << std::endl;
00055         oStream << " variance = " << ba::variance (iStatAcc) << std::endl;
00056
00057         // Reset formatting flags of output stream
00058         oStream.flags (oldFlags);
00059     }
00060
00061     struct Trademgener {
00062     public:
00063         std::string
00064         trademgen (const NbOfRuns_T& iNbOfRuns,
00065                  const std::string& iDemandGenerationMethodString) {

```

```

00078     std::ostringstream oStream;
00079
00080     // Convert the input string into a demand generation method enumeration
00081     const stdair::DemandGenerationMethod
00082         iDemandGenerationMethod (iDemandGenerationMethodString);
00083
00084     // Sanity check
00085     if (_logOutputStream == NULL) {
00086         oStream << "The log filepath is not valid." << std::endl;
00087         return oStream.str();
00088     }
00089     assert (_logOutputStream != NULL);
00090
00091     try {
00092
00093         // DEBUG
00094         *_logOutputStream << "Demand generation for " << iNbOfRuns << " runs, "
00095             << "with the following method: "
00096             << iDemandGenerationMethod << std::endl;
00097
00098         if (_trademngenService == NULL) {
00099             oStream << "The TraDemGen service has not been initialised, "
00100                 << "i.e., the init() method has not been called "
00101                 << "correctly on the Trademngen object. Please "
00102                 << "check that all the parameters are not empty and "
00103                 << "point to actual files.";
00104             *_logOutputStream << oStream.str();
00105             return oStream.str();
00106         }
00107         assert (_trademngenService != NULL);
00108
00109         // Initialise the statistics collector/accumulator
00110         stat_acc_type lStatAccumulator;
00111
00112         // Retrieve the expected (mean value of the) number of events to be
00113         // generated
00114         const stdair::Count_T& lExpectedNbOfEventsToBeGenerated =
00115             _trademngenService->getExpectedTotalNumberOfRequestsToBeGenerated();
00116
00117         // Initialise the (Boost) progress display object
00118         boost::progress_display
00119             lProgressDisplay (lExpectedNbOfEventsToBeGenerated * iNbOfRuns);
00120
00121         for (NbOfRuns_T runIdx = 1; runIdx <= iNbOfRuns; ++runIdx) {
00122             // //////////////////////////////////////
00123             *_logOutputStream << "Run number: " << runIdx << std::endl;
00124
00125             const stdair::Count_T& lActualNbOfEventsToBeGenerated =
00126                 _trademngenService->generateFirstRequests (iDemandGenerationMethod);
00127
00128             // DEBUG
00129             *_logOutputStream << "[" << runIdx << "] Expected: "
00130                 << lExpectedNbOfEventsToBeGenerated << ", actual: "
00131                 << lActualNbOfEventsToBeGenerated << std::endl;
00132
00133             while (_trademngenService->isQueueDone() == false) {
00134
00135                 // Extract the next event from the event queue
00136                 stdair::EventStruct lEventStruct;
00137                 stdair::ProgressStatusSet lProgressStatusSet =
00138                     _trademngenService->popEvent (lEventStruct);
00139
00140                 // DEBUG
00141                 // STDAIR_LOG_DEBUG ("[" << runIdx << "] Poped event: '"
00142                     << lEventStruct.describe() << "'.");
00143
00144                 // Extract the corresponding demand/booking request

```

```

00156         const stdair::BookingRequestStruct& lPoppedRequest =
00157             lEventStruct.getBookingRequest();
00158
00159         // DEBUG
00160         *_logOutputStream << "[" << runIdx << "]" Poped booking request: '"
00161             << lPoppedRequest.describe() << "'.'" << std::endl;
00162
00163         // Dump the request into the dedicated CSV file
00164         // stdair::BomDisplay::csvDisplay (output, lPoppedRequest);
00165
00166         // Retrieve the corresponding demand stream key
00167         const stdair::DemandGeneratorKey_T& lDemandStreamKey =
00168             lPoppedRequest.getDemandGeneratorKey();
00169
00170         // Assess whether more events should be generated for that
00171         // demand stream
00172         const bool stillHavingRequestsToBeGenerated = _trademgenService->
00173             stillHavingRequestsToBeGenerated (lDemandStreamKey,
00174                 lProgressStatusSet,
00175                 iDemandGenerationMethod);
00176
00177         // DEBUG
00178         *_logOutputStream << lProgressStatusSet.describe() << std::endl;
00179         *_logOutputStream << "=> [" << lDemandStreamKey
00180             << "]" is now processed. Still generate events "
00181             << "for that demand stream? "
00182             << stillHavingRequestsToBeGenerated << std::endl;
00183
00184         // If there are still events to be generated for that demand
00185         // stream, generate and add them to the event queue
00186         if (stillHavingRequestsToBeGenerated == true) {
00187
00188             stdair::BookingRequestPtr_T lNextRequest_ptr =
00189                 _trademgenService->generateNextRequest (lDemandStreamKey,
00190                     iDemandGenerationMethod);
00191
00192             assert (lNextRequest_ptr != NULL);
00193
00194             // Sanity check
00195             const stdair::Duration_T lDuration =
00196                 lNextRequest_ptr->getRequestDateTime()
00197                 - lPoppedRequest.getRequestDateTime();
00198             if (lDuration.total_milliseconds() < 0) {
00199                 *_logOutputStream << "[" << lDemandStreamKey
00200                     << "]" The date-time of the generated event ("
00201                     << lNextRequest_ptr->getRequestDateTime()
00202                     << ") is lower than the date-time "
00203                     << "of the current event ("
00204                     << lPoppedRequest.getRequestDateTime()
00205                     << ") " << std::endl;
00206                 assert (false);
00207             }
00208
00209             // DEBUG
00210             *_logOutputStream << "[" << lDemandStreamKey
00211                 << "]" Added request: '"
00212                 << lNextRequest_ptr->describe()
00213                 << "'.'" Is queue done? "
00214                 << _trademgenService->isQueueDone()
00215                 << std::endl;
00216         }
00217         // DEBUG
00218         *_logOutputStream << std::endl;
00219
00220         // Update the progress display
00221         ++lProgressDisplay;
00222     }

```

```

00223
00224         // Add the number of events to the statistics accumulator
00225         lStatAccumulator (lActualNbOfEventsToBeGenerated);
00226
00227         // Reset the service (including the event queue) for the next run
00228         _trademgenService->reset();
00229     }
00230
00231     // DEBUG
00232     *_logOutputStream << "End of the demand generation. Following are some "
00233         << "statistics for the " << iNbOfRuns << " runs."
00234         << std::endl;
00235     std::ostringstream oStatStr;
00236     stat_display (oStatStr, lStatAccumulator);
00237     *_logOutputStream << oStatStr.str() << std::endl;
00238
00239     // DEBUG
00240     const std::string& lBOMStr = _trademgenService->csvDisplay();
00241     *_logOutputStream << lBOMStr << std::endl;
00242
00243     // DEBUG
00244     *_logOutputStream << "TraDemGen output: "
00245         << oStream.str() << std::endl;
00246
00247     } catch (const stdair::RootException& eTrademgenError) {
00248         oStream << "TraDemGen error: " << eTrademgenError.what() << std::endl;
00249     }
00250     } catch (const std::exception& eStdError) {
00251         oStream << "Error: " << eStdError.what() << std::endl;
00252     }
00253     } catch (...) {
00254         oStream << "Unknown error" << std::endl;
00255     }
00256
00257     //
00258     oStream << "TraDemGen has completed the generation of the booking "
00259         << "requests. See the log file for more details." << std::endl;
00260
00261     return oStream.str();
00262 }
00263
00264 public:
00266 Trademgener() : _trademgenService (NULL), _logOutputStream (NULL) {
00267 }
00268
00270 Trademgener (const Trademgener& iTrademgener)
00271 : _trademgenService (iTrademgener._trademgenService),
00272   _logOutputStream (iTrademgener._logOutputStream) {
00273 }
00274
00276 ~Trademgener() {
00277     _trademgenService = NULL;
00278     _logOutputStream = NULL;
00279 }
00280
00284 bool init (const std::string& iLogFilepath,
00285           const stdair::RandomSeed_T& iRandomSeed, const bool isBuiltin,
00286           const stdair::Filename_T& iDemandInputFilename) {
00287     bool isEverythingOK = true;
00288
00289     try {
00290
00291         // Check that the file path given as input corresponds to an actual file
00292         const bool isWriteable = (iLogFilepath.empty() == false);
00293         // stdair::BasFileMgr::isWriteable (iLogFilepath);
00294         if (isWriteable == false) {
00295             isEverythingOK = false;

```

```

00296         return isEverythingOK;
00297     }
00298
00299     // Set the log parameters
00300     _logOutputStream = new std::ofstream;
00301     assert (_logOutputStream != NULL);
00302
00303     // Open and clean the log outputfile
00304     _logOutputStream->open (iLogFilePath.c_str());
00305     _logOutputStream->clear();
00306
00307     // DEBUG
00308     *_logOutputStream << "Python wrapper initialisation" << std::endl;
00309     const stdair::BasLogParams lLogParams (stdair::LOG::DEBUG,
00310                                           *_logOutputStream);
00311
00312     // Initialise the context
00313     _trademngenService = new TRADEMGEN_Service (lLogParams, iRandomSeed);
00314     assert (_trademngenService != NULL);
00315
00316     // Check whether or not a (CSV) input file should be read
00317     if (isBuiltin == true) {
00318         // Create a sample DemandStream object, and insert it within
00319         // the BOM tree
00320         _trademngenService->buildSampleBom();
00321
00322     } else {
00323         // Create the DemandStream objects, and insert them within
00324         // the BOM tree
00325         const DemandFilePath lDemandFilePath (iDemandInputFilename);
00326         _trademngenService->parseAndLoad (lDemandFilePath);
00327     }
00328
00329     // DEBUG
00330     *_logOutputStream << "Python wrapper initialised" << std::endl;
00331
00332     } catch (const stdair::RootException& eTrademngenError) {
00333         *_logOutputStream << "Trademngen error: " << eTrademngenError.what()
00334             << std::endl;
00335
00336     } catch (const std::exception& eStdError) {
00337         *_logOutputStream << "Error: " << eStdError.what() << std::endl;
00338
00339     } catch (...) {
00340         *_logOutputStream << "Unknown error" << std::endl;
00341     }
00342
00343     return isEverythingOK;
00344 }
00345
00346 private:
00347     TRADEMGEN_Service* _trademngenService;
00348     std::ofstream* _logOutputStream;
00349 };
00350
00351
00352 }
00353
00354 // //////////////////////////////////////
00355 BOOST_PYTHON_MODULE(libpytrademngen) {
00356     boost::python::class_<TRADEMGEN::Trademgener> ("Trademgener")
00357         .def ("trademngen", &TRADEMGEN::Trademgener::trademngen)
00358         .def ("init", &TRADEMGEN::Trademgener::init);
00359 }

```


25.101 trademgen/service/TRADEMGEN_Service.cpp File Reference

```

#include <cassert>
#include <sstream>
#include <boost/make_shared.hpp>
#include <soci/soci.h>
#include <stdair/basic/BasChronometer.hpp>
#include <stdair/basic/BasConst_General.hpp>
#include <stdair/basic/ProgressStatusSet.hpp>
#include <stdair/bom/BomRoot.hpp>
#include <stdair/bom/BookingRequestStruct.hpp>
#include <stdair/bom/AirlineStruct.hpp>
#include <stdair/bom/EventStruct.hpp>
#include <stdair/command/DBManagerForAirlines.hpp>
#include <stdair/service/Logger.hpp>
#include <stdair/service/DBSessionManager.hpp>
#include <stdair/STDAIR_Service.hpp>
#include <stdair/factory/FacBomManager.hpp>
#include <sevmgr/SEVMGR_Service.hpp>
#include <trademgen/basic/BasConst_TRADEMGEN_Service.hpp>
#include <trademgen/bom/BomDisplay.hpp>
#include <trademgen/bom/DemandStream.hpp>
#include <trademgen/bom/DemandStreamTypes.hpp>
#include <trademgen/factory/FacTRADEMGENServiceContext.hpp>
#include <trademgen/command/DemandParser.hpp>
#include <trademgen/command/DemandManager.hpp>
#include <trademgen/service/TRADEMGEN_ServiceContext.hpp>
#include <trademgen/TRADEMGEN_Service.hpp>

```

Namespaces

- namespace [TRADEMGEN](#)
- namespace [SEVMGR](#)

Functions

- template void [SEVMGR::SEVMGR_Service::addEventGenerator<TRADEMGEN::DemandStream>](#) ([TRADEMGEN::DemandStream](#) &) const
- template [TRADEMGEN::DemandStream](#) & [SEVMGR::SEVMGR_Service::getEventGenerator<TRADEMGEN::DemandStream, stdair::DemandStreamKeyStr_T>](#) (> (const [stdair::DemandStreamKeyStr_T](#) &) const

- template bool SEVMGR::SEVMGR_Service::hasEventGenerator< TRADEMGEN::DemandStream, stdair::DemandStreamKeyStr_T > (const stdair::DemandStreamKeyStr_T &) const
- template const TRADEMGEN::DemandStreamList_T SEVMGR::SEVMGR_Service::getEventGeneratorList< TRADEMGEN::DemandStream > () const
- template bool SEVMGR::SEVMGR_Service::hasEventGeneratorList< TRADEMGEN::DemandStream > () const

25.102 TRADEMGEN_Service.cpp

