TRANSCORE

Software Design Description

For A

Center-To-Center (C2C) Reference Implementation

Contract No. DTFH61-16-D-00051 T-0044

Final

Prepared for:

Federal Highway Administration
1200 New Jersey Ave., S.E.
Washington, DC 20590-9898

Prepared by:

TransCore
157 Technology Parkway, Suite 900
Peachtree Corners, Georgia 30092

December 1, 2021

REVISION HISTORY				
DATE	REV.	SPCR(S)	SECTION(S)	DESCRIPTION
05/18/2008				Initial Version
02/15/2009				Updated Version for SDD Walkthrough
10/3/2013				Updated for Release 3
10/26/2015				Updated to include Entity Emulation and Section1201 reporting and comments from Release 3.
01/17/2016				Updated to include changes identified during the design walkthrough
04/19/2016				Updated based on test procedure review comments.
07/14/2016	1.1	CTCRI-212, CTCRI-224, CTCRI-257, CTCRI-258		Updated to address items raised during Acceptance Test 1
11/23/2016	1.2	CTCRI-677		Edit SRS50.10; Add SRS49.3
05/25/2017	1.3			Final for Release 2
1/21/2019	1.4			Updated for Contract No. DTFH61-16-D-00051 T-0007
6/1/2021	1.5		5, 7	Updated for C2C RI Enhancements including: Removed the tables in Section 5 where the information is not directly traced to SRS requirements in the requirements traceability matrix. In future versions of the SDD, we envision replacing this section with more understandable design content similar to sections 6 and 7. Added Section 7 to include design information for the new Enhancements.

12/1/2021	1.6			For easier understanding and utilization, the organization of the Detailed Design section has been updated to to better follow the High Level Design section. Removed references to classes that weren't traced to requirements and the descriptions only referenced the source code. Merged the previous Section 7 Enhancements section into the appropriate parts of the High Level Design section.

TRANSCORE

TABLE OF CONTENTS

1 INTRODUCTION 1-1

1.1 IDENTIFICATION 1-1

1.2 PURPOSE 1-1

1.3 SCOPE 1-1

1.4 DEFINITIONS AND ACRONYMS 1-2

1.5 OVERVIEW 1-3

2 REFERENCES 2-1

3 RI SYSTEM 3-1

3.1 DESCRIPTION 3-1

3.2 DEPLOYMENT DIAGRAM 3-1

4 HIGH LEVEL ARCHITECTURE 4-1

4.1 BASIC RI OPERATION 4-1

4.2 ARCHITECTURAL COMPONENTS 4-2

4.2.1 GUI 4-3

4.2.2 Test Model 4-4

4.2.2.1 Test Control 4-5

4.2.2.2 Test Configuration 4-5

4.2.3 Center Model 4-6

4.2.3.1 Entity (devices) 4-6

4.2.3.2 Behavioral 4-7

4.2.4 Message Manager 4-8

4.2.5 Logger 4-9

4.2.6 Reporter 4-10

4.2.7 C2C SUT Interface 4-11

4.2.7.1 Center Services 4-11

4.2.7.2 Client Services 4-12

4.2.7.3 XML/SOAP Support 4-12

4.2.7.4 DATEX-ASN Support (Deprecated) 4-12

4.2.7.5 UDP/IP 4-12

4.2.7.6 TCP/IP 4-12

4.3 RI INTERFACES 4-14

5 DETAILED DESIGN 5-1

5.1 GUI 5-1

5.1.1 Main UI 5-1

5.1.1.1 Primary Presentation 5-1

5.1.1.2 Element Catalog 5-1

5.1.1.2.1 C2CMainUI 5-1

5.1.1.2.2 RIUICoordinator 5-6

5.1.1.2.3 BasicGUIActionWrapper 5-9

5.1.1.2.4 ProgressUI 5-11

5.1.1.2.5 HelpUI 5-13

5.1.1.2.6 OptionsUI 5-14

5.1.1.3 Variability Guide 5-17

5.1.1.4 Rationale 5-17

5.1.1.5 Context Diagram 5-17

5.1.2 Test Definition GUI 5-18

5.1.2.1 Primary Presentation 5-18

5.1.2.2 Element Catalog 5-18

5.1.2.2.1 ConfigurationBasePanel 5-18
5.1.2.2.2 AppLayerPanel 5-22
5.1.2.2.3 AppLayerTestCasePanel 5-24
5.1.2.2.4 InfoLayerPanel 5-25
5.1.2.2.5 InfoLayerTestCasePanel 5-28
5.1.2.2.6 ConfigurationPanel 5-29
5.1.2.2.7 EntityEmulationPanel 5-34
5.1.2.2.8 EntityDataViewerUI 5-36
5.1.2.2.9 SUTPanel 5-37
5.1.2.2.10 TestConfigCreateAction 5-40
5.1.2.2.11 TestConfigCreateUI 5-41
5.1.2.2.12 Test Case Editor 5-49
5.1.2.2.12.1 Primary Presentation 5-49
5.1.2.2.12.2 Element Catalog 5-50
5.1.2.2.12.2.1 TestCaseEditorDialog 5-51
5.1.2.2.12.2.2 TestCaseDataEditor 5-51
5.1.2.2.12.2.3 TestCaseFile 5-52
5.1.2.2.12.2.4 EditorModified 5-52
5.1.2.2.12.3 Context Diagram 5-52
5.1.2.2.12.4 Variability Guide 5-52
5.1.2.2.12.5 Rationale 5-52
5.1.2.2.13 C2C RI Wizard 5-52
5.1.2.2.13.1 Primary Presentation 5-52
5.1.2.2.13.2 Element Catalog 5-53
5.1.2.2.13.2.1 C2CRIWizardPanel 5-53
5.1.2.2.13.2.2 C2CRIWizardDialog 5-53
5.1.2.2.13.2.3 C2CRIWizardPageProvider 5-53
5.1.2.2.13.2.4 C2CRIWizardFactory 5-54
5.1.2.2.13.2.5 TestConfigCreateFactory 5-54
5.1.2.2.13.2.6 C2CRIWizardPage 5-54
5.1.2.2.13.2.7 CreateTestMainPage 5-54
5.1.2.2.13.2.8 EditOrFinishPage 5-55
5.1.2.2.13.2.9 TestConfigCreatePage 5-56
5.1.2.2.13.2.10 C2CRITestModePage 5-57
5.1.2.2.13.2.11 ConfigureEmulationDataPage 5-58
5.1.2.2.13.2.12 SelectNeedsPage 5-58
5.1.2.2.13.2.13 SelectRequirementsPage 5-59
5.1.2.2.13.2.14 SelectSUTSettingsPage 5-59
5.1.2.2.13.2.15 ConfigureTestCaseDataPage 5-59
5.1.2.2.13.3 Context Diagram 5-59
5.1.2.2.13.4 Variability Guide 5-59
5.1.2.2.13.5 Rationale 5-59
5.1.2.3 Context Diagram 5-59
5.1.2.4 Variability Guide 5-60
5.1.2.5 Rationale 5-60
5.1.3 Reports GUI 5-60
5.1.3.1 Primary Presentation 5-60
5.1.3.2 Element Catalog 5-60
5.1.3.2.1 ReportsCoordinator 5-60
5.1.3.2.2 TestConfigurationReportPane 5-62
5.1.3.2.3 TestReportsPanel 5-67
5.1.3.2.4 TestLogReportPane 5-69
5.1.3.2.5 PDFViewUI 5-74
5.1.3.3 Context Diagram 5-76
5.1.3.4 Variability Guide 5-76
5.1.3.5 Rationale 5-76
5.1.4 Test GUI 5-77
5.1.4.1 Primary Presentation 5-77
5.1.4.2 Element Catalog 5-77
5.1.4.2.1 TestSelectionUI 5-77
5.1.4.2.2 TestExecutionPanel 5-84
5.1.4.2.3 NavigationPanel 5-90

5.1.4.3 Context Diagram 5-91
5.1.4.4 Variability Guide 5-91
5.1.4.5 Rationale 5-91
5.1.5 Test File Management GUI 5-92
5.1.5.1 Primary Presentation 5-92
5.1.5.2 Element Catalog 5-92
5.1.5.2.1 MaintenanceUI 5-92
5.1.5.3 Context Diagram 5-96
5.1.5.4 Variability Guide 5-96
5.1.5.5 Rationale 5-96
5.2 TEST MODEL 5-97
5.2.1 Test Control 5-97
5.2.1.1 Test Controller 5-97
5.2.1.1.1 Primary Presentation 5-97
5.2.1.1.2 Element Catalog 5-97
5.2.1.1.2.1 RIExecuteTestCase 5-97
5.2.1.1.2.2 RITestEngine 5-99
5.2.1.1.2.3 TestExecutionCoordinator 5-106
5.2.1.1.2.4 RIParameters 5-115
5.2.1.1.3 Context Diagram 5-119
5.2.1.1.4 Variability Guide 5-119
5.2.1.1.5 Rationale 5-119
5.2.1.2 Test Sequencer 5-119
5.3 TEST CONFIGURATION 5-120
5.3.1 Test Script 5-120
5.3.1.1 Primary Presentation 5-120
5.3.1.2 Element Catalog 5-120
5.3.1.2.1 TestCases 5-120
5.3.1.2.2 PostconditionTag 5-123
5.3.1.2.3 PreconditionTag 5-124
5.3.1.2.4 FunctionTag 5-125
5.3.1.2.5 TestStepTag 5-152
5.3.1.2.6 FunctionResult 5-153
5.3.1.2.7 NTCIP2306FunctionTag 5-156
5.3.1.2.8 NTCIP2306SessionTag 5-158
5.3.1.2.9 FTPGetECTag 5-161
5.3.1.2.10 FTPGetOCTag 5-162
5.3.1.2.11 HTTPGetECTag 5-163
5.3.1.2.12 HTTPGetOCTag 5-164
5.3.1.2.13 HTTPPostECTag 5-165
5.3.1.2.14 HTTPPostOCReplyTag 5-166
5.3.1.2.15 HTTPPostOCRequestTag 5-167
5.3.1.2.16 RIValidateTag 5-168
5.3.1.2.17 SOAPPUBECTag 5-169
5.3.1.2.18 SOAPPUBOCTag 5-170
5.3.1.2.19 SOAPRRECTag 5-171
5.3.1.2.20 SOAPRROCReplyTag 5-173
5.3.1.2.21 SOAPRROCRequestTag 5-174
5.3.1.2.22 SOAPSUBECTag 5-175
5.3.1.2.23 SOAPSUBOCTag 5-176
5.3.1.2.24 RIExecuteScriptTag 5-177
5.3.1.2.25 RIMessageVerifyTag 5-182
5.3.1.2.26 RIPauseTag 5-183
5.3.1.2.27 RISubscriptTag 5-183
5.3.1.2.28 RITerminateTag 5-185
5.3.1.2.29 C2CRIFunctionTag 5-186
5.3.1.2.30 C2CRISessionTag 5-187
5.3.1.2.31 PublicationECReceiveTag 5-194
5.3.1.2.32 PublicationECReplyTag 5-195
5.3.1.2.33 PublicationOCTag 5-197

5.3.1.2.34 RIAddEntityElementTag 5-199
5.3.1.2.35 RIAddEntityTag 5-200
5.3.1.2.36 RIDeleteEntityElementTag 5-200
5.3.1.2.37 RIDeleteEntityTag 5-201
5.3.1.2.38 RIGetEntityElementValueTag 5-202
5.3.1.2.39 RIInitializeEntityTag 5-203
5.3.1.2.40 RIMessageVerifyTag 5-204
5.3.1.2.41 RIUpdateEntityElementTag 5-205
5.3.1.2.42 RequestResponseECTag 5-206
5.3.1.2.43 RequestResponseOCReceiveTag 5-208
5.3.1.2.44 RequestResponseOCReplyTag 5-211
5.3.1.2.45 SubscriptionECTag 5-215
5.3.1.2.46 SubscriptionOCReceiveTag 5-216
5.3.1.2.47 SubscriptionOCReplyTag 5-219
5.3.1.2.48 UserVerifyMessageTag 5-221
5.3.1.3 Context Diagram 5-222
5.3.1.4 Variability Guide 5-222
5.3.1.5 Rationale 5-222
5.3.2 Standards 5-223
5.3.2.1 Primary Presentation 5-223
5.3.2.2 Element Catalog 5-223
5.3.2.2.1 TestCaseMatrix 5-223
5.3.2.2.2 ApplicationLayerService 5-225
5.3.2.2.3 ApplicationLayerStandardFactory 5-226
5.3.2.2.4 ApplicationLayerStandard 5-230
5.3.2.2.5 TestSuites 5-238
5.3.2.2.5.1 TestSuite 5-243
5.3.2.3 Context Diagram 5-246
5.3.2.4 Variability Guide 5-246
5.3.2.5 Rationale 5-246
5.3.3 Test User Profile 5-246
5.3.4 Interface Spec 5-246
5.3.4.1 Primary Presentation 5-246
5.3.4.2 Element Catalog 5-247
5.3.4.2.1 NRTM 5-247
5.3.4.3 Context Diagram 5-250
5.3.4.4 Variability Guide 5-250
5.3.4.5 Rationale 5-250
5.3.5 SUT 5-250
5.3.6 Configuration Manager 5-251
5.3.6.1 Primary Presentation 5-251
5.3.6.2 Element Catalog 5-251
5.3.6.2.1 TestConfiguration 5-251
5.3.6.2.2 TestConfigurationList 5-257
5.3.6.2.3 TestDefinitionCoordinator 5-259
5.3.6.2.4 FilenameValidator 5-266
5.3.6.3 Context Diagram 5-267
5.3.6.4 Variability Guide 5-267
5.3.6.5 Rationale 5-267
5.4 CENTER MODEL 5-267
5.4.1 Entity (devices) 5-267
5.4.1.1 Entity Manager 5-267
5.4.1.1.1 Primary Presentation 5-267
5.4.1.1.2 Element Catalog 5-267
5.4.1.1.2.1 Center Active Verification Emulation 5-267
5.4.1.1.2.1.1 Primary Presentation 5-268
5.4.1.1.2.1.2 Element Catalog 5-268
5.4.1.1.2.1.2.1 CenterActiveVerificationHandler 5-268
5.4.1.1.2.1.2.2 CenterActiveVerification 5-268
5.4.1.1.2.1.2.3 TMDDv31EntityEmulator 5-268