```

00001 // //////////////////////////////////////
00002 // Import section
00003 // //////////////////////////////////////
00004 // STL
00005 #include <cassert>
00006 #include <sstream>
00007 // Boost
00008 #include <boost/make_shared.hpp>
00009 #if defined(SOCI_HEADERS_BURIED)
00010 #include <soci/core/soci.h>
00011 #else // SOCI_HEADERS_BURIED
00012 #include <soci/soci.h>
00013 #endif // SOCI_HEADERS_BURIED
00014 // StdAir
00015 #include <stdair/basic/BasChronometer.hpp>
00016 #include <stdair/basic/BasConst_General.hpp>
00017 #include <stdair/basic/ProgressStatusSet.hpp>
00018 #include <stdair/bom/BomRoot.hpp>
00019 #include <stdair/bom/BookingRequestStruct.hpp>
00020 #include <stdair/bom/AirlineStruct.hpp>
00021 #include <stdair/bom/EventStruct.hpp>
00022 #include <stdair/command/DBManagerForAirlines.hpp>
00023 #include <stdair/service/Logger.hpp>
00024 #include <stdair/service/DBSessionManager.hpp>
00025 #include <stdair/STDAIR_Service.hpp>
00026 #include <stdair/factory/FacBomManager.hpp>
00027 // SEvMgr
00028 #include <sevmgr/SEVMGR_Service.hpp>
00029 // TraDemGen
00030 #include <trademgen/basic/BasConst_TRADEMGEN_Service.hpp>
00031 #include <trademgen/bom/BomDisplay.hpp>
00032 #include <trademgen/bom/DemandStream.hpp>
00033 #include <trademgen/bom/DemandStreamTypes.hpp>
00034 #include <trademgen/factory/FacTRADEMGENServiceContext.hpp>
00035 #include <trademgen/command/DemandParser.hpp>
00036 #include <trademgen/command/DemandManager.hpp>
00037 #include <trademgen/service/TRADEMGEN_ServiceContext.hpp>
00038 #include <trademgen/TRADEMGEN_Service.hpp>
00039
00040 namespace TRADEMGEN {
00041
00042     // //////////////////////////////////////
00043     TRADEMGEN_Service::TRADEMGEN_Service() : _trademgenServiceContext (NULL) {
00044         assert (false);
00045     }
00046
00047     // //////////////////////////////////////
00048     TRADEMGEN_Service::TRADEMGEN_Service (const TRADEMGEN_Service& iService)
00049         : _trademgenServiceContext (NULL) {
00050         assert (false);
00051     }
00052
00053     // //////////////////////////////////////
00054     TRADEMGEN_Service::TRADEMGEN_Service (const stdair::BasLogParams& iLogParams,
00055                                             const stdair::RandomSeed_T& iRandomSeed)
00056         : _trademgenServiceContext (NULL) {
00057
00058         // Initialise the STDAIR service handler
00059         stdair::STDAIR_ServicePtr_T lSTDAIR_Service_ptr =
00060             initStdAirService (iLogParams);
00061
00062         // Initialise the service context
00063         initServiceContext (iRandomSeed);
00064
00065         // Add the StdAir service context to the TRADEMGEN service context

```

```

00066     // \note TRADEMGEN owns the STDAIR service resources here.
00067     const bool ownStdairService = true;
00068     addStdAirService (lSTDAIR_Service_ptr, ownStdairService);
00069
00070     // Initailise the SEvMgr service.
00071     initSEVMGRService();
00072
00073     // Initialise the (remaining of the) context
00074     initTrademgenService();
00075 }
00076
00077 // //////////////////////////////////////
00078 TRADEMGEN_Service::TRADEMGEN_Service (const stdair::BasLogParams& iLogParams,
00079                                         const stdair::BasDBParams& iDBParams,
00080                                         const stdair::RandomSeed_T& iRandomSeed)
00081 : _trademgenServiceContext (NULL) {
00082
00083     // Initialise the STDAIR service handler
00084     stdair::STDAIR_ServicePtr_T lSTDAIR_Service_ptr =
00085         initStdAirService (iLogParams, iDBParams);
00086
00087     // Initialise the service context
00088     initServiceContext (iRandomSeed);
00089
00090     // Add the StdAir service context to the TRADEMGEN service context
00091     // \note TRADEMGEN owns the STDAIR service resources here.
00092     const bool ownStdairService = true;
00093     addStdAirService (lSTDAIR_Service_ptr, ownStdairService);
00094
00095     // Initailise the SEvMgr service.
00096     initSEVMGRService();
00097
00098     // Initialise the (remaining of the) context
00099     initTrademgenService();
00100 }
00101
00102 // //////////////////////////////////////
00103 TRADEMGEN_Service::
00104 TRADEMGEN_Service (stdair::STDAIR_ServicePtr_T ioSTDAIR_Service_ptr,
00105                     SEVMGR::SEVMGR_ServicePtr_T ioSEVMGR_Service_ptr,
00106                     const stdair::RandomSeed_T& iRandomSeed)
00107 : _trademgenServiceContext (NULL) {
00108
00109     // Initialise the service context
00110     initServiceContext (iRandomSeed);
00111
00112     // Add the StdAir service context to the TRADEMGEN service context
00113     // \note TraDemGen does not own the STDAIR service resources here.
00114     const bool doesNotOwnStdairService = false;
00115     addStdAirService (ioSTDAIR_Service_ptr, doesNotOwnStdairService);
00116
00117     //Add the SEvMgr service to the TRADEMGEN service context.
00118     addSEVMGRService (ioSEVMGR_Service_ptr);
00119
00120     // Initialise the context
00121     initTrademgenService();
00122 }
00123
00124 // //////////////////////////////////////
00125 TRADEMGEN_Service::~TRADEMGEN_Service() {
00126     // Delete/Clean all the objects from memory
00127     finalise();
00128 }
00129
00130 // //////////////////////////////////////
00131 void TRADEMGEN_Service::finalise() {
00132     assert (_trademgenServiceContext != NULL);

```

```

00133     // Reset the (Boost.)Smart pointer pointing on the STDAIR_Service object.
00134     _trademgenServiceContext->reset();
00135 }
00136
00137 // //////////////////////////////////////
00138 void TRADEMGEN_Service::
00139 initServiceContext (const stdair::RandomSeed_T& iRandomSeed) {
00140     // Initialise the service context
00141     TRADEMGEN_ServiceContext& lTRADEMGEN_ServiceContext =
00142         FacTRADEMGENServiceContext::instance().create (iRandomSeed);
00143     _trademgenServiceContext = &lTRADEMGEN_ServiceContext;
00144 }
00145
00146 // //////////////////////////////////////
00147 void TRADEMGEN_Service::
00148 addStdAirService (stdair::STDAIR_ServicePtr_T ioSTDAIR_Service_ptr,
00149                 const bool iOwnStdairService) {
00150     // Retrieve the TraDemGen service context
00151     assert (_trademgenServiceContext != NULL);
00152     TRADEMGEN_ServiceContext& lTRADEMGEN_ServiceContext =
00153         *_trademgenServiceContext;
00154
00155     // Store the STDAIR service object within the (TRADEMGEN) service context
00156     lTRADEMGEN_ServiceContext.setSTDAIR_Service (ioSTDAIR_Service_ptr,
00157                                                iOwnStdairService);
00158 }
00159
00160 // //////////////////////////////////////
00161 void TRADEMGEN_Service::
00162 addSEVMGRService (SEVMGR::SEVMGR_ServicePtr_T ioSEVMGR_Service_ptr) {
00163
00164     // Retrieve the TraDemGen service context
00165     assert (_trademgenServiceContext != NULL);
00166     TRADEMGEN_ServiceContext& lTRADEMGEN_ServiceContext =
00167         *_trademgenServiceContext;
00168
00169     // Store the STDAIR service object within the (TRADEMGEN) service context
00170     lTRADEMGEN_ServiceContext.setSEVMGR_Service (ioSEVMGR_Service_ptr);
00171 }
00172
00173 // //////////////////////////////////////
00174 stdair::STDAIR_ServicePtr_T TRADEMGEN_Service::
00175 initStdAirService (const stdair::BasLogParams& iLogParams,
00176                  const stdair::BasDBParams& iDBParams) {
00177
00178     stdair::STDAIR_ServicePtr_T lSTDAIR_Service_ptr =
00179         boost::make_shared<stdair::STDAIR_Service> (iLogParams, iDBParams);
00180     assert (lSTDAIR_Service_ptr != NULL);
00181
00182     return lSTDAIR_Service_ptr;
00183 }
00184
00185 // //////////////////////////////////////
00186 stdair::STDAIR_ServicePtr_T TRADEMGEN_Service::
00187 initStdAirService (const stdair::BasLogParams& iLogParams) {
00188
00189     stdair::STDAIR_ServicePtr_T lSTDAIR_Service_ptr =
00190         boost::make_shared<stdair::STDAIR_Service> (iLogParams);
00191     assert (lSTDAIR_Service_ptr != NULL);
00192
00193     return lSTDAIR_Service_ptr;
00194 }
00195
00196 // //////////////////////////////////////
00197 void TRADEMGEN_Service::initSEVMGRService() {
00198
00199     // Retrieve the TraDemGen service context

```

```

00210     assert (_trademgenServiceContext != NULL);
00211     TRADEMGEN_ServiceContext& lTRADEMGEN_ServiceContext =
00212         *_trademgenServiceContext;
00213
00214     // Retrieve the StdAir service context
00215     stdair::STDAIR_ServicePtr_T lSTDAIR_Service_ptr =
00216         lTRADEMGEN_ServiceContext.getSTDAIR_ServicePtr();
00217
00225     SEVMGR::SEVMGR_ServicePtr_T lSEVMGR_Service_ptr =
00226         boost::make_shared<SEVMGR::SEVMGR_Service> (lSTDAIR_Service_ptr);
00227
00228     // Store the SEvMgr service object within the (TraDemGen) service context
00229     lTRADEMGEN_ServiceContext.setSEVMGR_Service (lSEVMGR_Service_ptr);
00230 }
00231
00232
00233 // //////////////////////////////////////
00234 void TRADEMGEN_Service::initTrademgenService() {
00235     // Do nothing at this stage. A sample BOM tree may be built by
00236     // calling the buildSampleBom() method
00237 }
00238
00239 // //////////////////////////////////////
00240 void TRADEMGEN_Service::
00241 parseAndLoad (const DemandFilePath& iDemandFilePath) {
00242
00243     // Retrieve the TraDemGen service context
00244     if (_trademgenServiceContext == NULL) {
00245         throw stdair::NonInitialisedServiceException ("The TraDemGen service has "
00246             "not been initialised");
00247     }
00248     assert (_trademgenServiceContext != NULL);
00249
00250     // Retrieve the TraDemGen service context and whether it owns the Stdair
00251     // service
00252     TRADEMGEN_ServiceContext& lTRADEMGEN_ServiceContext =
00253         *_trademgenServiceContext;
00254     const bool doesOwnStdairService =
00255         lTRADEMGEN_ServiceContext.getOwnStdairServiceFlag();
00256
00257     // Retrieve the StdAir service object from the (TRADEMGEN) service context
00258     stdair::STDAIR_Service& lSTDAIR_Service =
00259         lTRADEMGEN_ServiceContext.getSTDAIR_Service();
00260
00261     // Retrieve the persistent BOM root object.
00262     stdair::BomRoot& lPersistentBomRoot =
00263         lSTDAIR_Service.getPersistentBomRoot();
00264
00265     // Retrieve the pointer on the SEvMgr service handler.
00266     SEVMGR::SEVMGR_ServicePtr_T lSEVMGR_Service_ptr =
00267         lTRADEMGEN_ServiceContext.getSEVMGR_ServicePtr();
00268
00269     // Retrieve the shared generator
00270     stdair::RandomGeneration& lSharedGenerator =
00271         lTRADEMGEN_ServiceContext.getUniformGenerator();
00272
00273     // Retrieve the default POS distribution
00274     const POSProbabilityMass_T& lDefaultPOSProbabilityMass =
00275         lTRADEMGEN_ServiceContext.getPOSProbabilityMass();
00276
00280     stdair::BasChronometer lDemandGeneration; lDemandGeneration.start();
00281     DemandParser::generateDemand (iDemandFilePath, lSEVMGR_Service_ptr,
00282         lSharedGenerator, lDefaultPOSProbabilityMass);
00283     const double lGenerationMeasure = lDemandGeneration.elapsed();
00284
00294     buildComplementaryLinks (lPersistentBomRoot);
00295

```

```

00296     // DEBUG
00297     STDAIR_LOG_DEBUG ("Demand generation time: " << lGenerationMeasure);
00298
00303     if (doesOwnStdairService == true) {
00304         //
00305         clonePersistentBom ();
00306     }
00307 }
00308
00309 // //////////////////////////////////////
00310 void TRADEMGEN_Service::buildSampleBom() {
00311
00312     // Retrieve the TraDemGen service context
00313     if (_trademgenServiceContext == NULL) {
00314         throw stdair::NonInitialisedServiceException ("The TraDemGen service has "
00315                                                         "not been initialised");
00316     }
00317     assert (_trademgenServiceContext != NULL);
00318
00319     // Retrieve the TraDemGen service context and whether it owns the Stdair
00320     // service
00321     TRADEMGEN_ServiceContext& lTRADEMGEN_ServiceContext =
00322         *_trademgenServiceContext;
00323     const bool doesOwnStdairService =
00324         lTRADEMGEN_ServiceContext.getOwnStdairServiceFlag();
00325
00326     // Retrieve the StdAir service object from the (TraDemGen) service context
00327     stdair::STDAIR_Service& lSTDAIR_Service =
00328         lTRADEMGEN_ServiceContext.getSTDAIR_Service();
00329
00330     // Retrieve the persistent BOM root object.
00331     stdair::BomRoot& lPersistentBomRoot =
00332         lSTDAIR_Service.getPersistentBomRoot();
00333
00338     if (doesOwnStdairService == true) {
00339         //
00340         lSTDAIR_Service.buildSampleBom();
00341     }
00342
00353     // Retrieve the shared generator
00354     stdair::RandomGeneration& lSharedGenerator =
00355         lTRADEMGEN_ServiceContext.getUniformGenerator();
00356
00357     // Retrieve the default POS distribution
00358     const POSProbabilityMass_T& lDefaultPOSProbabilityMass =
00359         lTRADEMGEN_ServiceContext.getPOSProbabilityMass();
00360
00361     // Retrieve the pointer on the SEVMgr service handler.
00362     SEVMGR::SEVMGR_ServicePtr_T lSEVMGR_Service_ptr =
00363         lTRADEMGEN_ServiceContext.getSEVMGR_ServicePtr();
00364
00365     // Delegate the BOM building to the dedicated service
00366     DemandManager::buildSampleBom (lSEVMGR_Service_ptr, lSharedGenerator,
00367                                     lDefaultPOSProbabilityMass);
00368     // Build the complementary links
00369     buildComplementaryLinks (lPersistentBomRoot);
00370
00375     if (doesOwnStdairService == true) {
00376         //
00377         clonePersistentBom ();
00378     }
00379 }
00380
00381 // //////////////////////////////////////
00382 void TRADEMGEN_Service::clonePersistentBom () {
00383
00384     // Retrieve the TraDemGen service context

```

```

00385     if (_trademgenServiceContext == NULL) {
00386         throw stdair::NonInitialisedServiceException ("The TraDemGen service has "
00387                                                         "not been initialised");
00388     }
00389     assert (_trademgenServiceContext != NULL);
00390
00391     // Retrieve the TraDemGen service context and whether it owns the Stdair
00392     // service
00393     TRADEMGEN_ServiceContext& lTRADEMGEN_ServiceContext =
00394         *_trademgenServiceContext;
00395     const bool doesOwnStdairService =
00396         lTRADEMGEN_ServiceContext.getOwnStdairServiceFlag();
00397
00398     // Retrieve the StdAir service object from the (TraDemGen) service context
00399     stdair::STDAIR_Service& lSTDAIR_Service =
00400         lTRADEMGEN_ServiceContext.getSTDAIR_Service();
00401
00402     if (doesOwnStdairService == true) {
00403         //
00404         lSTDAIR_Service.clonePersistentBom ();
00405     }
00406
00407     stdair::BomRoot& lBomRoot = lSTDAIR_Service.getBomRoot();
00408     buildComplementaryLinks (lBomRoot);
00409 }
00410
00411 // //////////////////////////////////////
00412 void TRADEMGEN_Service::buildComplementaryLinks (stdair::BomRoot& ioBomRoot) {
00413     // Currently, no more things to do by TRADEMGEN at that stage.
00414 }
00415
00416 // //////////////////////////////////////
00417 stdair::BookingRequestStruct TRADEMGEN_Service::
00418 buildSampleBookingRequest (const bool isForCRS) {
00419     // Retrieve the TraDemGen service context
00420     if (_trademgenServiceContext == NULL) {
00421         throw stdair::NonInitialisedServiceException ("The TraDemGen service has "
00422                                                         "not been initialised");
00423     }
00424     assert (_trademgenServiceContext != NULL);
00425
00426     TRADEMGEN_ServiceContext& lTRADEMGEN_ServiceContext =
00427         *_trademgenServiceContext;
00428
00429     // Retrieve the STDAIR service object from the (TraDemGen) service context
00430     stdair::STDAIR_Service& lSTDAIR_Service =
00431         lTRADEMGEN_ServiceContext.getSTDAIR_Service();
00432
00433     // Delegate the BOM building to the dedicated service
00434     return lSTDAIR_Service.buildSampleBookingRequest (isForCRS);
00435 }
00436
00437 // //////////////////////////////////////
00438 std::string TRADEMGEN_Service::
00439 jsonHandler (const stdair::JSONString& iJSONString) const {
00440     // Retrieve the TraDemGen service context
00441     if (_trademgenServiceContext == NULL) {
00442         throw stdair::NonInitialisedServiceException ("The TraDemGen service has "
00443                                                         "not been initialised");
00444     }
00445     assert (_trademgenServiceContext != NULL);
00446
00447     TRADEMGEN_ServiceContext& lTRADEMGEN_ServiceContext =
00448         *_trademgenServiceContext;
00449 }

```



```

00459     // Retrieve the pointer on the SEvMgr service handler.
00460     SEVMGR::SEVMGR_ServicePtr_T lSEVMGR_Service_ptr =
00461         lTRADEMGEN_ServiceContext.getSEVMGR_ServicePtr();
00462
00463     return lSEVMGR_Service_ptr->jsonHandler (iJSONString);
00464 }
00465
00466 // //////////////////////////////////////
00467 std::string TRADEMGEN_Service::csvDisplay() const {
00468
00469     // Retrieve the TraDemGen service context
00470     if (_trademgenServiceContext == NULL) {
00471         throw stdair::NonInitialisedServiceException ("The TraDemGen service has "
00472             "not been initialised");
00473     }
00474     assert (_trademgenServiceContext != NULL);
00475
00476     TRADEMGEN_ServiceContext& lTRADEMGEN_ServiceContext =
00477         *_trademgenServiceContext;
00478
00479     // Retrieve the pointer on the SEvMgr service handler.
00480     SEVMGR::SEVMGR_ServicePtr_T lSEVMGR_Service_ptr =
00481         lTRADEMGEN_ServiceContext.getSEVMGR_ServicePtr();
00482
00483     // Delegate the BOM building to the dedicated service
00484     return BomDisplay::csvDisplay (lSEVMGR_Service_ptr);
00485 }
00486
00487 // //////////////////////////////////////
00488 std::string TRADEMGEN_Service::list() const {
00489
00490     // Retrieve the TraDemGen service context
00491     if (_trademgenServiceContext == NULL) {
00492         throw stdair::NonInitialisedServiceException ("The TraDemGen service has "
00493             "not been initialised");
00494     }
00495     assert (_trademgenServiceContext != NULL);
00496
00497     TRADEMGEN_ServiceContext& lTRADEMGEN_ServiceContext =
00498         *_trademgenServiceContext;
00499
00500     // Retrieve the pointer on the SEvMgr service handler.
00501     SEVMGR::SEVMGR_ServicePtr_T lSEVMGR_Service_ptr =
00502         lTRADEMGEN_ServiceContext.getSEVMGR_ServicePtr();
00503
00504     // Delegate the BOM building to the dedicated service
00505     return lSEVMGR_Service_ptr->list ();
00506 }
00507
00508 // //////////////////////////////////////
00509 std::string TRADEMGEN_Service::
00510 list(const stdair::EventType::EN_EventType& iEventType) const {
00511
00512     // Retrieve the TraDemGen service context
00513     if (_trademgenServiceContext == NULL) {
00514         throw stdair::NonInitialisedServiceException ("The TraDemGen service has "
00515             "not been initialised");
00516     }
00517     assert (_trademgenServiceContext != NULL);
00518
00519     TRADEMGEN_ServiceContext& lTRADEMGEN_ServiceContext =
00520         *_trademgenServiceContext;
00521
00522     // Retrieve the pointer on the SEvMgr service handler.

```

```

00526     SEVMGR::SEVMGR_ServicePtr_T lSEVMGR_Service_ptr =
00527         lTRADEMGEN_ServiceContext.getSEVMGR_ServicePtr();
00528
00529     // Delegate the BOM building to the dedicated service
00530     return lSEVMGR_Service_ptr->list (iEventType);
00531
00532 }
00533
00534 // ////////////////////////////////////////
00535 void TRADEMGEN_Service::displayAirlineListFromDB() const {
00536
00537     // Retrieve the TraDemGen service context
00538     if (_trademgenServiceContext == NULL) {
00539         throw stdair::NonInitialisedServiceException ("The TraDemGen service has "
00540             "not been initialised");
00541     }
00542     assert (_trademgenServiceContext != NULL);
00543     // TRADEMGEN_ServiceContext& lTRADEMGEN_ServiceContext =
00544     // *_trademgenServiceContext;
00545
00546     // Get the date-time for the present time
00547     boost::posix_time::ptime lNowDateTime =
00548         boost::posix_time::second_clock::local_time();
00549     //boost::gregorian::date lNowDate = lNowDateTime.date();
00550
00551     // DEBUG
00552     STDAIR_LOG_DEBUG (std::endl
00553         << "=====
00554         << std::endl
00555         << lNowDateTime);
00556
00557     // Delegate the query execution to the dedicated command
00558     stdair::BasChronometer lAirListChronometer;
00559     lAirListChronometer.start();
00560
00561     // Retrieve the database session handler
00562     stdair::DBSession_T& lDBSession =
00563         stdair::DBSessionManager::instance().getDBSession();
00564
00565     // Prepare and execute the select statement
00566     stdair::AirlineStruct lAirline;
00567     stdair::DBRequestStatement_T lSelectStatement (lDBSession);
00568     stdair::DBManagerForAirlines::prepareSelectStatement (lDBSession,
00569         lSelectStatement,
00570         lAirline);
00571
00572     // Prepare the SQL request corresponding to the select statement
00573     bool hasStillData = true;
00574     unsigned int idx = 0;
00575     while (hasStillData == true) {
00576         hasStillData =
00577             stdair::DBManagerForAirlines::iterateOnStatement (lSelectStatement,
00578                 lAirline);
00579
00580         // DEBUG
00581         STDAIR_LOG_DEBUG ("[" << idx << "]: " << lAirline);
00582
00583         // Iteration
00584         ++idx;
00585     }
00586
00587     const double lAirListMeasure = lAirListChronometer.elapsed();
00588
00589     // DEBUG
00590     STDAIR_LOG_DEBUG ("Sample service for airline list retrieval: "
00591         << lAirListMeasure);
00592 }

```

```

00593
00594 ///////////////////////////////////////////////////////////////////
00595 const stdair::Count_T& TRADEMGEN_Service::
00596 getExpectedTotalNumberOfRequestsToBeGenerated() const {
00597
00598     // Retrieve the TraDemGen service context
00599     assert (_trademgenServiceContext != NULL);
00600     TRADEMGEN_ServiceContext& lTRADEMGEN_ServiceContext =
00601         *_trademgenServiceContext;
00602
00603     // Retrieve the SEvMgr service context
00604     SEVMGR::SEVMGR_Service& lSEVMGR_Service =
00605         lTRADEMGEN_ServiceContext.getSEVMGR_Service();
00606
00607     // Delegate the call to the dedicated command
00608     const stdair::Count_T& oExpectedTotalNumberOfRequestsToBeGenerated =
00609         lSEVMGR_Service.getExpectedTotalNumberOfEventsToBeGenerated (stdair::EventT
00610     ype::BKG_REQ);
00611
00612     //
00613     return oExpectedTotalNumberOfRequestsToBeGenerated;
00614 }
00615
00616 ///////////////////////////////////////////////////////////////////
00617 const stdair::Count_T& TRADEMGEN_Service::
00618 getActualTotalNumberOfRequestsToBeGenerated() const {
00619
00620     // Retrieve the TraDemGen service context
00621     assert (_trademgenServiceContext != NULL);
00622     TRADEMGEN_ServiceContext& lTRADEMGEN_ServiceContext =
00623         *_trademgenServiceContext;
00624
00625     // Retrieve the SEvMgr service context
00626     SEVMGR::SEVMGR_Service& lSEVMGR_Service =
00627         lTRADEMGEN_ServiceContext.getSEVMGR_Service();
00628
00629     // Delegate the call to the dedicated command
00630     const stdair::Count_T& oActualTotalNumberOfRequestsToBeGenerated =
00631         lSEVMGR_Service.getActualTotalNumberOfEventsToBeGenerated (stdair::EventTyp
00632     e::BKG_REQ);
00633
00634     //
00635     return oActualTotalNumberOfRequestsToBeGenerated;
00636 }
00637
00638 ///////////////////////////////////////////////////////////////////
00639 const bool TRADEMGEN_Service::
00640 stillHavingRequestsToBeGenerated (const stdair::DemandStreamKeyStr_T& iKey,
00641     stdair::ProgressStatusSet& ioPSS,
00642     const stdair::DemandGenerationMethod& iDemand
00643     GenerationMethod) const {
00644
00645     // Retrieve the TraDemGen service context
00646     assert (_trademgenServiceContext != NULL);
00647     TRADEMGEN_ServiceContext& lTRADEMGEN_ServiceContext =
00648         *_trademgenServiceContext;
00649
00650     // Retrieve the pointer on the SEvMgr service handler.
00651     SEVMGR::SEVMGR_ServicePtr_T lSEVMGR_Service_ptr =
00652         lTRADEMGEN_ServiceContext.getSEVMGR_ServicePtr();
00653
00654     // Delegate the call to the dedicated command
00655     const bool oStillHavingRequestsToBeGenerated =
00656         DemandManager::stillHavingRequestsToBeGenerated (lSEVMGR_Service_ptr,
00657             iKey, ioPSS,
00658             iDemandGenerationMethod);
00659
00660 }

```

```

00657     //
00658     return oStillHavingRequestsToBeGenerated;
00659 }
00660
00661 // //////////////////////////////////////
00662 stdair::Count_T TRADEMGEN_Service::
00663 generateFirstRequests (const stdair::DemandGenerationMethod& iDemandGenerationM
ethod) const {
00664
00665     // Retrieve the TraDemGen service context
00666     assert (_trademgenServiceContext != NULL);
00667     TRADEMGEN_ServiceContext& lTRADEMGEN_ServiceContext =
00668         *_trademgenServiceContext;
00669
00670     // Retrieve the pointer on the SEvMgr service handler.
00671     SEVMGR::SEVMGR_ServicePtr_T lSEVMGR_Service_ptr =
00672         lTRADEMGEN_ServiceContext.getSEVMGR_ServicePtr();
00673
00674     // Retrieve the random generator
00675     stdair::RandomGeneration& lGenerator =
00676         lTRADEMGEN_ServiceContext.getUniformGenerator();
00677
00678     // Delegate the call to the dedicated command
00679     const stdair::Count_T& oActualTotalNbOfEvents =
00680         DemandManager::generateFirstRequests (lSEVMGR_Service_ptr, lGenerator,
00681                                                 iDemandGenerationMethod);
00682
00683     //
00684     return oActualTotalNbOfEvents;
00685 }
00686
00687 // //////////////////////////////////////
00688 stdair::BookingRequestPtr_T TRADEMGEN_Service::
00689 generateNextRequest (const stdair::DemandStreamKeyStr_T& iKey,
00690                     const stdair::DemandGenerationMethod& iDemandGenerationMet
hod) const {
00691
00692     // Retrieve the TraDemGen service context
00693     assert (_trademgenServiceContext != NULL);
00694     TRADEMGEN_ServiceContext& lTRADEMGEN_ServiceContext =
00695         *_trademgenServiceContext;
00696
00697     // Retrieve the pointer on the SEvMgr service handler.
00698     SEVMGR::SEVMGR_ServicePtr_T lSEVMGR_Service_ptr =
00699         lTRADEMGEN_ServiceContext.getSEVMGR_ServicePtr();
00700
00701     // Retrieve the random generator
00702     stdair::RandomGeneration& lGenerator =
00703         lTRADEMGEN_ServiceContext.getUniformGenerator();
00704
00705     // Delegate the call to the dedicated command
00706     return DemandManager::generateNextRequest (lSEVMGR_Service_ptr,
00707                                                 lGenerator, iKey,
00708                                                 iDemandGenerationMethod);
00709 }
00710
00711 // //////////////////////////////////////
00712 stdair::ProgressStatusSet TRADEMGEN_Service::
00713 popEvent (stdair::EventStruct& ioEventStruct) const {
00714
00715     // Retrieve the TraDemGen service context
00716     assert (_trademgenServiceContext != NULL);
00717     TRADEMGEN_ServiceContext& lTRADEMGEN_ServiceContext =
00718         *_trademgenServiceContext;
00719
00720     // Retrieve the SEvMgr service context
00721     SEVMGR::SEVMGR_Service& lSEVMGR_Service =

```