5.4.1.1.2.1.2.4 EntityControlRequestStatusCollector 5-269
5.4.1.1.2.1.2.5 EntityCommandProcessor 5-269
5.4.1.1.2.1.2.6 EntityDataOrganizationCollector 5-269
5.4.1.1.2.1.3 Context Diagram 5-269
5.4.1.1.2.1.4 Variability Guide 5-269
5.4.1.1.2.1.5 Rationale 5-269
5.4.1.1.2.2 Node Emulation 5-269
5.4.1.1.2.2.1 Primary Presentation 5-269
5.4.1.1.2.2.2 Element Catalog 5-270
5.4.1.1.2.2.2.1 NodeHandler 5-270
5.4.1.1.2.2.2.2 NodeInventory 5-270
5.4.1.1.2.2.2.3 NodeStatus 5-270
5.4.1.1.2.2.2.4 TMDDv31EntityEmulator 5-270
5.4.1.1.2.2.2.5 EntityControlRequestStatusCollector 5-271
5.4.1.1.2.2.2.6 EntityCommandProcessor 5-271
5.4.1.1.2.2.2.7 EntityDataOrganizationCollector 5-271
5.4.1.1.2.2.3 Context Diagram 5-271
5.4.1.1.2.2.4 Variability Guide 5-271
5.4.1.1.2.2.5 Rationale 5-271
5.4.1.1.2.3 Link Emulation 5-271
5.4.1.1.2.3.1 Primary Presentation 5-271
5.4.1.1.2.3.2 Element Catalog 5-272
5.4.1.1.2.3.2.1 LinkHandler 5-272
5.4.1.1.2.3.2.2 LinkInventory 5-272
5.4.1.1.2.3.2.3 LinkStatus 5-272
5.4.1.1.2.3.2.4 TMDDv31EntityEmulator 5-272
5.4.1.1.2.3.2.5 EntityControlRequestStatusCollector 5-272
5.4.1.1.2.3.2.6 EntityCommandProcessor 5-273
5.4.1.1.2.3.2.7 EntityDataOrganizationCollector 5-273
5.4.1.1.2.3.3 Context Diagram 5-273
5.4.1.1.2.3.4 Variability Guide 5-273
5.4.1.1.2.3.5 Rationale 5-273
5.4.1.1.2.4 CommandQueue 5-1
5.4.1.1.2.5 RIEmulation 5-2
5.4.1.1.2.6 CCTVHandler 5-18
5.4.1.1.2.7 DetectorHandler 5-21
5.4.1.1.2.8 DMSHandler 5-24
5.4.1.1.2.9 ESSHandler 5-28
5.4.1.1.2.10 EventHandler 5-31
5.4.1.1.2.11 GateHandler 5-33
5.4.1.1.2.12 HARHandler 5-37
5.4.1.1.2.13 LaneControlSignalHandler 5-40
5.4.1.1.2.14 RampMeterHandler 5-44
5.4.1.1.2.15 SignalSectionHandler 5-48
5.4.1.1.2.16 TrafficControllerHandler 5-52
5.4.1.1.2.17 VideoSwitchHandler 5-57
5.4.1.1.3 Context Diagram 5-59
5.4.1.1.4 Variability Guide 5-59
5.4.1.1.5 Rationale 5-59
5.4.1.2 Entity 5-60
5.4.1.2.1 Primary Presentation 5-60
5.4.1.2.2 Element Catalog 5-61
5.4.1.2.2.1 cctv 5-61
5.4.1.2.2.1.1 CCTVInventory 5-61
5.4.1.2.2.1.2 CCTVStatus 5-62
5.4.1.2.2.2 detector 5-64
5.4.1.2.2.2.1 ArchivedDataProcessingDocumentationMetaData 5-64
5.4.1.2.2.2.2 DetectorData 5-65
5.4.1.2.2.2.3 DetectorInventory 5-67
5.4.1.2.2.2.4 DetectorMaintenanceHistory 5-69
5.4.1.2.2.2.5 DetectorStatus 5-70
5.4.1.2.2.3 dms 5-72

5.4.1.2.2.3.1 DMSFontTable 5-72
5.4.1.2.2.3.2 DMSInventory 5-73
5.4.1.2.2.3.3 DMSMessageAppearance 5-75
5.4.1.2.2.3.4 DMSMessageInventory 5-77
5.4.1.2.2.3.5 DMSStatus 5-78
5.4.1.2.2.4 ess 5-80
5.4.1.2.2.4.1 ESSInventory 5-80
5.4.1.2.2.4.2 ESSObservationMetadata 5-81
5.4.1.2.2.4.3 ESSObservationReport 5-83
5.4.1.2.2.4.4 ESSStatus 5-85
5.4.1.2.2.5 events 5-86
5.4.1.2.2.5.1 EventUpdates 5-86
5.4.1.2.2.5.2 EventsActionLog 5-88
5.4.1.2.2.5.3 EventsIndex 5-89
5.4.1.2.2.6 gate 5-91
5.4.1.2.2.6.1 GateControlSchedule 5-91
5.4.1.2.2.6.2 GateInventory 5-93
5.4.1.2.2.6.3 GateStatus 5-94
5.4.1.2.2.7 har 5-96
5.4.1.2.2.7.1 HARControlSchedule 5-96
5.4.1.2.2.7.2 HARInventory 5-97
5.4.1.2.2.7.3 HARMessageInventory 5-99
5.4.1.2.2.7.4 HARStatus 5-101
5.4.1.2.2.8 lcs 5-102
5.4.1.2.2.8.1 LaneControlSignalControlSchedule 5-102
5.4.1.2.2.8.2 LaneControlSignalInventory 5-104
5.4.1.2.2.8.3 LaneControlSignalStatus 5-106
5.4.1.2.2.9 rampmeter 5-108
5.4.1.2.2.9.1 RampMeterControlSchedule 5-108
5.4.1.2.2.9.2 RampMeterInventory 5-110
5.4.1.2.2.9.3 RampMeterPlanInventory 5-111
5.4.1.2.2.9.4 RampMeterStatus 5-113
5.4.1.2.2.10 signalsection 5-115
5.4.1.2.2.10.1 SectionControlSchedule 5-115
5.4.1.2.2.10.2 SectionSignalTimingPatternInventory 5-116
5.4.1.2.2.10.3 SectionStatus 5-118
5.4.1.2.2.11 trafficcontroller 5-120
5.4.1.2.2.11.1 IntersectionSignalControllerControlSchedule 5-120
5.4.1.2.2.11.2 IntersectionSignalControllerInventory 5-122
5.4.1.2.2.11.3 IntersectionSignalControllerStatus 5-123
5.4.1.2.2.11.4 IntersectionSignalControllerTimingPatternInventory 5-125
5.4.1.2.2.12 videoswitch 5-126
5.4.1.2.2.12.1 VideoSwitchInventory 5-127
5.4.1.2.2.12.2 VideoSwitchStatus 5-128
5.4.1.2.3 Context Diagram 5-129
5.4.1.2.4 Variability Guide 5-129
5.4.1.2.5 Rationale 5-129
5.4.1.3 Parameter 5-129
5.4.1.3.1 Primary Presentation 5-129
5.4.1.3.2 Element Catalog 5-129
5.4.1.3.2.1 RIEmulationParameters 5-129
5.4.1.3.2.2 ParameterFile 5-130
5.4.1.3.3 Context Diagram 5-131
5.4.1.3.4 Variability Guide 5-131
5.4.1.3.5 Rationale 5-131
5.4.1.4 Entity Namer (1104) 5-131
5.4.2 Behavioral 5-132
5.4.2.1 Connection 5-132
5.4.2.2 Authentication 5-132
5.4.2.2.1 Primary Presentation 5-132
5.4.2.2.2 Element Catalog 5-132
5.4.2.2.2.1 TMDDAuthenticationProcessor 5-132

5.4.2.2.3 Context Diagram 5-133
5.4.2.2.4 Variability Guide 5-133
5.4.2.2.5 Rationale 5-133
5.4.2.3 Event Model 5-133
5.5 MESSAGE MANAGER 5-134
5.5.1 C2C Message Manager 5-134
5.5.1.1 Primary Presentation 5-134
5.5.1.2 Element Catalog 5-135
5.5.1.2.1 Message 5-135
5.5.1.3 Context Diagram 5-143
5.5.1.4 Variability Guide 5-143
5.5.1.5 Rationale 5-143
5.5.2 Validator 5-143
5.5.3 Parser 5-143
5.5.4 Distributer 5-143
5.5.5 Creator 5-143
5.5.5.1 Primary Presentation 5-143
5.5.5.2 Element Catalog 5-144
5.5.5.2.1 TMDDErrorResponseProcessor 5-144
5.5.5.3 Context Diagram 5-147
5.5.5.4 Variability Guide 5-147
5.5.5.5 Rationale 5-147
5.6 LOGGER 5-148
5.6.1 Log Manager 5-148
5.6.1.1 Primary Presentation 5-148
5.6.1.2 Element Catalog 5-148
5.6.1.2.1 RILogging 5-148
5.6.1.2.2 LogFileProcessor 5-152
5.6.1.3 Context Diagram 5-161
5.6.1.4 Variability Guide 5-161
5.6.1.5 Rationale 5-161
5.6.2 Logger 5-162
5.6.2.1 Primary Presentation 5-162
5.6.2.2 Element Catalog 5-162
5.6.2.2.1 RIXMLLayout 5-162
5.6.2.3 Context Diagram 5-164
5.6.2.4 Variability Guide 5-164
5.6.2.5 Rationale 5-164
5.6.3 Handler 5-164
5.6.3.1 Primary Presentation 5-164
5.6.3.2 Element Catalog 5-164
5.6.3.2.1 log4j 5-164
5.6.3.3 Context Diagram 5-166
5.6.3.4 Variability Guide 5-166
5.6.3.5 Rationale 5-166
5.6.4 Reader 5-166
5.7 REPORTER 5-167
5.7.1 Report Manager 5-167
5.7.1.1 Primary Presentation 5-167
5.7.1.2 Element Catalog 5-167
5.7.1.2.1 IcePDF 5-167
5.7.1.2.2 JasperReports 5-168
5.7.1.2.3 RIReports 5-170
5.7.1.3 Context Diagram 5-175
5.7.1.4 Variability Guide 5-175
5.7.1.5 Rationale 5-175
5.7.2 Test Config Report 5-175
5.7.3 PassFail Report 5-175

5.7.4 Test Activity Report *5-175*
5.8 C2C SUT INTERFACE 5-176
5.8.1 Center Services *5-176*
5.8.1.1 Primary Presentation 5-176
5.8.1.2 Element Catalog 5-176
5.8.1.2.1 InformationLayerService 5-176
5.8.1.2.2 InformationLayerStandardFactory 5-177
5.8.1.2.3 NTCIP2306ApplicationLayerOperationResults 5-180
5.8.1.2.4 NTCIP2306Controller 5-182
5.8.1.2.5 Publication 5-191
5.8.1.2.6 SubPubController 5-199
5.8.1.2.7 Subscription 5-206
5.8.1.2.8 NTCIP2306ApplicationLayerStandard 5-206
5.8.1.2.9 TMDDv303Controller 5-215
5.8.1.2.10 TMDDv303InformationLayerStandard 5-218
5.8.1.2.11 TMDDv303OperationResults 5-221
5.8.1.2.12 TMDDv303SubPubMapping 5-224
5.8.1.3 Context Diagram 5-226
5.8.1.4 Variability Guide 5-226
5.8.1.5 Rationale 5-226
5.8.2 Client Services *5-226*
5.8.3 XML/SOAP Support *5-226*
5.8.3.1 NTCIP 2306 5-226
5.8.3.2 WSDL 5-226
5.8.3.3 SOAP 5-226
5.8.3.4 XML 5-226
5.8.3.5 HTTP 5-227
5.8.3.5.1 Primary Presentation 5-227
5.8.3.5.2 Element Catalog 5-227
5.8.3.5.2.1 HTTPClient 5-227
5.8.3.5.2.2 Jetty 5-228
5.8.3.5.3 Context Diagram 5-229
5.8.3.5.4 Variability Guide 5-229
5.8.3.5.5 Rationale 5-229
5.8.4 DATEX-ASN Support (Deprecated) *5-229*
5.8.4.1 NTCIP 2304 5-229
5.8.4.2 ASN.1 5-229
5.8.4.3 UDP/IP 5-229
5.8.4.4 TCP/IP 5-229
5.9 FTPCLIENT 5-229
5.10 FTPSERVER 5-231
5.11 JAVA 5-232
6 FIGURES 6-1
6.1 REPORTGENERATION 6-1
6.1.1 :JasperReport *6-1*
6.1.1.1 :RIGUI 6-2
6.1.1.2 :RIReports 6-2
6.1.1.3 :ReportDatabase 6-2
6.1.1.4 :TestConfiguration 6-2
6.1.1.5 1201ConformanceReport.jrxml 6-3
6.1.1.6 TestLogxxx.xml 6-3
6.2 ENTITY EMULATION OVERVIEW 6-4
6.2.1 External System Under Test *6-9*
6.2.1.1 :ApplicationLayerStandard 6-10
6.2.1.2 :C2CRIFunctionTag 6-10
6.2.1.3 :EntityEmulationService 6-10
6.2.1.4 :Tmddv303Controller 6-10