```

00722         lTRADEMGEN_ServiceContext.getSEVMGR_Service();
00723
00724         // Extract the next event from the queue
00725         return lSEVMGR_Service.popEvent (ioEventStruct);
00726     }
00727
00728     // //////////////////////////////////////
00729     bool TRADEMGEN_Service::isQueueDone() const {
00730
00731         // Retrieve the TraDemGen service context
00732         assert (_trademgenServiceContext != NULL);
00733         TRADEMGEN_ServiceContext& lTRADEMGEN_ServiceContext =
00734             *_trademgenServiceContext;
00735
00736         // Retrieve the SEvMgr service context
00737         SEVMGR::SEVMGR_Service& lSEVMGR_Service =
00738             lTRADEMGEN_ServiceContext.getSEVMGR_Service();
00739
00740         // Calculates whether the event queue has been fully emptied
00741         const bool isQueueDone = lSEVMGR_Service.isQueueDone();
00742
00743         //
00744         return isQueueDone;
00745     }
00746
00747     // //////////////////////////////////////
00748     bool TRADEMGEN_Service::
00749     generateCancellation (const stdair::TravelSolutionStruct& iTravelSolution,
00750                         const stdair::PartySize_T& iPartySize,
00751                         const stdair::DateTime_T& iRequestTime,
00752                         const stdair::Date_T& iDepartureDate) const {
00753
00754         // Retrieve the TraDemGen service context
00755         assert (_trademgenServiceContext != NULL);
00756         TRADEMGEN_ServiceContext& lTRADEMGEN_ServiceContext =
00757             *_trademgenServiceContext;
00758
00759         // Retrieve the random generator
00760         stdair::RandomGeneration& lGenerator =
00761             lTRADEMGEN_ServiceContext.getUniformGenerator();
00762
00763         // Build an event structure with the default constructor.
00764         stdair::EventStruct lEventStruct;
00765         stdair::EventStruct& lRefEventStruct = lEventStruct;
00766
00767         // Generate the cancellation event
00768         const bool hasCancellationBeenGenerated =
00769             DemandManager::generateCancellation (lGenerator, iTravelSolution,
00770                                                 iPartySize, iRequestTime,
00771                                                 iDepartureDate, lRefEventStruct);
00772
00773         // If the cancellation has been not sucessfully gerenerated, return.
00774         if (hasCancellationBeenGenerated == false) {
00775             return hasCancellationBeenGenerated;
00776         }
00777         assert (hasCancellationBeenGenerated == true);
00778
00779         // Retrieve the SEvMgr service context
00780         SEVMGR::SEVMGR_Service& lSEVMGR_Service =
00781             lTRADEMGEN_ServiceContext.getSEVMGR_Service();
00782
00783         // Add the gerenerated cancellation event into the queue
00784         lSEVMGR_Service.addEvent (lRefEventStruct);
00785
00786         // Update the status of cancellation events within the event queue.
00787         const bool hasProgressStatus =
00788             lSEVMGR_Service.hasProgressStatus(stdair::EventType::CX);

```

```

00796     if (hasProgressStatus == false) {
00797         const stdair::Count_T lCancellationNumber = 1;
00798         lSEVMGR_Service.addStatus (stdair::EventType::CX, lCancellationNumber);
00799     } else {
00800         stdair::Count_T lCurrentCancellationNumber =
00801             lSEVMGR_Service.getActualTotalNumberOfEventsToBeGenerated (stdair::EventT
ype::CX);
00802         lCurrentCancellationNumber++;
00803         lSEVMGR_Service.updateStatus (stdair::EventType::CX, lCurrentCancellationNu
mber);
00804     }
00805
00806     return hasCancellationBeenGenerated;
00807 }
00808 }
00809
00810 // //////////////////////////////////////
00811 void TRADEMGEN_Service::reset() const {
00812
00813     // Retrieve the TraDemGen service context
00814     assert (_trademgenServiceContext != NULL);
00815     TRADEMGEN_ServiceContext& lTRADEMGEN_ServiceContext =
00816         *_trademgenServiceContext;
00817
00818     // Retrieve the pointer on the SEvMgr service handler.
00819     SEVMGR::SEVMGR_ServicePtr_T lSEVMGR_Service_ptr =
00820         lTRADEMGEN_ServiceContext.getSEVMGR_ServicePtr();
00821
00822     // Retrieve the shared generator
00823     stdair::RandomGeneration& lSharedGenerator =
00824         lTRADEMGEN_ServiceContext.getUniformGenerator();
00825
00826     // Delegate the call to the dedicated command
00827     DemandManager::reset (lSEVMGR_Service_ptr,
00828         lSharedGenerator.getBaseGenerator());
00829 }
00830
00831 const stdair::ProgressStatus& TRADEMGEN_Service::getProgressStatus() const {
00832
00833     // Retrieve the TraDemGen service context
00834     assert (_trademgenServiceContext != NULL);
00835     TRADEMGEN_ServiceContext& lTRADEMGEN_ServiceContext =
00836         *_trademgenServiceContext;
00837
00838     // Retrieve the pointer on the SEvMgr service handler.
00839     SEVMGR::SEVMGR_ServicePtr_T lSEVMGR_Service_ptr =
00840         lTRADEMGEN_ServiceContext.getSEVMGR_ServicePtr();
00841
00842     // Delegate the call to the dedicated service
00843     return lSEVMGR_Service_ptr->getStatus();
00844 }
00845
00846 }
00847
00848 const stdair::ProgressStatus& TRADEMGEN_Service::
00849 getProgressStatus (const stdair::EventType::EN_EventType& iEventType) const {
00850
00851     // Retrieve the TraDemGen service context
00852     assert (_trademgenServiceContext != NULL);
00853     TRADEMGEN_ServiceContext& lTRADEMGEN_ServiceContext =
00854         *_trademgenServiceContext;
00855
00856     // Retrieve the pointer on the SEvMgr service handler.
00857     SEVMGR::SEVMGR_ServicePtr_T lSEVMGR_Service_ptr =
00858         lTRADEMGEN_ServiceContext.getSEVMGR_ServicePtr();
00859
00860

```

```

00861     // Delegate the call to the dedicated service
00862     return lSEVMGR_Service_ptr->getStatus(iEventType);
00863
00864 }
00865
00866 bool TRADEMGEN_Service::
00867 hasDemandStream (const stdair::DemandStreamKeyStr_T& iDemandStreamKey) const {
00868
00869     // Retrieve the TraDemGen service context
00870     assert (_trademgenServiceContext != NULL);
00871     TRADEMGEN_ServiceContext& lTRADEMGEN_ServiceContext =
00872         *_trademgenServiceContext;
00873
00874     // Retrieve the pointer on the SEVMgr service handler.
00875     SEVMGR::SEVMGR_ServicePtr_T lSEVMGR_Service_ptr =
00876         lTRADEMGEN_ServiceContext.getSEVMGR_ServicePtr();
00877
00878     // Delegate the call to the dedicated service
00879     return lSEVMGR_Service_ptr->hasEventGenerator<DemandStream,
00880         stdair::DemandStreamKeyStr_T>(i
00881     DemandStreamKey);
00882 }
00883
00884 std::string TRADEMGEN_Service::displayDemandStream () const {
00885
00886     // Retrieve the TraDemGen service context
00887     assert (_trademgenServiceContext != NULL);
00888     TRADEMGEN_ServiceContext& lTRADEMGEN_ServiceContext =
00889         *_trademgenServiceContext;
00890
00891     // Retrieve the pointer on the SEVMgr service handler.
00892     SEVMGR::SEVMGR_ServicePtr_T lSEVMGR_Service_ptr =
00893         lTRADEMGEN_ServiceContext.getSEVMGR_ServicePtr();
00894
00895     // Delegate the call to the dedicated service
00896     const DemandStreamList_T lDemandStreamList =
00897         lSEVMGR_Service_ptr->getEventGeneratorList<DemandStream>();
00898
00899     // Output stream to store the display of demand streams.
00900     std::ostream oStream;
00901
00902     for (DemandStreamList_T::const_iterator itDemandStream =
00903         lDemandStreamList.begin(); itDemandStream !=
00904         lDemandStreamList.end(); itDemandStream++) {
00905         DemandStream* lDemandStream_ptr = *itDemandStream;
00906         assert (lDemandStream_ptr != NULL);
00907         oStream << lDemandStream_ptr->describeKey() << std::endl;
00908     }
00909     return oStream.str();
00910 }
00911 }
00912
00913 }
00914
00915 namespace SEVMGR {
00916
00917 // //////////////////////////////////////
00918 template<class EventGenerator>
00919 void SEVMGR_Service::addEventGenerator (EventGenerator& iEventGenerator) const
00920 {
00921     // Retrieve the StdAir service
00922     const stdair::STDAIR_Service& lSTDAIR_Service =
00923         this->getSTDAIR_Service();
00924
00925     // Retrieve the BOM root object instance
00926     stdair::BomRoot& lPersistentBomRoot =
00927         lSTDAIR_Service.getPersistentBomRoot();

```

```

00936
00937     // Link the DemandStream to its parent (EventQueue)
00938     stdair::FacBomManager::linkWithParent (lPersistentBomRoot, iEventGenerator);
00939
00940     // Add the DemandStream to the dedicated list and map
00941     stdair::FacBomManager::addToListAndMap (lPersistentBomRoot,
00942                                             iEventGenerator);
00943
00944 }
00945
00946 // //////////////////////////////////////
00947 template<class EventGenerator, class Key>
00948 EventGenerator& SEVMGR_Service::getEventGenerator(const Key& iKey) const {
00949
00950     // Retrieve the StdAir service
00951     const stdair::STDAIR_Service& lSTDAIR_Service =
00952         this->getSTDAIR_Service();
00953
00954     // Retrieve the BOM root object instance
00955     stdair::BomRoot& lPersistentBomRoot =
00956         lSTDAIR_Service.getPersistentBomRoot();
00957
00958     // Retrieve the DemandStream which corresponds to the given key.
00959     EventGenerator& lEventGenerator =
00960         stdair::BomManager::getObject<EventGenerator>(lPersistentBomRoot,
00961                                                       iKey);
00962
00963     return lEventGenerator;
00964
00965 }
00966
00967 // //////////////////////////////////////
00968 template<class EventGenerator, class Key>
00969 bool SEVMGR_Service::hasEventGenerator(const Key& iKey) const {
00970
00971     bool hasEventGenerator = true;
00972
00973     // Retrieve the StdAir service
00974     const stdair::STDAIR_Service& lSTDAIR_Service =
00975         this->getSTDAIR_Service();
00976
00977     // Retrieve the BOM root object instance
00978     stdair::BomRoot& lPersistentBomRoot =
00979         lSTDAIR_Service.getPersistentBomRoot();
00980
00981     // Retrieve the DemandStream which corresponds to the given key.
00982     EventGenerator* lEventGenerator_ptr =
00983         stdair::BomManager::getObjectPtr<EventGenerator>(lPersistentBomRoot,
00984                                                           iKey);
00985     if (lEventGenerator_ptr == NULL) {
00986         hasEventGenerator = false;
00987     }
00988
00989     return hasEventGenerator;
00990
00991 }
00992
00993 // //////////////////////////////////////
00994 template<class EventGenerator>
00995 const std::list<EventGenerator*> SEVMGR_Service::getEventGeneratorList() const
00996 {
00997     // Retrieve the StdAir service
00998     const stdair::STDAIR_Service& lSTDAIR_Service =
00999         this->getSTDAIR_Service();
01000
01001     // Retrieve the BOM root object instance

```



```

01002     stdair::BomRoot& lPersistentBomRoot =
01003         lSTDAIR_Service.getPersistentBomRoot();
01004
01005     // Retrieve the DemandStream list
01006     const std::list<EventGenerator*> lEventGeneratorList =
01007         stdair::BomManager::getList<EventGenerator> (lPersistentBomRoot);
01008
01009     return lEventGeneratorList;
01010 }
01011
01012 // //////////////////////////////////////
01013 template<class EventGenerator>
01014 bool SEVMGR_Service::hasEventGeneratorList() const {
01015
01016     // Retrieve the StdAir service
01017     const stdair::STDAIR_Service& lSTDAIR_Service =
01018         this->getSTDAIR_Service();
01019
01020     // Retrieve the BOM root object instance
01021     stdair::BomRoot& lPersistentBomRoot =
01022         lSTDAIR_Service.getPersistentBomRoot();
01023
01024     const bool hasListEventGenerator =
01025         stdair::BomManager::hasList<EventGenerator> (lPersistentBomRoot);
01026
01027     return hasListEventGenerator;
01028 }
01029
01030 // //////////////////////////////////////
01031 template void SEVMGR_Service::
01032 addEventGenerator<TRADEMGEN::DemandStream> (TRADEMGEN::DemandStream&) const;
01033
01034 template TRADEMGEN::DemandStream& SEVMGR_Service::
01035 getEventGenerator<TRADEMGEN::DemandStream, stdair::DemandStreamKeyStr_T> (const
stdair::DemandStreamKeyStr_T&) const;
01036
01037 template bool SEVMGR_Service::
01038 hasEventGenerator<TRADEMGEN::DemandStream, stdair::DemandStreamKeyStr_T> (const
stdair::DemandStreamKeyStr_T&) const;
01039
01040 template const TRADEMGEN::DemandStreamList_T SEVMGR_Service::
01041 getEventGeneratorList<TRADEMGEN::DemandStream> () const;
01042
01043 template bool SEVMGR_Service::hasEventGeneratorList<TRADEMGEN::DemandStream> ()
const;
01044 // //////////////////////////////////////
01045
01046 }
01047
01048 }

```

25.103 trademgen/service/TRADEMGEN_ServiceContext.cpp File Reference

```
#include <cassert>
#include <sstream>
#include <stdair/STDAIR_Service.hpp>
#include <stdair/basic/BasConst_General.hpp>
#include <trademgen/basic/BasConst_DemandGeneration.hpp>
#include <trademgen/service/TRADEMGEN_ServiceContext.hpp>
```

Namespaces

- namespace [TRADEMGEN](#)

25.104 TRADEMGEN_ServiceContext.cpp

```

00001 // //////////////////////////////////////
00002 // Import section
00003 // //////////////////////////////////////
00004 // STL
00005 #include <cassert>
00006 #include <sstream>
00007 // StdAir
00008 #include <stdair/STDAIR_Service.hpp>
00009 #include <stdair/basic/BasConst_General.hpp>
00010 // TraDemGen
00011 #include <trademgen/basic/BasConst_DemandGeneration.hpp>
00012 #include <trademgen/service/TRADEMGEN_ServiceContext.hpp>
00013
00014 namespace TRADEMGEN {
00015
00016 // //////////////////////////////////////
00017 TRADEMGEN_ServiceContext::TRADEMGEN_ServiceContext ()
00018     : _ownStdairService (false), _uniformGenerator (stdair::DEFAULT_RANDOM_SEED),
00019
00019     _posProbabilityMass (DEFAULT_POS_PROBALILITY_MASS) {
00020 }
00021
00022 // //////////////////////////////////////
00023 TRADEMGEN_ServiceContext::
00024 TRADEMGEN_ServiceContext (const TRADEMGEN_ServiceContext& iServiceContext)
00025     : _ownStdairService (false), _uniformGenerator (stdair::DEFAULT_RANDOM_SEED),
00026
00026     _posProbabilityMass (DEFAULT_POS_PROBALILITY_MASS) {
00027 }
00028
00029 // //////////////////////////////////////
00030 TRADEMGEN_ServiceContext::
00031 TRADEMGEN_ServiceContext (const stdair::RandomSeed_T& iRandomSeed)
00032     : _ownStdairService (false), _uniformGenerator (iRandomSeed),
00033     _posProbabilityMass (DEFAULT_POS_PROBALILITY_MASS) {
00034 }
00035
00036 // //////////////////////////////////////
00037 TRADEMGEN_ServiceContext::~TRADEMGEN_ServiceContext () {
00038 }
00039
00040 // //////////////////////////////////////
00041 const std::string TRADEMGEN_ServiceContext::shortDisplay() const {
00042     std::ostringstream oStr;
00043     oStr << "TRADEMGEN_ServiceContext -- Owns StdAir service: "
00044         << _ownStdairService << " -- Generator: " << _uniformGenerator;
00045     return oStr.str();
00046 }
00047
00048 // //////////////////////////////////////
00049 const std::string TRADEMGEN_ServiceContext::display() const {
00050     std::ostringstream oStr;
00051     oStr << shortDisplay();
00052     return oStr.str();
00053 }
00054
00055 // //////////////////////////////////////
00056 const std::string TRADEMGEN_ServiceContext::describe() const {
00057     return shortDisplay();
00058 }
00059
00060 // //////////////////////////////////////
00061 void TRADEMGEN_ServiceContext::reset() {
00062
00063     // The shared_ptr<>::reset() method drops the refcount by one.

```

```
00064    // If the count result is dropping to zero, the resource pointed to
00065    // by the shared_ptr<> will be freed.
00066
00067    // Reset the stdair shared pointer
00068    _stdairService.reset();
00069
00070    // Reset the sevmgr shared pointer
00071    _sevmgrService.reset();
00072 }
00073
00074 }
```

25.105 trademgen/service/TRADEMGEN_ServiceContext.hpp File Reference

```
#include <string>
#include <boost/shared_ptr.hpp>
#include <stdair/stdair_basic_types.hpp>
#include <stdair/stdair_service_types.hpp>
#include <stdair/basic/RandomGeneration.hpp>
#include <stdair/bom/BookingRequestTypes.hpp>
#include <stdair/service/ServiceAbstract.hpp>
#include <sevmgr/SEVMGR_Types.hpp>
#include <trademgen/TRADEMGEN_Types.hpp>
#include <trademgen/basic/DemandCharacteristicsTypes.hpp>
```

Classes

- class [TRADEMGEN::TRADEMGEN_ServiceContext](#)
Class holding the context of the Trademgen services.

Namespaces

- namespace [stdair](#)
Forward declarations.
- namespace [TRADEMGEN](#)

25.106 TRADEMGEN_ServiceContext.hpp

```

00001 #ifndef __TRADEMGEN_SVC_TRADEMGENSEVICECONTEXT_HPP
00002 #define __TRADEMGEN_SVC_TRADEMGENSEVICECONTEXT_HPP
00003
00004 // //////////////////////////////////////
00005 // Import section
00006 // //////////////////////////////////////
00007 // STL
00008 #include <string>
00009 // Boost
00010 #include <boost/shared_ptr.hpp>
00011 // StdAir
00012 #include <stdair/stdair_basic_types.hpp>
00013 #include <stdair/stdair_service_types.hpp>
00014 #include <stdair/basic/RandomGeneration.hpp>
00015 #include <stdair/bom/BookingRequestTypes.hpp>
00016 #include <stdair/service/ServiceAbstract.hpp>
00017 // SEvMgr
00018 #include <sevmgr/SEVMGR_Types.hpp>
00019 // TraDemGen
00020 #include <trademgen/TRADEMGEN_Types.hpp>
00021 #include <trademgen/basic/DemandCharacteristicsTypes.hpp>
00022
00023 // Forward declarations
00024 namespace stdair {
00025     struct DemandCharacteristics;
00026     struct DemandDistribution;
00027 }
00028
00029 namespace TRADEMGEN {
00030
00031     class TRADEMGEN_ServiceContext : public stdair::ServiceAbstract {
00032     friend class TRADEMGEN_Service;
00033     friend class FacTRADEMGENSEVICEContext;
00034
00035     private:
00036         // ////////////////////////////////// Getters //////////////////////////////////
00037         stdair::STDAIR_ServicePtr_T getSTDAIR_ServicePtr() const {
00038             return _stdairService;
00039         }
00040
00041         stdair::STDAIR_Service& getSTDAIR_Service() const {
00042             assert (_stdairService != NULL);
00043             return *_stdairService;
00044         }
00045
00046         const bool getOwnStdairServiceFlag() const {
00047             return _ownStdairService;
00048         }
00049
00050         stdair::RandomGeneration& getUniformGenerator() {
00051             return _uniformGenerator;
00052         }
00053
00054         const POSProbabilityMass_T& getPOSProbabilityMass() const {
00055             return _posProbabilityMass;
00056         }
00057
00058         SEVMGR::SEVMGR_ServicePtr_T getSEVMGR_ServicePtr() const {
00059             return _sevmgrService;
00060         }
00061
00062         SEVMGR::SEVMGR_Service& getSEVMGR_Service() const {
00063             assert (_sevmgrService != NULL);
00064             return *_sevmgrService;
00065         }
00066     }

```

```

00095
00096
00097 private:
00098     // ////////// Setters //////////
00102     void setSTDAIR_Service (stdair::STDAIR_ServicePtr_T ioSTDAIR_ServicePtr,
00103                             const bool iOwnStdairService) {
00104         _stdairService = ioSTDAIR_ServicePtr;
00105         _ownStdairService = iOwnStdairService;
00106     }
00107
00111     void setSEVMGR_Service (SEVMGR::SEVMGR_ServicePtr_T ioSEVMGR_ServicePtr) {
00112         _sevmgrService = ioSEVMGR_ServicePtr;
00113     }
00114
00115
00116 private:
00117     // ////////// Display Methods //////////
00121     const std::string shortDisplay() const;
00122
00126     const std::string display() const;
00127
00131     const std::string describe() const;
00132
00133
00134 private:
00136
00139     TRADEMGEN_ServiceContext (const stdair::RandomSeed_T&);
00143     TRADEMGEN_ServiceContext ();
00147     TRADEMGEN_ServiceContext (const TRADEMGEN_ServiceContext&);
00148
00152     ~TRADEMGEN_ServiceContext ();
00153
00157     void reset ();
00158
00159
00160 private:
00161     // ////////////////////////////////// Children //////////////////////////////////
00165     stdair::STDAIR_ServicePtr_T _stdairService;
00166
00170     SEVMGR::SEVMGR_ServicePtr_T _sevmgrService;
00171
00175     bool _ownStdairService;
00176
00177
00178 private:
00179     // ////////////////////////////////// Attributes //////////////////////////////////
00186     stdair::RandomGeneration _uniformGenerator;
00187
00191     const POSProbabilityMass_T _posProbabilityMass;
00192 };
00193
00194 }
00195 #endif // __TRADEMGEN_SVC_TRADEMGENSERVICECONTEXT_HPP

```

25.107 trademgen/TRADEMGEN_Abstract.hpp File Reference

```
#include <istream>
#include <ostream>
#include <sstream>
#include <string>
```

Classes

- struct [TRADEMGEN::TRADEMGEN_Abstract](#)

Namespaces

- namespace [TRADEMGEN](#)

Functions

- template<class charT , class traits >
std::basic_ostream< charT, traits > & [operator<<](#) (std::basic_ostream< charT, traits > &ioOut,
const [TRADEMGEN::TRADEMGEN_Abstract](#) &iStructure)
- template<class charT , class traits >
std::basic_istream< charT, traits > & [operator>>](#) (std::basic_istream< charT, traits > &ioIn,
[TRADEMGEN::TRADEMGEN_Abstract](#) &ioStructure)

25.107.1 Function Documentation

25.107.1.1 `template<class charT , class traits > std::basic_ostream<charT, traits>& operator<< (std::basic_ostream< charT, traits > & ioOut, const TRADEMGEN::TRADEMGEN_Abstract & iStructure) [inline]`

Piece of code given by Nicolai M. Josuttis, Section 13.12.1 "Implementing Output Operators" (p653) of his book "The C++ Standard Library: A Tutorial and Reference", published by Addison-Wesley.

Definition at line 49 of file [TRADEMGEN_Abstract.hpp](#).

25.107.1.2 `template<class charT , class traits > std::basic_istream<charT, traits>& operator>> (std::basic_istream< charT, traits > & ioIn, TRADEMGEN::TRADEMGEN_Abstract & ioStructure) [inline]`

Piece of code given by Nicolai M. Josuttis, Section 13.12.1 "Implementing Output Operators" (pp655-657) of his book "The C++ Standard Library: A Tutorial and Reference", published by Addison-Wesley.

Definition at line 77 of file [TRADEMGEN_Abstract.hpp](#).

References [TRADEMGEN::TRADEMGEN_Abstract::fromStream\(\)](#).

25.108 TRADEMGEN_Abstract.hpp

```

00001 #ifndef __TRADEMGEN_TRADEMGEN_ABSTRACT_HPP
00002 #define __TRADEMGEN_TRADEMGEN_ABSTRACT_HPP
00003
00004 // //////////////////////////////////////
00005 // Import section
00006 // //////////////////////////////////////
00007 // STL
00008 #include <istream>
00009 #include <ostream>
00010 #include <sstream>
00011 #include <string>
00012
00013 namespace TRADEMGEN {
00014
00016     struct TRADEMGEN_Abstract {
00017     public:
00018         // ////////////////////////////////// Display support methods //////////////////////////////////
00021         virtual void toStream (std::ostream& ioOut) const = 0;
00022
00025         virtual void fromStream (std::istream& ioIn) = 0;
00026
00028         virtual std::string toString() const = 0;
00029
00030     protected:
00033         TRADEMGEN_Abstract () {}
00034         TRADEMGEN_Abstract (const TRADEMGEN_Abstract&) {}
00035
00037         virtual ~TRADEMGEN_Abstract() {}
00038     };
00039 }
00040
00046 template <class charT, class traits>
00047 inline
00048 std::basic_ostream<charT, traits>&
00049 operator<< (std::basic_ostream<charT, traits>& ioOut,
00050           const TRADEMGEN::TRADEMGEN_Abstract& iStructure) {
00056     std::basic_ostringstream<charT,traits> ostr;
00057     ostr.copyfmt (ioOut);
00058     ostr.width (0);
00059
00060     // Fill string stream
00061     iStructure.toStream (ostr);
00062
00063     // Print string stream
00064     ioOut << ostr.str();
00065
00066     return ioOut;
00067 }
00068
00074 template <class charT, class traits>
00075 inline
00076 std::basic_istream<charT, traits>&
00077 operator>> (std::basic_istream<charT, traits>& ioIn,
00078            TRADEMGEN::TRADEMGEN_Abstract& ioStructure) {
00079     // Fill Bom object with input stream
00080     ioStructure.fromStream (ioIn);
00081     return ioIn;
00082 }
00083
00084 #endif // __TRADEMGEN_TRADEMGEN_ABSTRACT_HPP

```

25.109 trademgen/TRADEMGEN_Exceptions.hpp File Reference