6.2.1.5 TMDDv303cDatabase 6-11
6.2.2 Entity Emulation Dialog Processing 6-11
6.2.3 Processing of Commands from an EC 6-37
6.2.4 SubscriptionPublication 6-44
6.3 APPLICATION LAYER CONTROL 6-45
6.3.1 Application Layer Control Sequence Diagram - (Interaction diagram) 6-45
6.3.2 Application Layer Control with Entity Emulation Sequence Diagram - (Interaction diagram) 6-46
6.3.3 Information Layer and Application Layer Standard Creation Sequence Diagram - (Interaction diagram) 6-47
6.3.4 Test Execution Sequence Diagram - (Interaction diagram) 6-48
6.3.5 Other Figures 6-49
7 APPENDIX A - REQUIREMENTS TRACEABILITY MATRIX 7-1

LIST OF FIGURES

FIGURE 3-1: RI OPERATIONAL ENVIRONMENT (ADAPTED FROM TMDD CENTER-TO-CENTER CONCEPT OF OPERATIONS AND REQUIREMENTS STANDARD) 3-1
FIGURE 3-2: RI SYSTEM DEPLOYMENT DIAGRAM 3-2
FIGURE 4-1: RI HIGH LEVEL ARCHITECTURE 4-2
FIGURE 4-2: GUI SUB-COMPONENTS 4-3
FIGURE 4-3: TEST MODEL HIGH LEVEL SUB-COMPONENTS 4-4
FIGURE 4-4: TEST CONTROL COMPONENTS 4-5
FIGURE 4-5: TEST CONFIGURATION SUB-COMPONENTS 4-6
FIGURE 4-6: CENTER MODEL COMPONENT 4-8
FIGURE 4-7: MESSAGE MANAGER COMPONENT 4-9
FIGURE 4-8: LOGGER SUB-COMPONENTS 4-10
FIGURE 4-9: REPORTER COMPONENT 4-11
FIGURE 4-10: C2C SUT INTERFACE COMPONENT 4-13
FIGURE 5-1 MAIN UI MODULES 5-1
FIGURE 5-2 TEST DEFIINITION GUI MODULES 5-18
FIGURE 5-3 TEST CASE DATA EDITOR MODULES 5-50
FIGURE 5-4 TEST CASE DATA EDITOR 5-51
FIGURE 5-5: C2C RI WIZARD MODULES 5-52
FIGURE 5-6 C2C RI WIZARD INITIAL SELECTION PANEL 5-53
FIGURE 5-7 CREATE TEST CONFIGURATION MAIN PAGE 5-55
FIGURE 5-8 EXIT OR EDIT PAGE 5-56
FIGURE 5-9 NEW TEST CONFIGURATION PAGE 5-57
FIGURE 5-10 TEST MODE SELECTION PAGE 5-58
FIGURE 5-11 REPORTS GUI MODULES 5-60
FIGURE 5-12 TEST GUI MODULES 5-77
FIGURE 5-13: TEST DEFINITION SCREEN 5-84
FIGURE 5-14 TEST FILE MANAGEMENT GUI MODULES 5-92
FIGURE 5-15 TEST CONTROLLER MODULES 5-97
FIGURE 5-16 TEST SCRIPT MODULES 5-120
FIGURE 5-17 STANDARDS MODULES 5-223
FIGURE 5-18 INTERFACE SPEC MODULES 5-246
FIGURE 5-19 CONFIGURATION MANAGER MODULES 5-251
FIGURE 5-20 CENTER ACTIVE VERIFICATION EMULATION MODULES 5-268
FIGURE 5-21 NODE EMULATION MODULES 5-270
FIGURE 5-22 LINK EMULATION MODULES 5-272
FIGURE 5-23 ENTITY MODULES 5-60
FIGURE 5-24 PARAMETER MODULES 5-129
FIGURE 5-25 AUTHENTICATION MODULES 5-132
FIGURE 5-26 C2C MESSAGE MANAGER MODULES 5-134
FIGURE 5-27 CREATOR MODULES 5-143
FIGURE 5-28 LOG MANAGER MODULES 5-148
FIGURE 5-29 LOGGER MODULES 5-162
FIGURE 5-30 HANDLER MODULES 5-164
FIGURE 5-31 REPORT MANAGER MODULES 5-167
FIGURE 5-32 CENTER SERVICES MODULES 5-176

FIGURE 5-33 HTTP MODULES 5-227
FIGURE 6-1: REPORT GENERATION COMPONENTS 6-1
FIGURE 6-2: TMDD ENTITY EMULATION OVERVIEW COMPONENT DIAGRAM 6-4
FIGURE 6-3: OBJECTS INVOLVED IN SUBSCRIPTION PUBLICATION 6-44
FIGURE 6-4: APPLICATION LAYER CONTROL SEQUENCE DIAGRAM 6-45
FIGURE 6-5: APPLICATION LAYER CONTROL WITH ENTITY EMULATION SEQUENCE DIAGRAM 6-46
FIGURE 6-6: INFORMATION LAYER AND APPLICATION LAYER STANDARD CREATION SEQUENCE DIAGRAM 6-47
FIGURE 6-7: TEST EXECUTION SEQUENCE DIAGRAM 6-48
FIGURE 6-8: DRAFT 1201 CONFORMANCE REPORT 6-49
FIGURE 6-9: RUN A TEST COMMUNICATION DIAGRAM 6-50
FIGURE 6-10: CREATE LOG REPORT COMMUNICATION DIAGRAM 6-51
FIGURE 6-11: CREATE LOG REPORT SEQUENCE DIAGRAM 6-52

1 INTRODUCTION

1.1 Identification

The Center-to-Center (C2C) Reference Implementation (RI) Software Design Document (SDD) specifies the software design for use during the implementation phase of the RI application.

1.2 Purpose

This document describes the software design for a Center-to-Center *Reference Implementation* (RI). The software design was developed to satisfy the user needs specified in the Concept of Operations (ConOps) for a Center-to-Center Reference Implementation document and the requirements specified in the Software Requirements Specification (SRS). This design document will be used during the implementation and support phases of the RI lifecycle to guide the development of the software. The RI software designer and developer will update and maintain this document throughout the product life cycle as new requirements are incorporated into the product.

It is anticipated that the FHWA will use this design description to monitor the software development and to verify that the fully developed RI will satisfy both the requirements specification and user needs.

1.3 Scope

There is currently no established procedure for demonstrating/testing conformance with the ITS C2C standards developed with funding from the FHWA by the various Standards Development Organizations including ITE, NEMA, AASHTO, IEEE, and SAE. The C2C RI is intended to support the efficient deployment of systems using C2C communications by providing a means of verifying conformance to the standards. Providing the C2C RI to organizations deploying C2C communications will reduce development risks, effort and costs. Where ambiguities exist within the standards, it is hoped that the RI will provide a common interpretation that will improve the interoperability among systems from various integrators.

The primary objective of the C2C RI is to provide consistent methods and tools to verify conformance with the C2C communications standards. Secondary objectives are to reduce the cost and effort of deploying C2C communications systems, which in turn encourages deployment of more systems. Increasing the number of C2C systems deployed increases the ability to manage the transportation network by supporting data sharing and shared control of traffic management devices (our ITS assets) as well as providing information to travelers enabling them to base trip decisions on near real-time information.

Finally, a RI can simplify and lower the risks of deploying the regional network for real-time data collection since that program (1201) is intended to leverage from the existing standards.

A survey of potential RI users revealed the following needs, results and conclusions:

- It is desired that the RI be computer platform independent [i.e. it should run on Linux, Windows, and potentially other environments.]
- The RI uses will include C2C development, integration and conformance testing. Note that each of these uses has differing requirements and could lead to different RI implementations.
- There is a desire for both predefined and user developed tests.

- There are varying desires for standards to be supported by the RI. The following are consistently recognized as important:
    - Traffic Management Data Dictionary (TMDD) and Message Sets for External Traffic Management Center Communications (MS/ETMCC)
    - IEEE 1512 A family of standards[1] for Incident Management and Traffic Incident Related messages relating to public safety and emergency management centers and services.
    - SAE J2354 Data Dictionary and Message Sets for Advanced Traveler Information Systems (ATIS)
    - SAE J2266 Local Referencing Message Specification (LRMS)
- All agencies planning new deployments that responded to the survey are planning to use SOAP and XML; however, there are existing deployments using DATEX and XML and as technology progresses there will be new implementation approaches.
- Responders were concerned that the RI will attempt to accomplish too much initially. It was preferred to get a basic capability working and then enhance the RI over time.
- The RI should be developed to determine conformance to C2C standards. When local extensions are used, it will be the users' responsibility to determine conformance with the local extensions.

1.4 Definitions and Acronyms

IEEE 1512	A series of standards for Incident Management and Traffic Incident Related messages relating to public safety and emergency management centers and services.
AASHTO	American Association of State Highway and Transportation Officials
Application Layer Standards	ITS C2C standards which define how conversations between centers are managed, how data transferred is encoded, and the syntax and grammar rules for the conversation.
ATIS	Advance Traveler Information System

[1] The IEEE 1512 Family of Standards consists of a Base Standard and a set of companion volumes. The purpose of these volumes is to more fully establish the message sets used by various user groups within the realm of transportation management, public safety, and hazardous materials. Throughout the ConOps, they will be referred to as the *1512 Family of Standards* which is intended to refer to the following:

- IEEE Std 1512-2000, **Common Incident Management Message Sets** for Use by Emergency Management Centers, or The Base Standard
- IEEE Std 1512.1-2003, **Traffic Incident Management Message Sets** for Use by Emergency Management Centers
- IEEE Std 1512.2-2004, **Public Safety Incident Management Message Sets** for Use by Emergency Management Centers
- IEEE Std 1512.3-2002, **Hazardous Material Incident Management Message Sets** for Use by Emergency Management Centers
- IEEE Std 1512a-2002, **Emergency Management Data Dictionary** (available electronically only)

C2C	Center-to-Center
CCTV	Closed Circuit Television
ConOps	Concept of Operations
COTS	Commercial Off the Shelf
CSV	Comma Separated Value File Format
DATEX	A general purpose C2C data exchange protocol stack.
Html	Hypertext Markup Language
IEEE	The Institute of Electrical and Electronics Engineers
Information Layer Standards	ITS standards for data elements, objects and messages to be transmitted between ITS entities.
ITE	Institute of Transportation Engineers
ITS	Intelligent Transportation Systems
LRMS	Location Reference Message Specification
MS/ETMCC	Message Sets and External Traffic Management Center Communication
NEMA	National Electrical Manufacturers Association
Owner Center	A traffic management center that has direct control of the traffic devices
PDF	Portable Document Format
pre-defined tests	Standards conformance tests included with the RI.
Requesting center	A center that uses the center-to-center services provided by an Owner Center.
RI	C2C Reference Implementation
RI-SUT Communications interface	The communication interface that supports communications between the RI and the SUT.
SAE	Society of Automotive Engineers
SOAP	Simple Object Access Protocol
SRS	Software Requirements Specification
SUT	System Under Test
TCP/IP	Transmission Control Protocol / Internet Protocol
Test Configuration	The set of parameters, scripts and settings which are needed to establish a conformance test.
TMDD	Traffic Management Data Dictionary
UDP	User Datagram Protocol
User-Defined Tests	Standards conformance tests which are created by the RI users
XML	Extensible Markup Language

1.5 Overview

The format and content of this SDD is based on the IEEE standards for Software Design Descriptions. An SDD provides a model of a software systems implementation. It includes technical language and model diagrams appropriate for software developers and technical reviewers.

A common practice for the design of complex software systems is to divide the system into logical components. Each component contains a manageable set of like system functions and goals. This practice is applied within this document. Architectural components (high-levels of grouped functionality)

are introduced first. Then each design component (low-levels of grouped functionality) is presented and discussed in relation to their associated Architectural component(s).

Section 3 provides an overall description of the RI system, its physical deployment view as well as a view of its logical packages. Detailed descriptions of the logical software packages are reserved for subsequent sections.

Section 4 describes each major architectural component of the RI system. The elements of each component are described in text format, and by diagrams which provide additional design insight on the component.

Section 5 provides detailed software design content for the RI. Each subsection begins with an overview of the design components and their relation to the major architectural components discussed in Section 4. The overview section is followed by detailed descriptions of the RI design components and classes.

Section 6 provides a number of behavioral UML diagrams (sequence, communications, etc) that illustrate the interaction of the components detailed in Section 5.

Section 7 provides design information applicable to the following enhancements to the C2C RI:

- Guided (Wizard) Mode
- Test Case Data Editor
- Center Active Verification, Link and Node entity emulation

Appendix A contains the detailed RI software requirements and provides traceability back to the appropriate section(s) of the document where the requirement is addressed by the design.

2 REFERENCES

	Concept of Operations for a Center to Center Reference Implementation
	Software Requirements Specification for a Center to Center Reference Implementation
ITE TMDD	Traffic Management Data Dictionary (TMDD) and Message Sets for External Traffic Management Center Communications (MS/ETMCC) (Version 3.03c and newer)
IEEE 1512	A series of standards for Incident Management and Traffic Incident Related messages relating to public safety and emergency management services.
SAE J2354	Data Dictionary and Message Sets for Advanced Traveler Information Systems (ATIS)
SAE J2266	Local Referencing Message Specification (LRMS)
IEEE STD 1016-1998	IEEE Recommended Practice for Software Design Descriptions
IEEE STD 829-1998	Standard for Software Test Documentation
IEEE STD 830-1998	Recommended Practice for Software Requirements Specifications
IEEE STD 1063-2001	Standard for Software User Documentation
NTCIP 2306	Application Profile for XML Message Encoding and Transport in ITS Center-to-Center Communications (NTCIP-C2C XML)
NTCIP 2304	Application Profile for DATEX-ASN (AP-DATEX)

3 RI SYSTEM

3.1 Description

The RI System will provide a C2C standards conformance tool for agencies and system developers involved in the deployment of C2C compliant centers.