```
#include <exception>
#include <stdair/stdair_exceptions.hpp>
```

Classes

- class [TRADEMGEN::TrademgenGenerationException](#)
- class [TRADEMGEN::DemandInputFileNotFoundException](#)
- class [TRADEMGEN::IndexOutOfRangeException](#)

Namespaces

- namespace [TRADEMGEN](#)

25.110 TRADEMGEN_Exceptions.hpp

```
00001 #ifndef __TRADEMGEN_TRADEMGEN_EXCEPTIONS_HPP
00002 #define __TRADEMGEN_TRADEMGEN_EXCEPTIONS_HPP
00003
00004 // //////////////////////////////////////
00005 // Import section
00006 // //////////////////////////////////////
00007 // STL
00008 #include <exception>
00009 // StdAir
00010 #include <stdair/stdair_exceptions.hpp>
00011
00012 namespace TRADEMGEN {
00013
00014     // /////////// Exceptions ///////////
00018     class TrademgenGenerationException : public stdair::RootException {
00019     public:
00023         TrademgenGenerationException (const std::string& iWhat)
00024             : stdair::RootException (iWhat) {}
00025     };
00026
00030     class DemandInputFileNotFoundException
00031     : public stdair::FileNotFoundException {
00032     public:
00036         DemandInputFileNotFoundException (const std::string& iWhat)
00037             : stdair::FileNotFoundException (iWhat) {}
00038     };
00039
00043     class IndexOutOfRangeException : public TrademgenGenerationException {
00044     public:
00048         IndexOutOfRangeException (const std::string& iWhat)
00049             : TrademgenGenerationException (iWhat) {}
00050     };
00051
00052 }
00053 #endif // __TRADEMGEN_TRADEMGEN_EXCEPTIONS_HPP
00054
```

25.111 trademgen/TRADEMGEN_Service.hpp File Reference

```
#include <stdair/stdair_basic_types.hpp>
#include <stdair/stdair_demand_types.hpp>
#include <stdair/stdair_maths_types.hpp>
#include <stdair/stdair_json.hpp>
#include <stdair/stdair_service_types.hpp>
#include <stdair/basic/DemandGenerationMethod.hpp>
#include <stdair/bom/BookingRequestTypes.hpp>
#include <stdair/bom/EventTypes.hpp>
#include <stdair/bom/EventStruct.hpp>
#include <sevmgr/SEVMGR_Types.hpp>
#include <trademgen/TRADEMGEN_Types.hpp>
```

Classes

- class [TRADEMGEN::TRADEMGEN_Service](#)
class holding the services related to Travel Demand Generation.

Namespaces

- namespace [stdair](#)
Forward declarations.
- namespace [TRADEMGEN](#)

25.112 TRADEMGEN_Service.hpp

```

00001 #ifndef __TRADEMGEN_TRADEMGEN_SERVICE_HPP
00002 #define __TRADEMGEN_TRADEMGEN_SERVICE_HPP
00003
00004 // //////////////////////////////////////
00005 // Import section
00006 // //////////////////////////////////////
00007 // StdAir
00008 #include <stdair/stdair_basic_types.hpp>
00009 #include <stdair/stdair_demand_types.hpp>
00010 #include <stdair/stdair_maths_types.hpp>
00011 #include <stdair/stdair_json.hpp>
00012 #include <stdair/stdair_service_types.hpp>
00013 #include <stdair/basic/DemandGenerationMethod.hpp>
00014 #include <stdair/bom/BookingRequestTypes.hpp>
00015 #include <stdair/bom/EventTypes.hpp>
00016 #include <stdair/bom/EventStruct.hpp>
00017 // SEvMgr
00018 #include <sevmgr/SEVMGR_Types.hpp>
00019 // TraDemGen
00020 #include <trademgen/TRADEMGEN_Types.hpp>
00021
00022 // Forward declarations
00023 namespace stdair {
00024     class BomRoot;
00025     struct ProgressStatusSet;
00026     struct BasLogParams;
00027     struct BasDBParams;
00028     struct BookingRequestStruct;
00029     struct DemandCharacteristics;
00030     struct DemandDistribution;
00031     struct EventStruct;
00032     struct TravelSolutionStruct;
00033 }
00034
00035 namespace TRADEMGEN {
00036
00037     class TRADEMGEN_ServiceContext;
00038     struct DemandStreamKey;
00039
00040     class TRADEMGEN_Service {
00041     public:
00042         // ////////////////////////////////// Constructors and Destructors //////////////////////////////////
00043         TRADEMGEN_Service (const stdair::BasLogParams&, const stdair::BasDBParams&,
00044             const stdair::RandomSeed_T&);
00045
00046         TRADEMGEN_Service (const stdair::BasLogParams&, const stdair::RandomSeed_T&);
00047
00048         TRADEMGEN_Service (stdair::STDAIR_ServicePtr_T,
00049             SEVMGR::SEVMGR_ServicePtr_T,
00050             const stdair::RandomSeed_T&);
00051
00052         void parseAndLoad (const DemandFilePath&);
00053
00054         ~TRADEMGEN_Service ();
00055
00056     public:
00057         // ////////////////////////////////// Business support methods //////////////////////////////////
00058         void buildSampleBom ();
00059
00060         void clonePersistentBom ();
00061
00062         void buildComplementaryLinks (stdair::BomRoot&);
00063
00064     };
00065
00066 }
00067
00068 #endif

```

```

00241     stdair::BookingRequestStruct
00242     buildSampleBookingRequest (const bool isForCRS = false);
00243
00244     void displayAirlineListFromDB() const;
00248
00264     const stdair::Count_T& getExpectedTotalNumberOfRequestsToBeGenerated() const;
00265
00280     const stdair::Count_T& getActualTotalNumberOfRequestsToBeGenerated() const;
00281
00296     const bool
00297     stillHavingRequestsToBeGenerated (const stdair::DemandStreamKeyStr_T&,
00298                                     stdair::ProgressStatusSet&,
00299                                     const stdair::DemandGenerationMethod&) cons
00299     t;
00300
00313     stdair::Count_T
00314     generateFirstRequests (const stdair::DemandGenerationMethod&) const;
00315
00330     stdair::BookingRequestPtr_T
00331     generateNextRequest (const stdair::DemandStreamKeyStr_T&,
00332                         const stdair::DemandGenerationMethod&) const;
00333
00341     bool hasDemandStream (const stdair::DemandStreamKeyStr_T&) const;
00342
00359     stdair::ProgressStatusSet popEvent (stdair::EventStruct&) const;
00360
00369     bool isQueueDone() const;
00370
00374     bool generateCancellation (const stdair::TravelSolutionStruct&,
00375                               const stdair::PartySize_T&,
00376                               const stdair::DateTime_T&,
00377                               const stdair::Date_T&) const;
00378
00383     void reset() const;
00384
00388     const stdair::ProgressStatus& getProgressStatus () const;
00389
00394     const stdair::ProgressStatus& getProgressStatus (const stdair::EventType::EN_
00394     EventType&) const;
00395
00396     public:
00397         // ////////////////////////////////// Export support methods //////////////////////////////////
00407         std::string jsonHandler (const stdair::JSONString&) const;
00408
00409     public:
00410         // ////////////////////////////////// Display support methods //////////////////////////////////
00418         std::string csvDisplay() const;
00419
00426         std::string list () const;
00427
00434         std::string list (const stdair::EventType::EN_EventType&) const;
00435
00442         std::string displayDemandStream () const;
00443
00444
00445     private:
00446         // ////////////////////////////////// Constructors and Destructors //////////////////////////////////
00450         TRADEMGEN_Service();
00451
00455         TRADEMGEN_Service (const TRADEMGEN_Service&);
00456
00468         stdair::STDAIR_ServicePtr_T initStdAirService (const stdair::BasLogParams&,
00469                                                         const stdair::BasDBParams&);
00470
00480         stdair::STDAIR_ServicePtr_T initStdAirService (const stdair::BasLogParams&);
00481

```

```
00485     void initSEVMGRService();
00486
00495     void addStdAirService (stdair::STDAIR_ServicePtr_T ioSTDAIR_ServicePtr,
00496                           const bool iOwnStdairService);
00497
00503     void addSEVMGRService (SEVMGR::SEVMGR_ServicePtr_T ioSEVMGR_ServicePtr);
00504
00511     void initServiceContext (const stdair::RandomSeed_T&);
00512
00519     void initTrademgenService();
00520
00524     void finalise();
00525
00526
00527 private:
00528     // ////////// Service Context //////////
00532     TRADEMGEN_ServiceContext* _trademgenServiceContext;
00533 };
00534
00535 }
00536 #endif // __TRADEMGEN_TRADEMGEN_SERVICE_HPP
```

25.113 trademgen/TRADEMGEN_Types.hpp File Reference

```
#include <boost/shared_ptr.hpp>
#include <stdair/stdair_file.hpp>
#include <trademgen/TRADEMGEN_Exceptions.hpp>
```

Classes

- class [TRADEMGEN::DemandFilePath](#)

Namespaces

- namespace [TRADEMGEN](#)

Typedefs

- typedef boost::shared_ptr< TRADEMGEN_Service > [TRADEMGEN::TRADEMGEN_ServicePtr_T](#)

25.114 TRADEMGEN_Types.hpp

```
00001 #ifndef __TRADEMGENTRADEMGENTYPES_HPP
00002 #define __TRADEMGENTRADEMGENTYPES_HPP
00003
00004 // //////////////////////////////////////
00005 // Import section
00006 // //////////////////////////////////////
00007 // Boost
00008 #include <boost/shared_ptr.hpp>
00009 // StdAir
00010 #include <stdair/stdair_file.hpp>
00011 // TraDemGen
00012 #include <trademgen/TRADEMGEN_Exceptions.hpp>
00013
00014 namespace TRADEMGEN {
00015
00016     // Forward declarations
00017     class TRADEMGEN_Service;
00018
00019
00020     // ////////// Type definitions specific to DSim //////////
00024     typedef boost::shared_ptr<TRADEMGEN_Service> TRADEMGEN_ServicePtr_T;
00025
00026     // ////////// Files //////////
00030     class DemandFilePath : public stdair::InputFilePath {
00031     public:
00035         explicit DemandFilePath (const stdair::Filename_T& iFilename)
00036             : stdair::InputFilePath (iFilename) {}
00037     };
00038
00039 }
00040 #endif // __TRADEMGENTRADEMGENTYPES_HPP
00041
```

25.115 trademgen/ui/cmdline/trademgen.cpp File Reference

25.116 trademgen.cpp

```

00001
00005 // STL
00006 #include <cassert>
00007 #include <iostream>
00008 #include <sstream>
00009 #include <fstream>
00010 #include <string>
00011 // Boost (Extended STL)
00012 #include <boost/program_options.hpp>
00013 #include <boost/tokenizer.hpp>
00014 #include <boost/regex.hpp>
00015 #include <boost/swap.hpp>
00016 #include <boost/algorithm/string/case_conv.hpp>
00017 // GNU Readline Wrapper
00018 #include <stdair/ui/cmdline/SReadline.hpp>
00019 // StdAir
00020 #include <stdair/stdair_basic_types.hpp>
00021 #include <stdair/stdair_json.hpp>
00022 #include <stdair/basic/BasConst_General.hpp>
00023 #include <stdair/basic/ProgressStatusSet.hpp>
00024 #include <stdair/basic/DemandGenerationMethod.hpp>
00025 #include <stdair/bom/EventStruct.hpp>
00026 #include <stdair/bom/BookingRequestStruct.hpp>
00027 #include <stdair/bom/BomDisplay.hpp>
00028 #include <stdair/service/Logger.hpp>
00029 // TraDemGen
00030 #include <trademgen/TRADEMGEN_Service.hpp>
00031 #include <trademgen/config/trademgen-paths.hpp>
00032
00033
00034 // ////////// Type definitions //////////
00035 typedef std::vector<std::string> WordList_T;
00036
00037
00038 // ////////// Specific type definitions //////////
00039 typedef unsigned int NbofRuns_T;
00040
00041
00042 // ////////// Constants //////////
00043
00044 const stdair::Filename_T K_TRADEMGEN_DEFAULT_LOG_FILENAME ("trademgen.log");
00045
00046
00047 const stdair::Filename_T K_TRADEMGEN_DEFAULT_INPUT_FILENAME (STDAIR_SAMPLE_DIR
00048                                                                "/rds01/demand05.csv
00049 ");
00050
00051
00052 const stdair::DemandGenerationMethod
00053 K_TRADEMGEN_DEFAULT_DEMAND_GENERATION_METHOD =
00054     stdair::DemandGenerationMethod::POI_PRO;
00055
00056
00057 const char K_TRADEMGEN_DEFAULT_DEMAND_GENERATION_METHOD_CHAR =
00058     K_TRADEMGEN_DEFAULT_DEMAND_GENERATION_METHOD.getMethodAsChar();
00059
00060
00061 const stdair::RandomSeed_T K_TRADEMGEN_DEFAULT_RANDOM_SEED =
00062     stdair::DEFAULT_RANDOM_SEED;
00063
00064
00065 const NbofRuns_T K_TRADEMGEN_DEFAULT_RANDOM_DRAWS = 1;
00066
00067
00068 const bool K_TRADEMGEN_DEFAULT_BUILT_IN_INPUT = false;
00069
00070
00071 const int K_TRADEMGEN_EARLY_RETURN_STATUS = 99;
00072
00073
00074 // //////////////////////////////////////
00075 typedef std::vector<std::string> TokenList_T;
00076
00077
00078 struct Command_T {
00079     typedef enum {
00080         NOP = 0,

```