The RI System will consist of a software application hosted on a common PC platform which will tie into the SUT via an Ethernet network utilizing TCP/IP transport protocol. Figure 3-1 presents a high level concept diagram of the RI system and how it ties into an external SUT.

C2C Systems developers will use the RI as they are developing C2C standards compliant interfaces, to ensure that their implementation of the standards is consistent with the established practice. The RI will provide them with the means to specify with which of the established dialogs, messages, data elements and protocols their systems are compliant. The RI may be used to determine compliance to both subsets (particular requirements, messages, etc.), as they develop their interfaces, as well as complete standards during integration or full compliance testing. In support of this testing, the RI will produce reports which indicate which of the compliance tests completed successfully and which failed with details on what failure occurred.

An additional expected use of the RI systems is that agencies acquiring C2C software solutions can use the RI to confirm vendor claims on C2C standards compliance.

The RI system will be maintained by a 3rd party which will provide updates for the RI as standards continue to evolve. The initial RI design makes accommodations for Web Services and DATEX-ASN application layer standards, but will be extendible to future technologies which may be adopted.

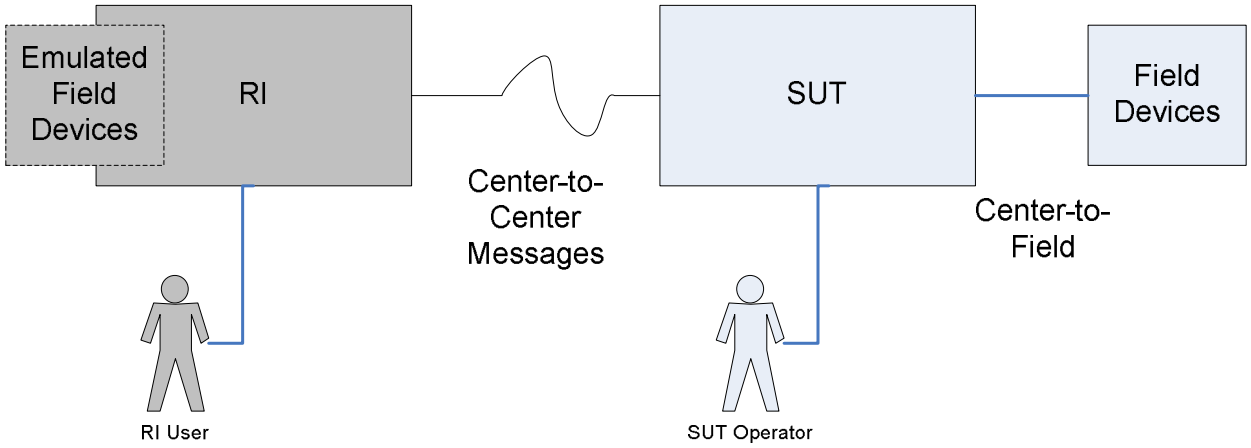

Figure 3-1: RI Operational Environment
(Adapted from TMDD Center-to-Center Concept of Operations and Requirements Standard)

3.2 Deployment Diagram

The fundamental deployment hardware and software for the RI system is as shown in Figure 3-2. The RI system software will be made available for users to deploy on a standard PC. The user will be able to run the RI software on a number of operating systems including Windows and Linux. All application

executables, source code, script files and any necessary 3rd party software will be included with the RI software system installation package.

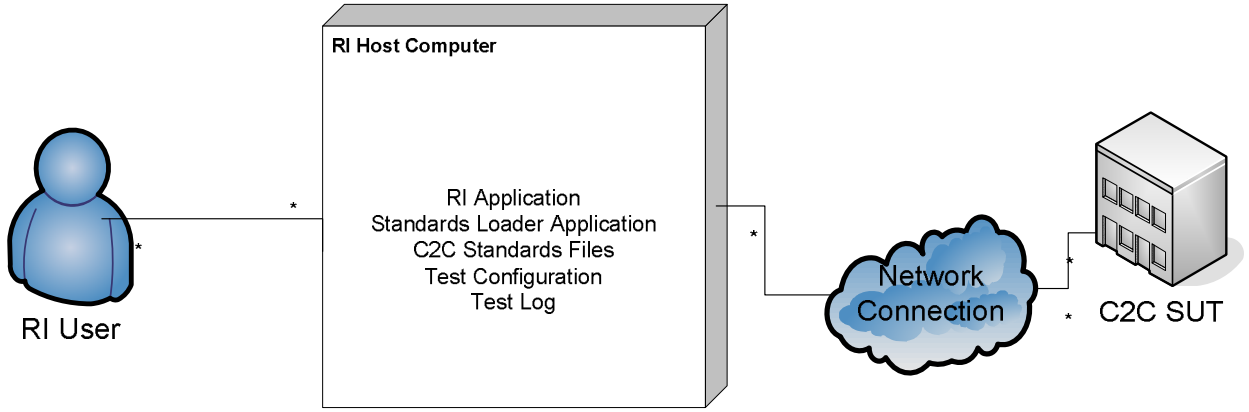

Figure 3-2: RI System Deployment Diagram

4 HIGH LEVEL ARCHITECTURE

This section provides a high level overview of the RI's basic operation and the primary architectural elements, including a description of their functions and interfaces. Describing the RI architecture at a high level provides the following benefits:

- facilitates project stakeholder and developer understanding of the RI's expected capabilities and operation
- provides an intermediate level of concept development between the RI's software requirements and detailed design
- allows for a logical flow of traceability between RI user needs, RI software requirements and RI software design packages, classes and methods

4.1 Basic RI Operation

RI operation begins with the configuration of the system by the RI User. The user creates or selects a test configuration applicable to the planned test of the SUT. In performing this step the user specifies the following:

- the information level standards (TMDD, IEE1512, etc.) to be included in the test
- the set of information level standard requirements which will be tested
- the application level standard (NTCIP 2306 or NTCIP 2304) and associated profile (i.e. SOAP/XML, XML Direct, DATEX-ASN.1, etc.)
- parameters defining the SUT (name, IP address, etc.)
- entities (CCTV, DMS, events, etc.) to be maintained by the RI when acting as an Owner Center during the test
- The test parameters to be used in executing the test defined in Test Suites. Test Suites include test scripts and associated data to implement test cases and associated test steps required to verify conformance to the selected standard requirements. Test parameters are used during the performance of Test Suites to determine how the Test Suites perform.
- options for the Logging of messages to the GUI and test log during the test

The user then initiates the start of the test. After this, the RI executes the defined test scenarios, sending and receiving messages to the SUT as either an Owner Center or an External Center. The interface for message exchange with the SUT (information and application layer protocols) will be as specified in the test configuration. The RI continues to run the test, logging messages and diagnostics, until either the test scenario is completed or the RI user halts the test.

Following the test, the RI user can generate a number of reports from the logged data to analyze test activities and determine the SUT's conformance to the tested standards.

4.2 Architectural Components

The high level architecture of the RI software is represented in Figure 4-1. The major architectural components represent major groups of RI functionality and are defined as follows:

- GUI – Provides the user interface to the RI system for defining, running and analyzing tests
- Test Model – Manages the test configuration and provides for control and execution of the test
- Center Model – Establishes and maintains the emulated devices and information elements associated with a center
- Message Manager – Manages the creation, validation and distribution of messages between the RI and the SUT
- Logger – Provides the mechanism to log the activities and messages of a test and to retrieve the information for reporting
- Reporter – Provides the means to generate various reports from the test configuration and the test log
- C2C SUT Interface – Establishes the application and transport level interface between the RI and the SUT

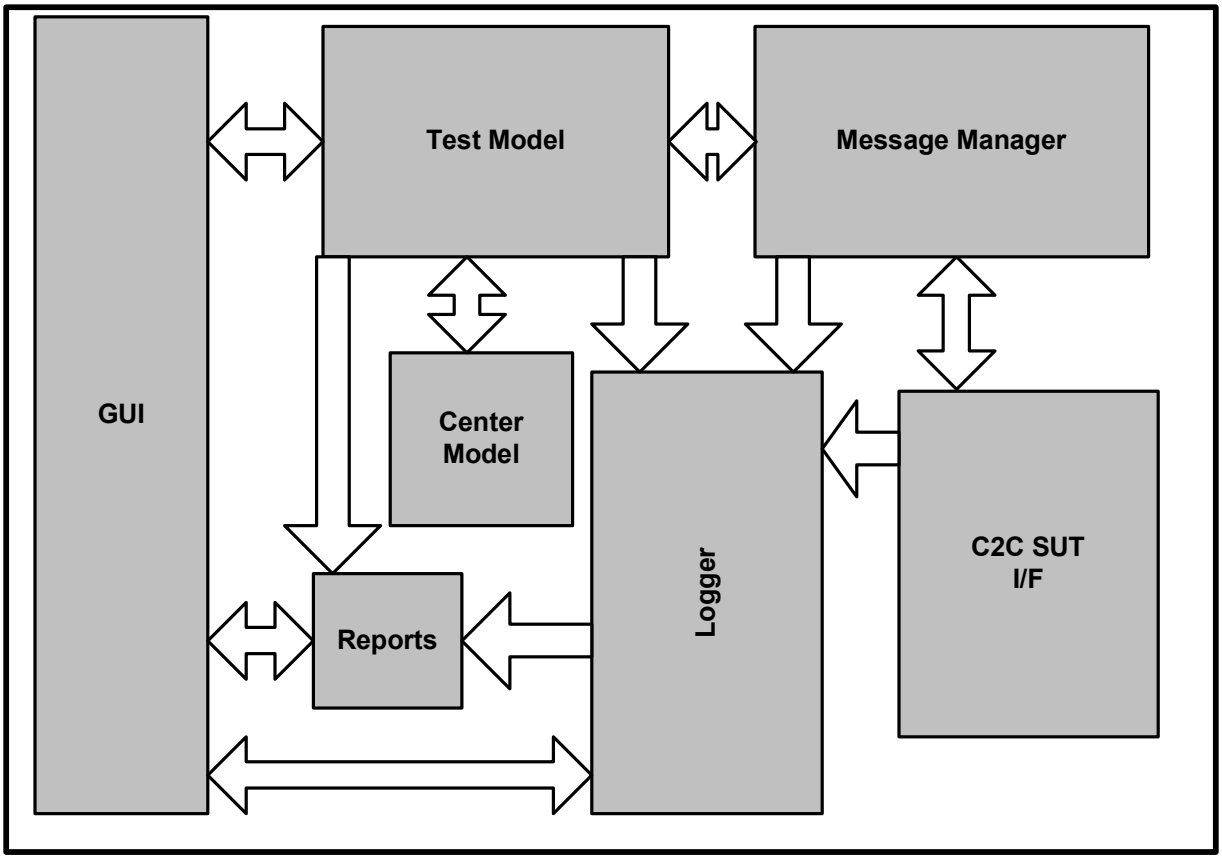

Figure 4-1: RI High Level Architecture

The following sections further describe each of these major components, and their interactions with other components.

4.2.1 GUI

The GUI provides the interface for the RI operator to define and conduct tests. It presents various menus and dialog windows to the user which allow him to specify test requirements, specify the C2C interface, generate and view reports and conduct tests.

GUI interfaces with the Test Model, Reports and Logger elements. The GUI interfaces with Test Model to define and view test configurations, and to control the execution of a test. The Reports interface allows the GUI to specify the generation of various reports. All user actions performed at the initiation, during and conclusion of a test are logged by the Logger.

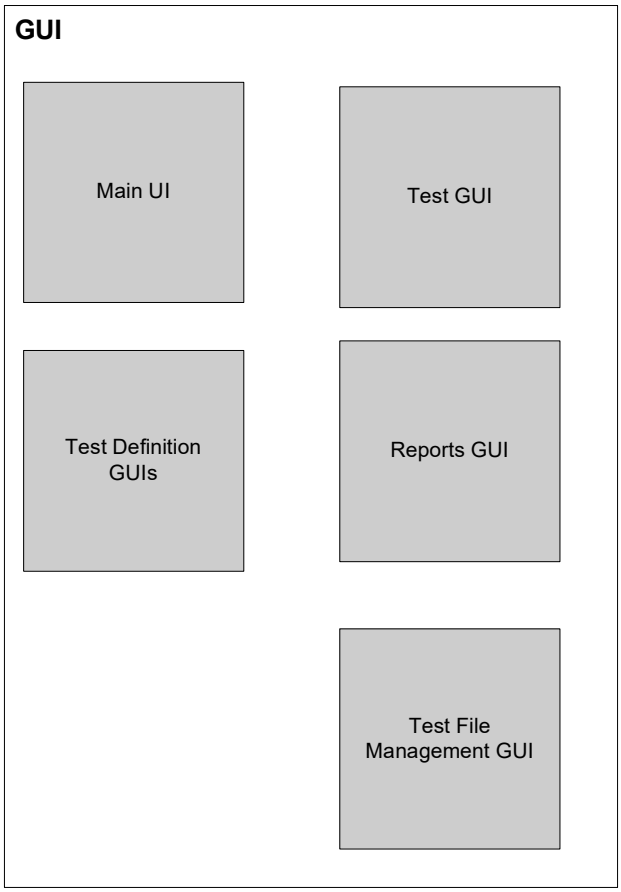

Figure 4-2: GUI sub-components

The GUI component can be divided into a number of GUI sub-components (Figure 4-2), each providing the user with access to distinct areas of RI functionality. Each GUI sub-component is described below.