```

00103     QUIT,
00104     HELP,
00105     LIST_EVENT,
00106     LIST_DEMAND_STREAM,
00107     RESET,
00108     NEXT,
00109     GENERATE_NEXT_BR,
00110     GENERATE_FIRST_BR,
00111     GENERATE_ALL_BR,
00112     JSON_LIST,
00113     LAST_VALUE
00114 } Type_T;
00115 };
00116
00117 // //////////////////////////////////////
00118 void tokeniseStringIntoWordList (const std::string& iPhrase,
00119                                 WordList_T& ioWordList) {
00120     // Empty the word list
00121     ioWordList.clear();
00122
00123     // Boost Tokeniser
00124     typedef boost::tokenizer<boost::char_separator<char> > Tokeniser_T;
00125
00126     // Define the separators
00127     const boost::char_separator<char> lSeparatorList(" .,:|+-*/_= !@#$$%^&(){}[]?'<>
00128 \");
00129
00130     // Initialise the phrase to be tokenised
00131     Tokeniser_T lTokens(iPhrase, lSeparatorList);
00132     for (Tokeniser_T::const_iterator tok_iter = lTokens.begin();
00133          tok_iter != lTokens.end(); ++tok_iter) {
00134         const std::string& lTerm = *tok_iter;
00135         ioWordList.push_back(lTerm);
00136     }
00137 } // //////////////////////////////////////
00138 std::string createStringFromWordList (const WordList_T& iWordList) {
00139     std::ostream oStr;
00140
00141     unsigned short idx = iWordList.size();
00142     for (WordList_T::const_iterator itWord = iWordList.begin();
00143          itWord != iWordList.end(); ++itWord, --idx) {
00144         const std::string& lWord = *itWord;
00145         oStr << lWord;
00146         if (idx > 1) {
00147             oStr << " ";
00148         }
00149     }
00150
00151     return oStr.str();
00152 }
00153
00154
00155 // /////////// Parsing of Options & Configuration ///////////
00156 // A helper function to simplify the main part.
00157 template<class T> std::ostream& operator<< (std::ostream& os,
00158                                             const std::vector<T>& v) {
00159     std::copy(v.begin(), v.end(), std::ostream_iterator<T>(std::cout, " "));
00160     return os;
00161 }
00162
00163 int readConfiguration (int argc, char* argv[], bool& ioIsBuiltin,
00164                       stdair::RandomSeed_T& ioRandomSeed,
00165                       stdair::Filename_T& ioInputFilename,
00166                       stdair::Filename_T& ioOutputFilename,
00167                       stdair::Filename_T& ioLogFilename,
00168                       stdair::DemandGenerationMethod& ioDemandGenerationMethod)
00169 {

```

```

00171
00172 // Demand generation method as a single char (e.g., 'P' or 'S').
00173 char lDemandGenerationMethodChar;
00174
00175 // Default for the built-in input
00176 ioIsBuiltin = K_TRADEMGEN_DEFAULT_BUILT_IN_INPUT;
00177
00178 // Declare a group of options that will be allowed only on command line
00179 boost::program_options::options_description generic ("Generic options");
00180 generic.add_options()
00181     ("prefix,p", "print installation prefix")
00182     ("version,v", "print version string")
00183     ("help,h", "produce help message");
00184
00185 // Declare a group of options that will be allowed both on command
00186 // line and in config file
00187 boost::program_options::options_description config ("Configuration");
00188 config.add_options()
00189     ("builtin,b",
00190      "The sample BOM tree can be either built-in or parsed from an input file. Th
00191      at latter must then be given with the -i/--input option")
00192     ("seed,s",
00193      boost::program_options::value<std::random::RandomSeed_T>(&ioRandomSeed)->default_
00194      value(K_TRADEMGEN_DEFAULT_RANDOM_SEED),
00195      "Seed for the random generation")
00196     ("demandgeneration,G",
00197      boost::program_options::value< char >(&lDemandGenerationMethodChar)->default_
00198      _value(K_TRADEMGEN_DEFAULT_DEMAND_GENERATION_METHOD_CHAR),
00199      "Method used to generate the demand (i.e., the booking requests): Poisson Pr
00200      ocess (P) or Order Statistics (S)")
00201     ("input,i",
00202      boost::program_options::value< std::string >(&ioInputFilename)->default_valu
00203      e(K_TRADEMGEN_DEFAULT_INPUT_FILENAME),
00204      "(CSV) input file for the demand distributions")
00205     ("log,l",
00206      boost::program_options::value< std::string >(&ioLogFilename)->default_value(
00207      K_TRADEMGEN_DEFAULT_LOG_FILENAME),
00208      "Filepath for the logs")
00209     ;
00210
00211 // Hidden options, will be allowed both on command line and
00212 // in config file, but will not be shown to the user.
00213 boost::program_options::options_description hidden ("Hidden options");
00214 hidden.add_options()
00215     ("copyright",
00216      boost::program_options::value< std::vector<std::string> >(),
00217      "Show the copyright (license)");
00218
00219 boost::program_options::options_description cmdline_options;
00220 cmdline_options.add(generic).add(config).add(hidden);
00221
00222 boost::program_options::options_description config_file_options;
00223 config_file_options.add(config).add(hidden);
00224
00225 boost::program_options::options_description visible ("Allowed options");
00226 visible.add(generic).add(config);
00227
00228 boost::program_options::positional_options_description p;
00229 p.add ("copyright", -1);
00230
00231 boost::program_options::variables_map vm;
00232 boost::program_options::
00233     store (boost::program_options::command_line_parser (argc, argv).
00234             options (cmdline_options).positional(p).run(), vm);
00235
00236 std::ifstream ifs ("trademgen.cfg");
00237 boost::program_options::store (parse_config_file (ifs, config_file_options),

```

```

00232                                     vm);
00233 boost::program_options::notify (vm);
00234
00235 if (vm.count ("help")) {
00236     std::cout << visible << std::endl;
00237     return K_TRADEMGEN_EARLY_RETURN_STATUS;
00238 }
00239
00240 if (vm.count ("version")) {
00241     std::cout << PACKAGE_NAME << ", version " << PACKAGE_VERSION << std::endl;
00242     return K_TRADEMGEN_EARLY_RETURN_STATUS;
00243 }
00244
00245 if (vm.count ("prefix")) {
00246     std::cout << "Installation prefix: " << PREFIXDIR << std::endl;
00247     return K_TRADEMGEN_EARLY_RETURN_STATUS;
00248 }
00249
00250 if (vm.count ("builtin")) {
00251     ioIsBuiltin = true;
00252 }
00253 const std::string isBuiltinStr = (ioIsBuiltin == true)?"yes":"no";
00254 std::cout << "The BOM should be built-in? " << isBuiltinStr << std::endl;
00255
00256 if (ioIsBuiltin == false) {
00257
00258     // The BOM tree should be built from parsing a demand input file
00259     if (vm.count ("input")) {
00260         ioInputFilename = vm["input"].as< std::string >();
00261         std::cout << "Input filename is: " << ioInputFilename << std::endl;
00262
00263     } else {
00264         // The built-in option is not selected. However, no demand input file
00265         // is specified
00266         std::cerr << "Either one among the -b/--builtin and -i/--input "
00267             << "options must be specified" << std::endl;
00268     }
00269 }
00270
00271 if (vm.count ("output")) {
00272     ioOutputFilename = vm["output"].as< std::string >();
00273     std::cout << "Output filename is: " << ioOutputFilename << std::endl;
00274 }
00275
00276 if (vm.count ("log")) {
00277     ioLogFilename = vm["log"].as< std::string >();
00278     std::cout << "Log filename is: " << ioLogFilename << std::endl;
00279 }
00280
00281 if (vm.count ("demandgeneration")) {
00282     ioDemandGenerationMethod =
00283         stdair::DemandGenerationMethod (lDemandGenerationMethodChar);
00284     std::cout << "Date-time request generation method is: "
00285         << ioDemandGenerationMethod.describe() << std::endl;
00286 }
00287
00288 //
00289 std::cout << "The random generation seed is: " << ioRandomSeed << std::endl;
00290
00291 return 0;
00292 }
00293
00294 // //////////////////////////////////////
00295 void initReadline (swift::SReadline& ioInputReader) {
00296
00297     // Prepare the list of my own completers
00298     std::vector<std::string> Completers;

```

```

00299
00300 // The following is supported:
00301 // - "identifiers"
00302 // - special identifier %file - means to perform a file name completion
00303 Completers.push_back ("help");
00304 Completers.push_back ("list_event");
00305 Completers.push_back ("list_demand_stream");
00306 Completers.push_back ("reset");
00307 Completers.push_back ("generate_next_br");
00308 Completers.push_back ("generate_first_br");
00309 Completers.push_back ("generate_all_br");
00310 Completers.push_back ("next");
00311 Completers.push_back ("json_list");
00312 Completers.push_back ("quit");
00313
00314 // Now register the completers.
00315 // Actually it is possible to re-register another set at any time
00316 ioInputReader.RegisterCompletions (Completers);
00317 }
00318
00319 // //////////////////////////////////////
00320 Command_T::Type_T extractCommand (TokenList_T& ioTokenList) {
00321     Command_T::Type_T oCommandType = Command_T::LAST_VALUE;
00322
00323     // Interpret the user input
00324     if (ioTokenList.empty() == false) {
00325         TokenList_T::iterator itTok = ioTokenList.begin();
00326         std::string lCommand (*itTok);
00327         boost::algorithm::to_lower (lCommand);
00328
00329         if (lCommand == "help") {
00330             oCommandType = Command_T::HELP;
00331
00332         } else if (lCommand == "list_event") {
00333             oCommandType = Command_T::LIST_EVENT;
00334
00335         } else if (lCommand == "list_demand_stream") {
00336             oCommandType = Command_T::LIST_DEMAND_STREAM;
00337
00338         } else if (lCommand == "reset") {
00339             oCommandType = Command_T::RESET;
00340
00341         } else if (lCommand == "delete_first") {
00342             oCommandType = Command_T::NEXT;
00343
00344         } else if (lCommand == "generate_first_br") {
00345             oCommandType = Command_T::GENERATE_FIRST_BR;
00346
00347         } else if (lCommand == "generate_next_br") {
00348             oCommandType = Command_T::GENERATE_NEXT_BR;
00349
00350         } else if (lCommand == "generate_all_br") {
00351             oCommandType = Command_T::GENERATE_ALL_BR;
00352
00353         } else if (lCommand == "json_list") {
00354             oCommandType = Command_T::JSON_LIST;
00355
00356         } else if (lCommand == "quit") {
00357             oCommandType = Command_T::QUIT;
00358         }
00359
00360         // Remove the first token (the command), as the corresponding information
00361         // has been extracted in the form of the returned command type enumeration
00362         ioTokenList.erase (itTok);
00363
00364     } else {
00365         oCommandType = Command_T::NOP;

```

```

00366     }
00367
00368     return oCommandType;
00369 }
00370
00371 // //////////////////////////////////////
00372 std::string toString (const TokenList_T& iTokenList) {
00373     std::ostringstream oStr;
00374
00375     // Re-create the string with all the tokens, trimmed by read-line
00376     unsigned short idx = 0;
00377     for (TokenList_T::const_iterator itTok = iTokenList.begin();
00378          itTok != iTokenList.end(); ++itTok, ++idx) {
00379         if (idx != 0) {
00380             oStr << " ";
00381         }
00382         oStr << *itTok;
00383     }
00384
00385     return oStr.str();
00386 }
00387
00388 // ////////////////////////////////// M A I N //////////////////////////////////
00389 int main (int argc, char* argv[]) {
00390
00391     // Readline history
00392     const unsigned int lHistorySize (100);
00393     const std::string lHistoryFilename ("trademgen.hist");
00394     const std::string lHistoryBackupFilename ("trademgen.hist.bak");
00395
00396     // Default parameters for the interactive session
00397     stdair::EventStruct lCurrentInteractiveEventStruct;
00398     stdair::DateTime_T lCurrentInteractiveDateTime;
00399     std::string lDefaultDemandStreamKey;
00400
00401     // State whether the BOM tree should be built-in or parsed from an input file
00402     bool isBuiltin;
00403
00404     // Random generation seed
00405     stdair::RandomSeed_T lRandomSeed;
00406
00407     // Input file name
00408     stdair::Filename_T lInputFilename;
00409
00410     // Output file name
00411     stdair::Filename_T lOutputFilename;
00412
00413     // Output log File
00414     stdair::Filename_T lLogFilename;
00415
00416     // Demand generation method.
00417     stdair::DemandGenerationMethod
00418         lDemandGenerationMethod (K_TRADEMGEN_DEFAULT_DEMAND_GENERATION_METHOD);
00419
00420     // Call the command-line option parser
00421     const int lOptionParserStatus =
00422         readConfiguration (argc, argv, isBuiltin, lRandomSeed,
00423                          lInputFilename, lOutputFilename, lLogFilename,
00424                          lDemandGenerationMethod);
00425
00426     if (lOptionParserStatus == K_TRADEMGEN_EARLY_RETURN_STATUS) {
00427         return 0;
00428     }
00429
00430     // Set the log parameters
00431     std::ofstream logOutputFile;
00432     // Open and clean the log outputfile

```



```

00433 logOutputFile.open (lLogFilename.c_str());
00434 logOutputFile.clear();
00435
00436 // Set up the log parameters
00437 const stdair::BasLogParams lLogParams (stdair::LOG::DEBUG, logOutputFile);
00438
00439 // Initialise the TraDemGen service object
00440 TRADEMGEN::TRADEMGEN_Service trademgenService (lLogParams, lRandomSeed);
00441
00442 // Check whether or not a (CSV) input file should be read
00443 if (isBuiltin == true) {
00444     // Create a sample DemandStream object, and insert it within the BOM tree
00445     trademgenService.buildSampleBom();
00446     lDefaultDemandStreamKey = "SIN-BKK 2010-Feb-08 Y";
00447 } else {
00448     // Create the DemandStream objects, and insert them within the BOM tree
00449     const TRADEMGEN::DemandFilePath lDemandFilePath (lInputFilename);
00450     trademgenService.parseAndLoad (lDemandFilePath);
00451     lDefaultDemandStreamKey = "SIN-BKK 2009-Feb-09 Y";
00452 }
00453
00454 // DEBUG
00455 STDAIR_LOG_DEBUG ("=====");
00456 STDAIR_LOG_DEBUG ("=          Beginning of the interactive session          =");
00457 STDAIR_LOG_DEBUG ("=====");
00458
00459 // Initialise the GNU readline wrapper
00460 swift::SReadline lReader (lHistoryFilename, lHistorySize);
00461 initReadline (lReader);
00462
00463 // Now we can ask user for a line
00464 std::string lUserInput;
00465 bool EndOfInput (false);
00466 Command_T::Type_T lCommandType (Command_T::NOP);
00467
00468 while (lCommandType != Command_T::QUIT && EndOfInput == false) {
00469     // Update the interactive parameters which have not been updated yet
00470     //lCurrentInteractiveDateTime = lCurrentInteractiveEventStruct.getEventTime (
00471 );
00472     //lCurrentInteractiveEventType = lCurrentInteractiveEventStruct.getEventType
00473 ();
00474
00475     // Prompt
00476     std::ostream oPromptStr;
00477     oPromptStr << "trademgen " << "> ";
00478     // << stdair::EventType::getTypeLabelAsString(lCurrentInteractiveEventType)
00479
00480     // << " / " << lCurrentInteractiveDateTime << "> ";
00481     // Call read-line, which will fill the list of tokens
00482     TokenList_T lTokenListByReadline;
00483     lUserInput = lReader.GetLine (oPromptStr.str(), lTokenListByReadline,
00484                                     EndOfInput);
00485
00486     // The history can be saved to an arbitrary file at any time
00487     lReader.SaveHistory (lHistoryBackupFilename);
00488
00489     // The end-of-input typically corresponds to a CTRL-D typed by the user
00490     if (EndOfInput) {
00491         std::cout << std::endl;
00492         break;
00493     }
00494
00495     // Interpret the user input
00496     lCommandType = extractCommand (lTokenListByReadline);

```

```

00497     switch (lCommandType) {
00498
00499         // /////////////////////////////////// Help ///////////////////////////////////
00500     case Command_T::HELP: {
00501         std::cout << std::endl;
00502         std::cout << "Commands: " << std::endl;
00503         std::cout << " help" << "\t\t" << "Display this help" << std::endl;
00504         std::cout << " quit" << "\t\t" << "Quit the application" << std::endl;
00505         std::cout << " list_event" << "\t\t"
00506             << "List all the events in the queue" << std::endl;
00507         std::cout << " list_demand_stream" << "\t\t"
00508             << "List the streams used to generate demand" << std::endl;
00509         std::cout << " reset" << "\t\t" << "Reset the service (including the "
00510             << "event queue)" << std::endl;
00511         std::cout << " generate_first_br" << "\t\t" << "Generate the first booking "
00512             << "request for each demand stream and add it to the event queue"
00513
00514             << std::endl;
00515         std::cout << " generate_next_br" << "\t\t" << "Generate the next event for "
00516             << "the specified demand stream and add it to the event queue"
00517             << "\n\t\t\t\t\tFor instance:"
00518             << "\n\t\t\t\t\t 'generate_next_br " << lDefaultDemandStreamKey
00519             << "' " << std::endl;
00520         std::cout << " generate_all_br" << "\t\t" << "Generate all the events for "
00521             << "the specified demand stream and add it to the event queue"
00522             << "\n\t\t\t\t\tFor instance:"
00523             << "\n\t\t\t\t\t 'generate_all_br " << lDefaultDemandStreamKey
00524             << "' " << std::endl;
00525         std::cout << " delete_first" << "\t\t"
00526             << "Pop the next event from the queue"
00527             << std::endl;
00528         std::cout << " \nDebug Commands" << std::endl;
00529         std::cout << " json_list" << "\t\t"
00530             << "List events in the queue in a JSON format"
00531             << std::endl;
00532         std::cout << std::endl;
00533     }
00534
00535     // /////////////////////////////////// Quit ///////////////////////////////////
00536     case Command_T::QUIT: {
00537         break;
00538     }
00539
00540     // /////////////////////////////////// List ///////////////////////////////////
00541     case Command_T::LIST_EVENT: {
00542         //
00543         std::cout << "List of events" << std::endl;
00544
00545         std::ostringstream oEventListStr;
00546         oEventListStr << trademgenService.list ();
00547         std::cout << oEventListStr.str() << std::endl;
00548         STDAIR_LOG_DEBUG (oEventListStr.str());
00549
00550         //
00551         break;
00552     }
00553
00554     // /////////////////////////////////// List ///////////////////////////////////
00555     case Command_T::LIST_DEMAND_STREAM: {
00556         //
00557         std::cout << "List of demand streams" << std::endl;
00558
00559         std::ostringstream oEventListStr;
00560         oEventListStr << trademgenService.displayDemandStream ();
00561         std::cout << oEventListStr.str() << std::endl;
00562         STDAIR_LOG_DEBUG (oEventListStr.str());

```

```

00563
00564     //
00565     break;
00566 }
00567
00568     // ////////////////////////////////// Reset //////////////////////////////////
00569 case Command_T::RESET: {
00570
00571     std::cout << "Reset" << std::endl;
00572
00573     // Reset the service (including the event queue) for the next run
00574     trademgenService.reset();
00575
00576     break;
00577 }
00578
00579     // ////////////////////////////////// Generate next request //////////////////////////////////
00580     //////////////////////////////////////////////////
00581 case Command_T::GENERATE_NEXT_BR: {
00582
00583     // Retrieve the corresponding demand stream key
00584     const stdair::DemandGeneratorKey_T lDemandStreamKey =
00585         toString(lTokenListByReadline);
00586
00587     // Check that such demand stream exists
00588     const bool hasDemandStream =
00589         trademgenService.hasDemandStream(lDemandStreamKey);
00590
00591     if (hasDemandStream == false) {
00592         // DEBUG
00593         std::ostringstream oNoDemandStreamStr;
00594         oNoDemandStreamStr << "Wrong demand stream key: '"
00595             << lDemandStreamKey << "'."
00596             << "\nExisting demand streams are:\n"
00597             << trademgenService.displayDemandStream();
00598         std::cout << oNoDemandStreamStr.str() << std::endl;
00599         STDAIR_LOG_DEBUG (oNoDemandStreamStr.str());
00600         break;
00601     }
00602     assert (hasDemandStream == true);
00603
00604     stdair::ProgressStatusSet lProgressStatusSet (stdair::EventType::BKG_REQ);
00605     const bool stillHavingRequestsToBeGenerated =
00606         trademgenService.stillHavingRequestsToBeGenerated (lDemandStreamKey,
00607             lProgressStatusSet,
00608             lDemandGenerationMetho
00609 d);
00610
00611     if (stillHavingRequestsToBeGenerated == false) {
00612         // DEBUG
00613         std::ostringstream oNoMoreEventToGenerateStr;
00614         oNoMoreEventToGenerateStr << "No more events to generate for the demand "
00615             << "stream: '" << lDemandStreamKey << "'.";
00616         std::cout << oNoMoreEventToGenerateStr.str() << std::endl;
00617         STDAIR_LOG_DEBUG (oNoMoreEventToGenerateStr.str());
00618         break;
00619     }
00620     assert (stillHavingRequestsToBeGenerated == true);
00621
00622     trademgenService.generateNextRequest (lDemandStreamKey, lDemandGenerationMe
00623 thod);
00624
00625     // DEBUG
00626     std::ostringstream oOneMoreEventGeneratedStr;
00627     oOneMoreEventGeneratedStr << "One more event have been generated for the de
00628 mand "
00629     << "stream: '" << lDemandStreamKey << "'.";

```

```

00625         std::cout << oOneMoreEventGeneratedStr.str() << std::endl;
00626         STDAIR_LOG_DEBUG (oOneMoreEventGeneratedStr.str());
00627
00628         break;
00629     }
00630
00631     // ////////////////////////////////// Generate first requests //////////////////////////////////
00632     // //////////////////////////////////
00633     case Command_T::GENERATE_FIRST_BR: {
00634         std::cout << "Generate first requests" << std::endl;
00635
00636         // Generate the first event for each demand stream.
00637         trademgenService.generateFirstRequests (lDemandGenerationMethod);
00638
00639         break;
00640     }
00641
00642     // ////////////////////////////////// Generate all requests //////////////////////////////////
00643     // //////////////////////////////////
00644     case Command_T::GENERATE_ALL_BR: {
00645         // Retrieve the corresponding demand stream key
00646         const stdair::DemandGeneratorKey_T lDemandStreamKey =
00647             toString(lTokenListByReadline);
00648
00649         // Check that such demand stream exists
00650         const bool hasDemandStream =
00651             trademgenService.hasDemandStream(lDemandStreamKey);
00652
00653         if (hasDemandStream == false) {
00654             // DEBUG
00655             std::ostringstream oNoDemandStreamStr;
00656             oNoDemandStreamStr << "Wrong demand stream key: '"
00657                 << lDemandStreamKey << "'."
00658                 << "\nExisting demand streams are:\n"
00659                 << trademgenService.displayDemandStream();
00660             std::cout << oNoDemandStreamStr.str() << std::endl;
00661             STDAIR_LOG_DEBUG (oNoDemandStreamStr.str());
00662             break;
00663         }
00664         assert (hasDemandStream == true);
00665
00666         stdair::ProgressStatusSet lProgressStatusSet (stdair::EventType::BKG_REQ);
00667         bool stillHavingRequestsToBeGenerated =
00668             trademgenService.stillHavingRequestsToBeGenerated (lDemandStreamKey,
00669                 lProgressStatusSet,
00670                 lDemandGenerationMetho
00671 d);
00672
00673         if (stillHavingRequestsToBeGenerated == false) {
00674             // DEBUG
00675             std::ostringstream oNoMoreEventToGenerateStr;
00676             oNoMoreEventToGenerateStr << "No more events to generate for the demand "
00677
00678                 << "stream: '" << lDemandStreamKey << "'.";
00679             std::cout << oNoMoreEventToGenerateStr.str() << std::endl;
00680             STDAIR_LOG_DEBUG (oNoMoreEventToGenerateStr.str());
00681             break;
00682         }
00683         assert (stillHavingRequestsToBeGenerated == true);
00684
00685         stdair::Count_T lNumberOfRequests = 0;
00686         while (stillHavingRequestsToBeGenerated == true) {
00687             lNumberOfRequests++;
00688             trademgenService.generateNextRequest (lDemandStreamKey, lDemandGeneration

```

```

Method);
00688         stillHavingRequestsToBeGenerated =
00689         trademgenService.stillHavingRequestsToBeGenerated (lDemandStreamKey,
00690                                                             lProgressStatusSet,
00691                                                             lDemandGenerationMet
hod);
00692
00693
00694     }
00695     // DEBUG
00696     std::ostringstream oOneMoreEventGeneratedStr;
00697     oOneMoreEventGeneratedStr << lNumberOfRequests
00698     << " more event(s) have been generated for the de
mand "
00699     << "stream: '" << lDemandStreamKey << "'.";
00700     std::cout << oOneMoreEventGeneratedStr.str() << std::endl;
00701     STDAIR_LOG_DEBUG (oOneMoreEventGeneratedStr.str());
00702
00703     break;
00704 }
00705
00706 // ////////////////////////////////// Next //////////////////////////////////
00707 case Command_T::NEXT: {
00708     //
00709     std::cout << "Next" << std::endl;
00710
00711     if (trademgenService.isQueueDone() == true) {
00712
00713         // DEBUG
00714         std::ostringstream oEmptyQueueStr;
00715         oEmptyQueueStr << "The event queue is empty: no event can be popped out."
;
00716         std::cout << oEmptyQueueStr.str() << std::endl;
00717         STDAIR_LOG_DEBUG (oEmptyQueueStr.str());
00718
00719         //
00720         break;
00721
00722     }
00723
00724     // Get the next event from the event queue
00725     trademgenService.popEvent (lCurrentInteractiveEventStruct);
00726
00727     // DEBUG
00728     std::ostringstream oEventStr;
00729     oEventStr << "Poped event: '"
00730     << lCurrentInteractiveEventStruct.describe() << "'.";
00731     std::cout << oEventStr.str() << std::endl;
00732     STDAIR_LOG_DEBUG (oEventStr.str());
00733
00734     //
00735     break;
00736 }
00737 // ////////////////////////////////// JSon Event List //////////////////////////////////
00738
00739 case Command_T::JSON_LIST: {
00740     //
00741     std::cout << "JSON Event List" << std::endl;
00742
00743     std::ostringstream lMyCommandJSONstream;
00744     lMyCommandJSONstream << "{\"event_list\":"
00745     << "{ \"event_type\":" << "all"
00746     << "\"}"}";
00747
00748     // Delegate the call to the dedicated service
00749     const stdair::JSONString lJSONCommandString (lMyCommandJSONstream.str());
00750     const std::string& lCSVEventListDump =

```

```
00751         trademgenService.jsonHandler (lJSONCommandString);
00752
00753         // DEBUG: Display the events queue JSON string
00754         std::cout << lCSVEventListDump << std::endl;
00755         STDAIR_LOG_DEBUG (lCSVEventListDump);
00756
00757         break;
00758     }
00759
00760     // ////////////////////////////////// Default / No value //////////////////////////////////
00761     case Command_T::NOP: {
00762         break;
00763     }
00764
00765     case Command_T::LAST_VALUE:
00766     default: {
00767         // DEBUG
00768         std::ostream oStr;
00769         oStr << "That command is not yet understood: '" << lUserInput
00770             << "' => " << lTokenListByReadline;
00771         STDAIR_LOG_DEBUG (oStr.str());
00772         std::cout << oStr.str() << std::endl;
00773     }
00774 }
00775 }
00776
00777 // DEBUG
00778 STDAIR_LOG_DEBUG ("End of the session. Exiting.");
00779 std::cout << "End of the session. Exiting." << std::endl;
00780
00781 // Close the Log outputFile
00782 logOutputFile.close();
00783
00784 /*
00785     Note: as that program is not intended to be run on a server in
00786     production, it is better not to catch the exceptions. When it
00787     happens (that an exception is throwned), that way we get the
00788     call stack.
00789 */
00790
00791 return 0;
00792 }
```