Main UI – the Main UI sub-component provides the user with top level access and control of the RI and all other GUI sub-components.

Test Definition GUIs – provides the RI user with the menu dialogs or guided wizard screens needed to define and manage the RI test configuration.

Reports GUI – provides the RI user with options to generate and view reports on test configurations and test logs.

Test GUI – provides the RI user with the menu dialogs necessary to control the start/stop/pause/restart of an RI test and to view messages and diagnostics as the test is running.

Test File Management GUI – provides the RI user with the capability to manage (delete, export) RI test logs and to delete Test Configurations.

4.2.2 Test Model

The Test Model maintains all things related to test configurations including the test suites and test configuration parameters. It also provides test control and test script execution functionality. The Test Model interfaces with the following components:

- GUI component, to provide the user with the ability to define and manage test configurations and execute tests.
- Test Model interfaces with the Logger to ensure that all test actions are recorded.
- Center Model component, to trigger updates to the emulated center devices.
- Message Manager component, to process received messages from and to trigger the creation and sending of messages to the SUT.
- Reporter, to allow for the generation of reports detailing the test definition.

The Test Model consists of two major sub-components, Test Control and Test Configuration as shown in Figure 4-3.

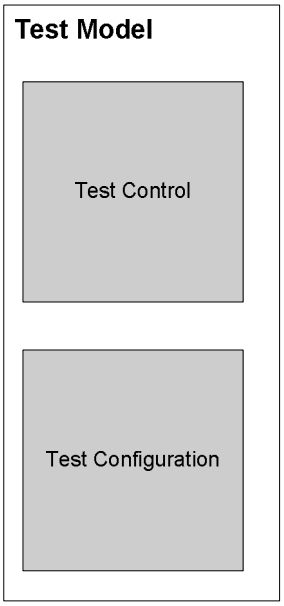

Figure 4-3: Test Model High Level Sub-components

4.2.2.1 Test Control

The Test Control sub-component includes all functions required to execute the defined test with the SUT. The sub-components which make up this component are shown in Figure 4-4 and are described below:

Test Controller – includes the functionality required to start, stop, pause and restart the RI test.

Test Sequencer – includes the functionality required to establish, execute and provide status on test scenarios that are run during the test.

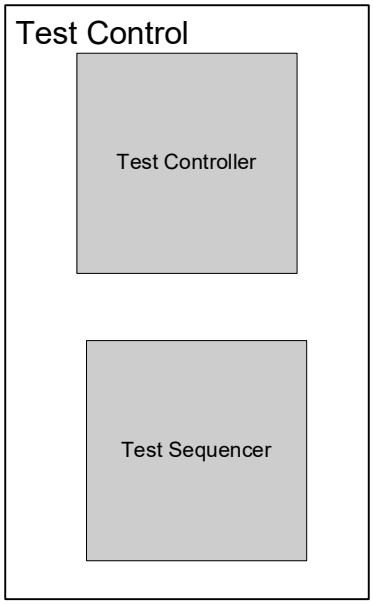

Figure 4-4: Test Control Components

4.2.2.2 Test Configuration

The Test Configuration sub-component includes all functions required to define the parameters for an RI test. It interfaces with the GUI component to allow the user to view and specify the configuration elements. The sub-components which make up this component are shown in Figure 4-5 and are described below:

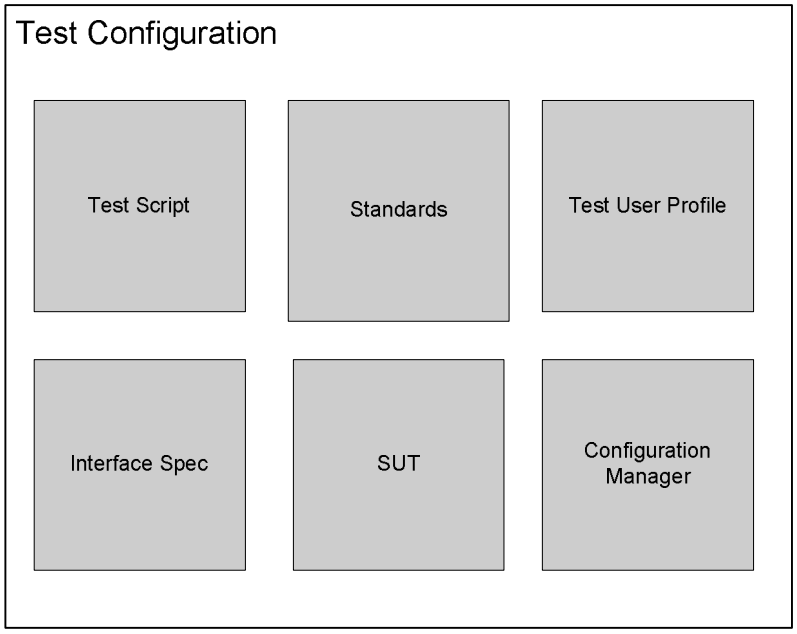

Figure 4-5: Test Configuration sub-components

Test Script – includes the capability to maintain a test script (or scenario) for a test.

Standards –maintains the Standards Library which are available to RI users in the definition of a test.

Test User Profile – allows for the identification of the RI user which defines the test configuration and /or executes a test. This information is maintained within the test configuration and the test log.

Interface Spec – contains the definition of the specific standards (information-level and application-level) that are to be tested between the RI and the SUT.

SUT – maintains the data which is required to define and connect to the SUT including system name, IP address, authentication info, etc.

Configuration Manager – provides the functions necessary to manage the test configurations including create, delete, and modify.

4.2.3 Center Model

The Center Model component encompasses the RI's ability to model the entities associated with the Centers. The Center Model receives its configuration definitions from the Test Configuration defined within the Test Model. The Center Model also responds to entity update and status requests from the test model. The Center Model consists of the Entity (devices) and behavioral sub-components. The architecture of the Center Model and its sub-components are shown in Figure 4-5 and are described in the paragraphs below:

4.2.3.1 Entity (devices)

All entities for a center are grouped within the Entity (devices) sub-components. Its sub-components include the following:

Entity Manager – manages the entities defined for the system and provides external access to the inventory and entity status.

Entity – represents the actual devices or information objects associated with the center. Entities may include CCTV, DMS, HAR, Events and other Center related elements which must be emulated by the RI.

Parameter – represents the particular fields that are associated with an entity.

Entity Namer (1104) – provides a naming service for the Center's entities.

4.2.3.2 Behavioral

The Behavioral sub-component maintains behavioral information about the center. The sub-components covered by this component include Connection, Authorization, and Event Model. Each sub-component is described below:

Connection – establishes and maintains state information about a centers C2C connection with the SUT.

Authentication – establishes, maintains and models any authentication methods to be utilized between the center and SUT.

Event Model – models the management of events

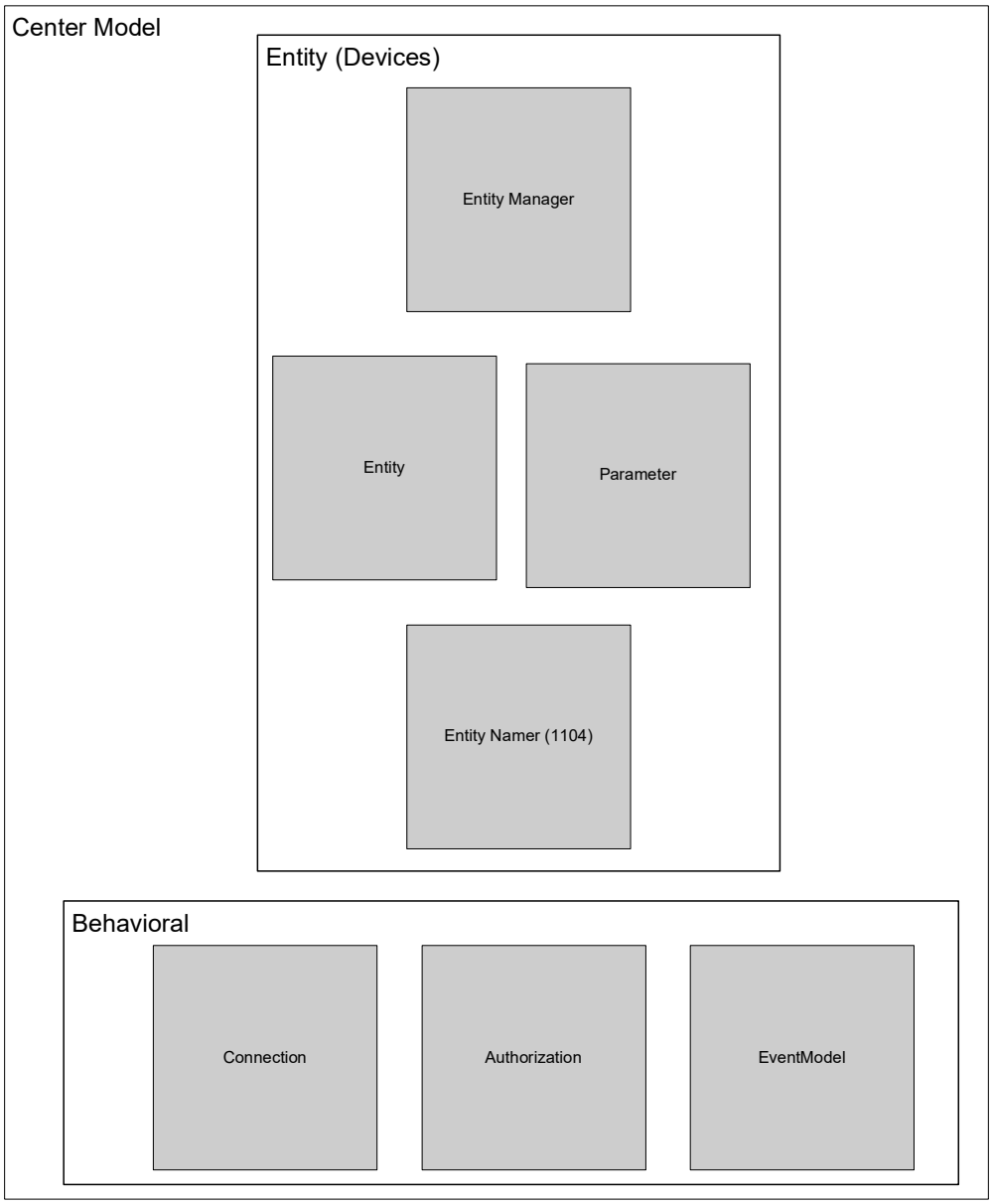

Figure 4-6: Center Model Component

4.2.4 Message Manager

The Message Manager component is responsible for creating, validating and distributing all RI-SUT messages within the RI system. It sends and receives messages to the SUT through the C2C Interface component. The Message Manager works with the Test Model to provide access to messages that have been sent and received. It also forwards all of these messages to the Logger component to be stored in the test log. The sub-components of the Message Manager are shown in Figure 4-7 and include the C2C Message Manager, Validator, Parser, Distributer and Creator. Each sub-component is described below:

C2C Message Manager – manages the flow of messages received from the SUT. It also responds to requests from the Test Model to generate messages for transmission to the SUT.

Validator – all messages received from a SUT are validated against the appropriate standard schemas defined for that test.

Parser – all messages received from the SUT will be run through a parser to ensure that they are properly encoded and to place them in an appropriate format for reference by the Test Model and Test Center components.

Distributer – manages the distribution of messages between the test model, the logger and the SUT.

Creator – provides a mechanism to create the messages defined within the standards schema defined for the test.

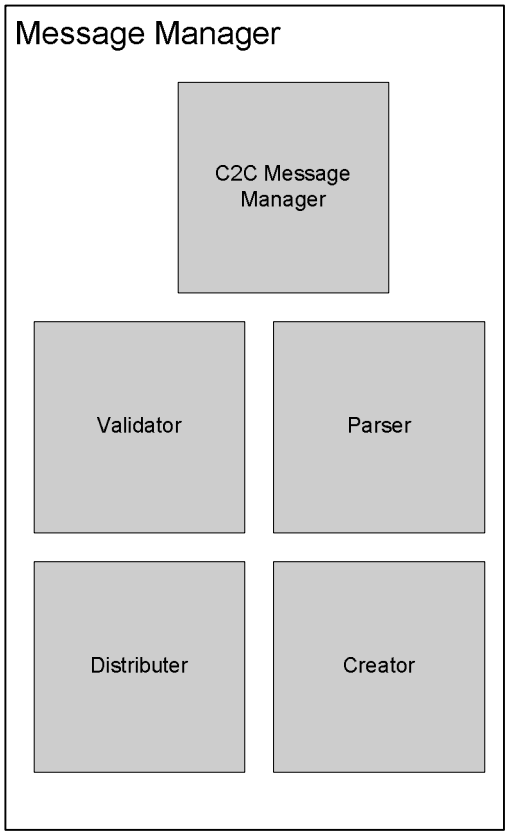

Figure 4-7: Message Manager Component

4.2.5 Logger

The Logger component consists of all functions necessary to create, store, delete and read test logs. It interfaces with all other RI components which require message or event logging. It is the primary source for the generation of test reports by the reporter component. It consists of the sub-components shown in Figure 4-8 and listed below:

Log Manager – responsible for managing the creation, deletion and storing of RI logs.

Logger – responsible for receiving the RI logging events received from other RI components.

Handler – responsible for storing the received logging events in a log file, providing a capability to turn logging on and off, and also forwarding events to the operator GUI.

Reader – responsible for reading the log events from the log file for exporting and report generation.

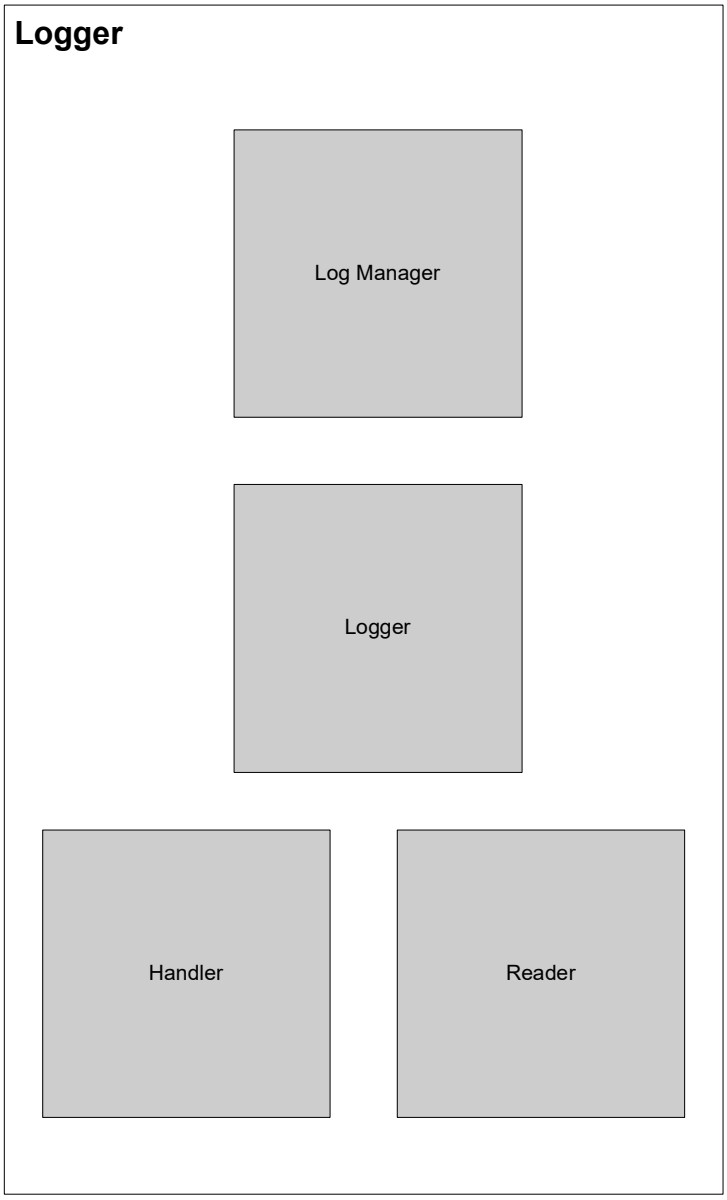

Figure 4-8: Logger sub-components

4.2.6 Reporter

The Reporter component creates user specified reports from the data held in the test log or test configuration as appropriate. It interfaces with the Logger to retrieve the actual log events necessary to create the report. It interfaces with the Test Model to retrieve test configuration settings for test configuration reports. The GUI provides the settings used to determine what type of report needs to be generated. The sub-components of Reporter are shown in Figure 4-9 and described below:

Report Manager – provides the means to specify the particular report to be generated and any special parameters.

TestConfig Report – generates a report based on the specified test configuration.

PassFail Report – provides a report of a test specifying the pass and fail status of test cases related to C2C standards.

Test Activity Report – provides a report of a test detailing all user action, test sequence and message activity during a test.

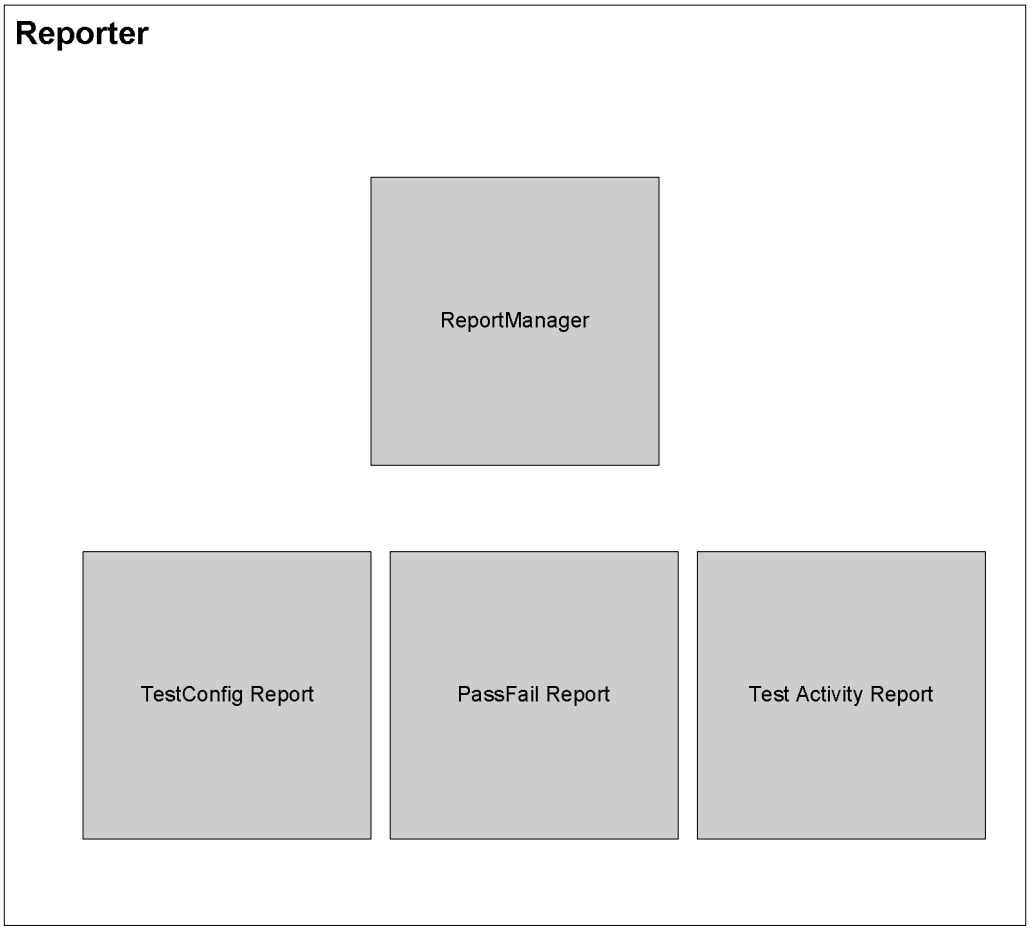

Figure 4-9: Reporter Component

4.2.7 C2C SUT Interface

The C2C SUT Interface component provides the RI's communications interface with the SUT. It receives messages from the Message Manager Component to send to the SUT, and forwards messages received from the SUT to the Message Manager. All messages sent or received over the SUT interface are forwarded to the Logger component in order to include the raw messages in the test log. The sub-components of the C2C SUT interface are shown in Figure 4-10 and are described below:

4.2.7.1 Center Services

When the RI is operating as an Owner Center the Center Services sub-component establishes itself as a server for receiving and responding to messages from the External Center (SUT). As necessary, this sub-component will manage establishing the interface consistent with the test configuration definition.

4.2.7.2 Client Services

When the RI is operating as an External Center, the Center Services sub-component will establish itself as a client for requesting and receiving messages from the Owner Center (SUT). As necessary, this sub-component will manage establishing the interface consistent with the test configuration definition.

4.2.7.3 XML/SOAP Support

The XML/SOAP Support sub-component provides the RI's support for NTCIP 2306 and consists of the following sub-components:

NTCIP 2306 – manages the interface to ensure its compliance with the NTCIP 2306 standard.

WSDL – establishes and publishes the WSDL when acting as an Owner Center. To confirm NTCIP 2306 compliance, this sub-component provides the ability to verify that the SUT publishes a WSDL consistent with the test configuration schema when the RI is acting as an External Center.

SOAP – provides the means to process SOAP encoded messages.

XML – provides the means to process XML encoded messages.

HTTP – provides the capability to transmit messages using the HTTP and HTTPs protocols.

4.2.7.4 DATEX-ASN Support (Deprecated)

The DATEX-ASN Support sub-component provides the RI's support for NTCIP 2304 and consists of the following sub-components:

NTCIP 2304 – manages the interface to ensure its compliance with the NTCIP 2304 standard, including encoding rules.

ASN.1 – provides the means to process ASN.1 encoded messages.

4.2.7.5 UDP/IP

The UDP/IP sub-component provides RI's support for supporting NTCIP 2304 using the UDP/IP transport.

4.2.7.6 TCP/IP

The TCP/IP sub-component provides RI's support for supporting NTCIP 2304 and NTCIP 2306 using the TCP/IP transport.

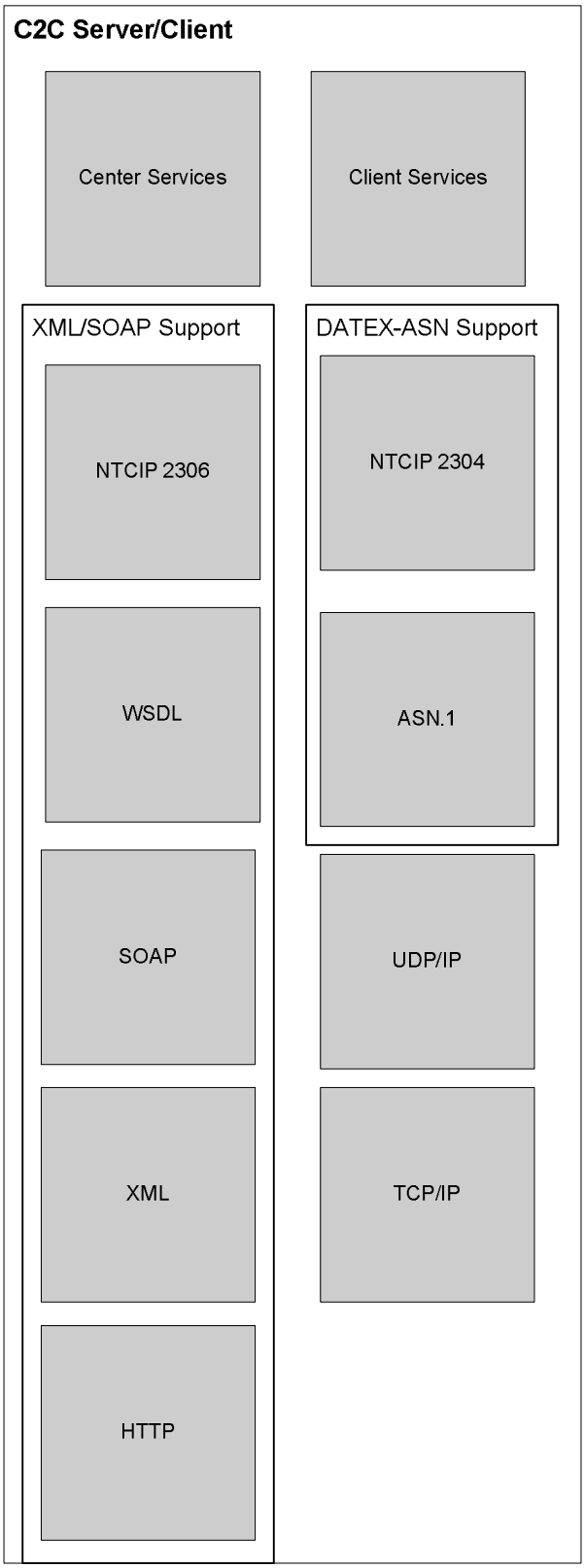

Figure 4-10: C2C SUT Interface Component

4.3 RI Interfaces

In order for the RI software to perform its functions the following additional interfaces have been defined:

- User – The RI provides control and configuration functionality to the RI via the GUI component.
- SUT – The RI exchanges messages with the SUT according to the interface specification defined in the test configuration.
- Configuration File – the RI maintains test information within Test Configuration Files.
- Reports – The RI produces reports in PDF, HTML and CSV formats.
- Test Log – The test actions and messages are stored within the RI's Test Log.
- Test Suite – a collection of test scripts, test data and other parameters that are used by the RI Test Engine in performing conformance testing for a particular standard.

5 DETAILED DESIGN

5.1 GUI

5.1.1 Main UI

5.1.1.1 Primary Presentation

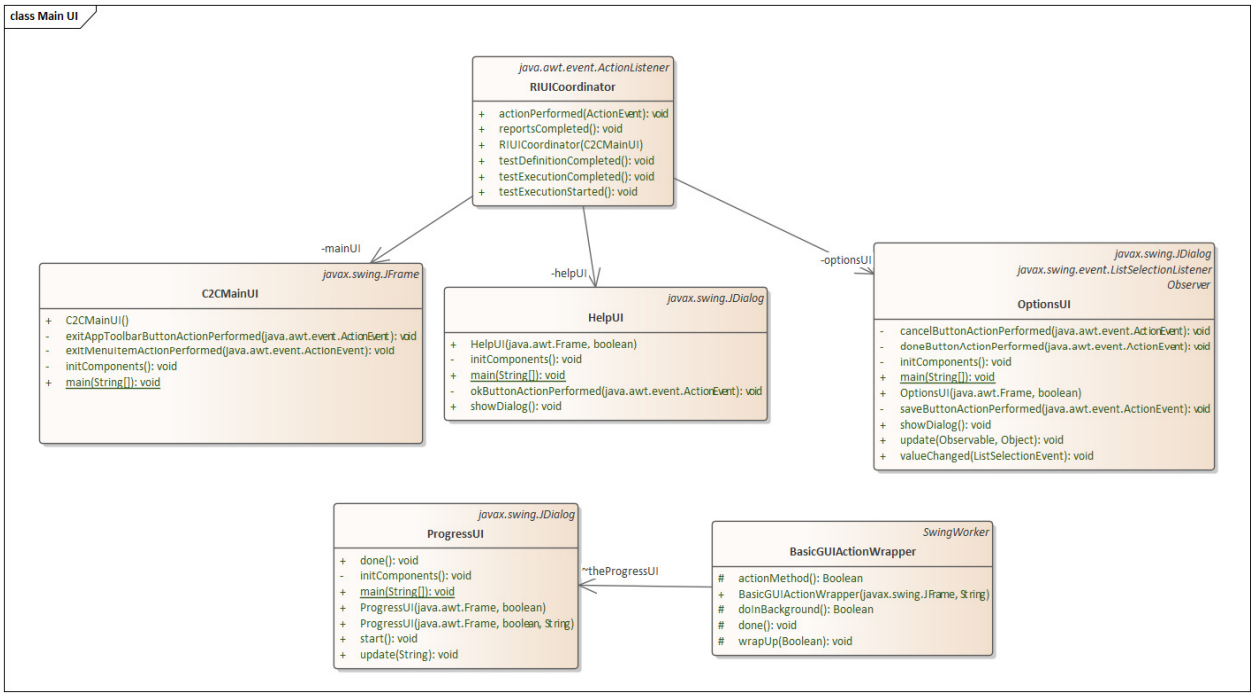

Figure 5-1 Main UI Modules

The Main UI provides the primary application frame and functions which provide access to the various C2C RI user features.

5.1.1.2 Element Catalog

5.1.1.2.1 C2CMainUI

Type: **Class** javax.swing.JFrame *Package:* gui

Provides the main user interface view for the C2C RI application. Its main method acts is the main method for the C2CRI application.

Connections

Connector	Source	Target	Notes
Association	Public	Private mainUI	

Connector	Source	Target	Notes
Source -> Destination	ReportsCoordinator	C2CMainUI	
Realization SRS36.3 to C2CMainUI Source -> Destination	Public SRS36.3	Public C2CMainUI	
Realization Source -> Destination	Public SRS3.1	Public C2CMainUI	
Realization SRS1 to C2CMainUI Source -> Destination	Public SRS1	Public C2CMainUI	
Realization SRS84 to C2CMainUI Source -> Destination	Public SRS84	Public C2CMainUI	
Realization SRS1.1 to C2CMainUI Source -> Destination	Public SRS1.1	Public C2CMainUI	
Realization SRS3 to C2CMainUI Source -> Destination	Public SRS3	Public C2CMainUI	
Realization Source -> Destination	Public SRS3.2	Public C2CMainUI	
Realization SRS2 to C2CMainUI Source -> Destination	Public SRS2	Public C2CMainUI	
Realization SRS1.3 to C2CMainUI Source -> Destination	Public SRS1.3	Public C2CMainUI	
Association Source -> Destination	Public TestDefinitionCoordinator	Private mainUI C2CMainUI	
Realization SRS3.4 to C2CMainUI Source -> Destination	Public SRS3.4	Public C2CMainUI	
Association Source -> Destination	Public RIUICoordinator	Private mainUI C2CMainUI	
Association Source -> Destination	Public TestExecutionCoordinator	Private mainUI C2CMainUI	

Attributes

Attribute	Notes	Constraints and tags
aboutMenuItem javax.swing.JMenuItem Protected	Variables declaration - do not modify//GEN-BEGIN:variables	*Default:*
c2criMainPanel javax.swing.JPanel Protected		*Default:*
createNewConfigToolbarButton javax.swing.JButton Protected		*Default:*
editorScrollPane javax.swing.JScrollPane Protected Static		*Default:*
executeMenuItem javax.swing.JMenuItem Protected		*Default:*
executeTestToolbarButton javax.swing.JButton Protected		*Default:*
exitAppToolbarButton javax.swing.JButton Protected		*Default:*
exitMenuItem javax.swing.JMenuItem Protected		*Default:*
fileMenu javax.swing.JMenu Protected		*Default:*
helpAboutToolbarButton javax.swing.JButton Protected		*Default:*

Attribute	Notes	Constraints and tags
helpMenu javax.swing.JMenu Protected		*Default:*
jEditorPane1 javax.swing.JEditorPane Protected		*Default:*
jToolBar1 javax.swing.JToolBar Private		*Default:*
mainMenuBar javax.swing.JMenuBar Private		*Default:*
maintenanceMenuItem javax.swing.JMenuItem Protected		*Default:*
newMenuItem javax.swing.JMenuItem Protected		*Default:*
openExistingToolbarButton javax.swing.JButton Protected		*Default:*
openMenuItem javax.swing.JMenuItem Protected		*Default:*
optionsMenuItem javax.swing.JMenuItem Protected		*Default:*
reportsMenuItem javax.swing.JMenuItem Protected		*Default:*

Attribute	Notes	Constraints and tags
reportsToolbarButton javax.swing.JButton Protected		*Default:*
riOptionsToolbarButton javax.swing.JButton Protected		*Default:*
toolsMenu javax.swing.JMenu Protected		*Default:*

Operations

Method	Notes	Parameters
C2CMainUI() Public	Creates new JFrame C2CMainUI, then initiates the loading of the available Test Suites. Test suites represent the standards and associated test cases/procedures necessary to verify their conformance. Finally, pass this C2CMainUI to a new instance of the RIUICoordinator, which then acts as a controller for all components that are attached to this frame.	
exitAppToolbarButtonActionPerformed() void Private	Exit app toolbar button action performed. Pre-Conditions: N/A Post-Conditions: N/A	java.awt.event.ActionEvent [in] evt the evt
exitMenuItemActionPerformed() void Private	Callback method for when a user selects file exit. It calls the JFrame.dispose() method and then calls System.exit method.	java.awt.event.ActionEvent [in] evt - the event that triggered this callback
initComponents() void Private	This method is called from within the constructor to initialize the C2CMainUI components. WARNING: Do NOT modify this code. The content of this method is always regenerated by the Form Editor.	
Static **main()** void Public	This is the main method for the RI application. It creates an instance of C2CMainUI Class and displays it by invoking the C2CMainUI setVisible method as part of the java.awt.Eventqueue dispatch thread.	String[] [in] args the command line arguments

Method	Notes	Parameters

5.1.1.2.2 RIUICoordinator

Type: **Class** java.awt.event.ActionListener *Package:* gui

The Class RIUICoordinator is the primary User Interface class for the RI. It handles events that occur on the main menu and toolbar. It delegates control to other coordinators depending on the state of the UI and user actions.

Connections

Connector	Source	Target	Notes
Realization	Public	Public	
Source -> Destination	RIUICoordinator	TestExecutionListener	
Association	Public	Private optionsUI	
Source -> Destination	RIUICoordinator	OptionsUI	
Association	Public	Private reportsPanel	
Source -> Destination	RIUICoordinator	TestReportsPanel	
Association	Public	Private executionPanel	
Source -> Destination	RIUICoordinator	TestExecutionPanel	
Association	Public	Private definitionUICoordinator	
Source -> Destination	RIUICoordinator	TestDefinitionCoordinator	
Association	Public	Private helpUI	
Source -> Destination	RIUICoordinator	HelpUI	
Realization	Public	Public	
Source -> Destination	RIUICoordinator	TestDefinitionListener	
Association	Public	Private reportsUICoordinator	
Source -> Destination	RIUICoordinator	ReportsCoordinator	
Realization	Public	Public	
Source -> Destination	RIUICoordinator	ReportsListener	
Association	Public	Private corePanel	
Source -> Destination	RIUICoordinator	ConfigurationBasePanel	
Association	Public	Private executionUICoordinator	
Source -> Destination	RIUICoordinator	TestExecutionCoordinator	
Association	Public	Private mainUI	

Connector	Source	Target	Notes
Source -> Destination	RIUICoordinator	C2CMainUI	
Association	Public	Private maintenanceUI	
Source -> Destination	RIUICoordinator	MaintenanceUI	
Realization	Public	Public	
Source -> Destination	SRS3.1	RIUICoordinator	
Realization SRS84 to RIUICoordinator	Public	Public	
Source -> Destination	SRS84	RIUICoordinator	
Realization SRS48 to RIUICoordinator	Public	Public	
Source -> Destination	SRS48	RIUICoordinator	
Realization	Public	Public	
Source -> Destination	SRS3.2	RIUICoordinator	
Realization SRS3 to RIUICoordinator	Public	Public	
Source -> Destination	SRS3	RIUICoordinator	

Attributes

Attribute	Notes	Constraints and tags
c2criMainPanelCL CardLayout Private Static	The c2cri main panel cl.	*Default:*
corePanel ConfigurationBasePanel Private Static	The core panel.	*Default:*
definitionUICoordinator TestDefinitionCoordinator Private	The definition ui coordinator.	*Default:*
executionPanel TestExecutionPanel Private Static	The execution panel.	*Default:*
executionUICoordinator TestExecutionCoordinator Private	The execution ui coordinator.	*Default:*

Attribute	Notes	Constraints and tags
helpUI HelpUI Private	The help ui.	*Default:*
maintenanceUI MaintenanceUI Private	The maintenance ui.	*Default:*
mainUI C2CMainUI Private	The main ui.	*Default:*
optionsUI OptionsUI Private	The options ui.	*Default:*
reportsPanel TestReportsPanel Private Static	The reports panel.	*Default:*
reportsUICoordinator ReportsCoordinator Private	The reports ui coordinator.	*Default:*
RI_DEFINE_STATE_NAME String Protected Static	The state of the RI UI Coordinator while defining a test configuration.	*Default:* "Defining"
RI_EXECUTE_STATE_NAME String Protected Static	The state of the RI UI Coordinator while executing a test.	*Default:* "Executing"
RI_INIT_STATE_NAME String Protected Static	The initial default state of the RI UI Coordinator.	*Default:* "Initial"
RI_MAINTENANCE_STATE_NAME String Protected Static	The state of the RI UI Coordinator while working with system maintenance.	*Default:* "Maintenance"

Attribute	Notes	Constraints and tags
RI_OPTIONS_STATE_NAME String Protected Static	The state of the RI UI Coordinator while working with system options.	*Default:* "Options"
RI_REPORT_STATE_NAME String Protected Static	The state of the RI UI Coordinator while creating reports.	*Default:* "Reporting"
riUIState String Private	The ri ui state.	*Default:*

Operations

Method	Notes	Parameters
actionPerformed() void Public	(non-Javadoc) @see java.awt.event.ActionListener#actionPerformed(java.awt.event.ActionEvent)	ActionEvent [in] e
reportsCompleted() void Public	Method called to indicate that the user has completed report creation. Return the state to initialization. *	
RIUICoordinator() Public	Constructor for the RIUI Coordinator.	C2CMainUI [in] parentFrame - reference to the C2CMainUI
testDefinitionCompleted() void Public	Method called to indicate that the user has completed test definition. Return the state to initialization.	
testExecutionCompleted() void Public	Method called to indicate that the user has completed test execution. Return the state to initialization.	
testExecutionStarted() void Public	Method called to indicate that the user has running a test. Disable the main UI menu items and disable the defaultCloseOperation for the main window. *	

5.1.1.2.3 BasicGUIActionWrapper

Type: **Class** SwingWorker *Package:* gui

BasicGUIActionWrapper is used to bundle actions that are triggered by user interface events that might take some time to process. It implements the functions of the SwingWorker, but also adds the use of a ProgressMonitor object to display progress to the user when an action takes more than a pre-defined amount of time (specified within ProgressMonitor to complete. Users of this class will typically override the actionMethod and wrapUp methods.

Connections

Connector	Source	Target	Notes
Association	Public	Package theProgressUI	
Source -> Destination	BasicGUIActionWrapper	ProgressUI	
Association	Public	Package monitor	
Source -> Destination	BasicGUIActionWrapper	ProgressMonitor	

Attributes

Attribute	Notes	Constraints and tags
monitor ProgressMonitor Package	The monitor.	*Default:*
parentFrame javax.swing.JFrame Package	The parent frame.	*Default:*
theActionName String Package	The action name.	*Default:*
theProgressUI ProgressUI Package	The progress ui.	*Default:*
TIMEOUTINSECS int Private Static	The timeoutinsecs.	*Default:* 2

Operations

Method	Notes	Parameters
actionMethod() Boolean Protected	This method is usually overridden by users of this class, and contains the actions that are to be performed in response to a user interface event. @return - indicator of whether the	

Method	Notes	Parameters
	actionMethod completed successfully.	
BasicGUIActionWrapper() Public	Constructor for the BasicGUIActionWrapper.	javax.swing.JFrame [in] parentFrame - the parent JFrame that this progressUI will be associated with.String [in] actionName - the name provided for this action.
Const **doInBackground()** Boolean Protected	(non-Javadoc) @see javax.swing.SwingWorker#doInBackground()	
Const **done()** void Protected	(non-Javadoc) @see javax.swing.SwingWorker#done()	
wrapUp() void Protected	This method sometimes overridden by users of this class, and contains close out steps for the Progress UI monitor.	Boolean [in] result - indicates whether the action completed successfully.

5.1.1.2.4 ProgressUI

Type: **Class** javax.swing.JDialog *Package:* gui

The Class ProgressUI.

Connections

Connector	Source	Target	Notes
Realization	Public	Public	
Source -> Destination	ProgressUI	ProgressReporter	
Realization SRS61.2 to ProgressUI Source -> Destination	Public SRS61.2	Public ProgressUI	
Association	Public	Package theProgressUI	
Source -> Destination	BasicGUIActionWrapper	ProgressUI	

Connector	Source	Target	Notes
Realization SRS52 to ProgressUI Source -> Destination	Public SRS52	Public ProgressUI	
Association Source -> Destination	Public ReturnObjectGUIAction Wrapper	Package theProgressUI ProgressUI	
Realization SRS61.1 to ProgressUI Source -> Destination	Public SRS61.1	Public ProgressUI	
Realization SRS61 to ProgressUI Source -> Destination	Public SRS61	Public ProgressUI	
Association Source -> Destination	Public BasicGUIActionWrappe r	Package theProgressUI ProgressUI	

Attributes

Attribute	Notes	Constraints and tags
jProgressBar1 javax.swing.JProgressBar Private	The j progress bar1.	*Default:*
operationProgressLabel javax.swing.JLabel Private	The operation progress label.	*Default:*
timer Timer Package	the timer that sets off the dialog message	*Default:*
totalDelay int Private Static	The total delay.	*Default:* 500

Operations

Method	Notes	Parameters
done() void Public	Done. Pre-Conditions: N/A Post-Conditions: N/A	
initComponents() void	This method is called from within the constructor to initialize the form.	

Method	Notes	Parameters
Private	WARNING: Do NOT modify this code. The content of this method is always regenerated by the Form Editor.	
Static **main()** void Public	The main method. Pre-Conditions: N/A Post-Conditions: N/A	String[] [in] args the command line arguments
ProgressUI() Public	Creates new form ProgressUI.	java.awt.Frame [in] parent the parentboolean [in] modal the modal
ProgressUI() Public	Creates new form ProgressUI.	java.awt.Frame [in] parent the parentboolean [in] modal the modalString [in] actionTitle the action title
start() void Public	Start. Pre-Conditions: N/A Post-Conditions: N/A	
update() void Public	Update. Pre-Conditions: N/A Post-Conditions: N/A	String [in] updateText the update text

5.1.1.2.5 HelpUI

Type: **Class** javax.swing.JDialog *Package:* gui

The Help about dialog box.

Connections

Connector	Source	Target	Notes
Realization SRS1.3 to HelpUI Source -> Destination	Public SRS1.3	Public HelpUI	
Association Source -> Destination	Public RIUICoordinator	Private helpUI HelpUI	
Realization SRS1.2 to HelpUI Source -> Destination	Public SRS1.2	Public HelpUI	

Attributes

Attribute	Notes	Constraints and tags

Attribute	Notes	Constraints and tags
contactInfoLabel javax.swing.JLabel Private	Variables declaration - do not modify//GEN-BEGIN:variables	*Default:*
okButton javax.swing.JButton Private		*Default:*
riVersionLabel javax.swing.JLabel Public		*Default:*

Operations

Method	Notes	Parameters
HelpUI() Public	Creates new form HelpUI.	java.awt.Frame [in] parent the parent frame for this objectboolean [in] modal indicates whether the dialog should be modal
initComponents() void Private	This method is called from within the constructor to initialize the form. WARNING: Do NOT modify this code. The content of this method is always regenerated by the Form Editor.	
Static **main()** void Public	The main method. Pre-Conditions: N/A Post-Conditions: N/A	String[] [in] args the command line arguments
okButtonActionPerformed() void Private	Ok button action performed. Pre-Conditions: N/A Post-Conditions: N/A	java.awt.event.ActionEvent [in] evt the evt
showDialog() void Public	Makes the HelpUI dialog visible to the operator.	

5.1.1.2.6 OptionsUI

Type: **Class** javax.swing.JDialog, javax.swing.event.ListSelectionListener, Observer *Package:* gui

The Class OptionsUI presents the c2c ri options dialog.

Connections

Connector	Source	Target	Notes
Association Source -> Destination	Public OptionsUI	Private ePanel PropertyEditorPanel	
Realization SRS26 to OptionsUI Source -> Destination	Public SRS26	Public OptionsUI	
Association Source -> Destination	Public RIUICoordinator	Private optionsUI OptionsUI	
Realization SRS23 to OptionsUI Source -> Destination	Public SRS23	Public OptionsUI	
Realization SRS1.3 to OptionsUI Source -> Destination	Public SRS1.3	Public OptionsUI	
Realization SRS1.2 to OptionsUI Source -> Destination	Public SRS1.2	Public OptionsUI	

Attributes

Attribute	Notes	Constraints and tags
cancelButton javax.swing.JButton Protected		*Default:*
cPane javax.swing.JPanel Private	Variables declaration - do not modify//GEN-BEGIN:variables	*Default:*
docEditorPane JEditorPane Public Static	The doc editor pane.	*Default:*
doneButton javax.swing.JButton Private		*Default:*
ePanel org.fhwa.c2cri.gui.propertyeditor.PropertyEditorPanel Private	The e panel.	*Default:*

Attribute	Notes	Constraints and tags
jPanel1 javax.swing.JPanel Private		*Default:*
jTp JTabbedPane Private Static	The j tp.	*Default:*
parametersModified boolean Private	The parameters modified.	*Default:* false
riParametersLabel javax.swing.JLabel Private		*Default:*
saveButton javax.swing.JButton Protected		*Default:*
tables JTable Public Static	The tables.	*Default:*

Operations

Method	Notes	Parameters
cancelButtonActionPerformed() void Private	Cancel button action performed. Pre-Conditions: N/A Post-Conditions: N/A	java.awt.event.ActionEvent [in] evt the evt
doneButtonActionPerformed() void Private	Done button action performed. Pre-Conditions: N/A Post-Conditions: N/A	java.awt.event.ActionEvent [in] evt the evt
initComponents() void Private	This method is called from within the constructor to initialize the form. WARNING: Do NOT modify this code. The content of this method is always regenerated by the Form Editor.	
Static **main()** void Public	The main method. Pre-Conditions: N/A Post-Conditions: N/A	String[] [in] args the command line arguments
OptionsUI()	Creates new form OptionsUI.	java.awt.Frame [in] parent

Method	Notes	Parameters
Public		the parentboolean [in] modal the modal
saveButtonActionPerformed() void Private	Save button action performed. Pre-Conditions: N/A Post-Conditions: N/A	java.awt.event.ActionEvent [in] evt the evt
showDialog() void Public	make the optionsUI dialog visible.	
update() void Public	(non-Javadoc) @see java.util.Observer#update(java.util.Observable, java.lang.Object)	Observable [in] o Object [in] arg
valueChanged() void Public	(non-Javadoc) @see javax.swing.event.ListSelectionListener#valueChanged(javax.swing.event.ListSelectionEvent)	ListSelectionEvent [in] e

5.1.1.3 Variability Guide

The C2C RI GUI utilizes the swing library included within Java. Since the Java Virtual Machine is capable of being used in both Windows and Linux environments, it provides a path for the program to be deployed in both environments.

5.1.1.4 Rationale

The modules within this section are intended to address requirements related to the user features. Additionally, the implementation needs to also satisfy the Section508 requirements allocated to the user interface.

5.1.1.5 Context Diagram

N/A

5.1.2 Test Definition GUI

5.1.2.1 Primary Presentation

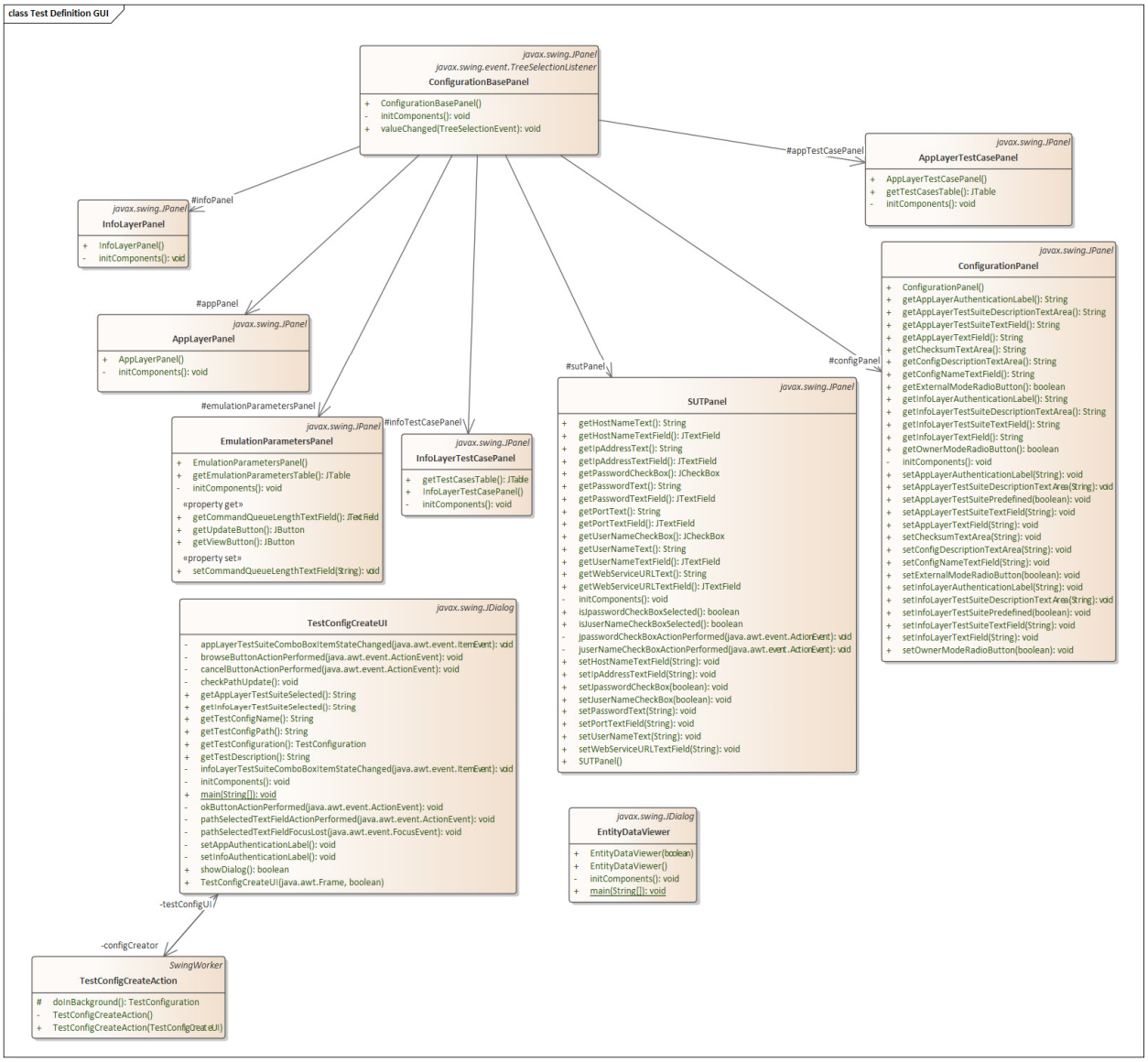

Figure 5-2 Test Defiinition GUI Modules

This module provides access to user features necessary for creating and editing Test Configuration files.

5.1.2.2 Element Catalog

5.1.2.2.1 ConfigurationBasePanel

Type: **Class** javax.swing.JPanel, javax.swing.event.TreeSelectionListener *Package:* gui

Main Panel for managing the configuration settings. It serves as a container for other configuration definition tab panels. It also contains the Save, Ok, and Cancel button controls.

Connections

Connector	Source	Target	Notes
Association	Public	Protected appPanel	
Source -> Destination	ConfigurationBasePanel	AppLayerPanel	
Association	Public	Protected infoTestCasePanel	
Source -> Destination	ConfigurationBasePanel	InfoLayerTestCasePanel	
Association	Public	Protected configPanel	
Source -> Destination	ConfigurationBasePanel	ConfigurationPanel	
Association	Public	Protected appTestCasePanel	
Source -> Destination	ConfigurationBasePanel	AppLayerTestCasePanel	
Association	Public	Protected infoPanel	
Source -> Destination	ConfigurationBasePanel	InfoLayerPanel	
Association	Public	Protected sutPanel	
Source -> Destination	ConfigurationBasePanel	SUTPanel	
Realization	Public	Public	
Source -> Destination	SRS3.1	ConfigurationBasePanel	
Realization SRS3 to ConfigurationBasePanel	Public	Public	
Source -> Destination	SRS3	ConfigurationBasePanel	
Realization SRS12.2 to ConfigurationBasePanel	Public	Public	
Source -> Destination	SRS12.2	ConfigurationBasePanel	
Realization SRS16.8 to ConfigurationBasePanel	Public	Public	
Source -> Destination	SRS16.8	ConfigurationBasePanel	
Realization SRS1.3 to ConfigurationBasePanel	Public	Public	
Source -> Destination	SRS1.3	ConfigurationBasePanel	
Association	Public	Private corePanel	
Source -> Destination	TestDefinitionCoordinator	ConfigurationBasePanel	
Realization SRS16 to ConfigurationBasePanel	Public	Public	
Source -> Destination	SRS16	ConfigurationBasePanel	
Realization	Public	Public	

Connector	Source	Target	Notes
Source -> Destination	SRS3.2	ConfigurationBasePanel	
Association	Public	Private corePanel	
Source -> Destination	RIUICoordinator	ConfigurationBasePanel	
Realization SRS11.3 to ConfigurationBasePanel	Public	Public	
Source -> Destination	SRS11.3	ConfigurationBasePanel	
Realization SRS32.7 to ConfigurationBasePanel	Public	Public	
Source -> Destination	SRS32.7	ConfigurationBasePanel	
Realization SRS1.2 to ConfigurationBasePanel	Public	Public	
Source -> Destination	SRS1.2	ConfigurationBasePanel	
Realization SRS16.9 to ConfigurationBasePanel	Public	Public	
Source -> Destination	SRS16.9	ConfigurationBasePanel	

Attributes

Attribute	Notes	Constraints and tags
appPanel AppLayerPanel Protected	the AppLayerPanel tab.	*Default:*
appTestCasePanel AppLayerTestCasePanel Protected	the AppLayer Test Case Panel tab.	*Default:*
cancelButton javax.swing.JButton Protected	</editor-fold>//GEN-END:initComponents Variables declaration - do not modify//GEN-BEGIN:variables	*Default:*
configPanel ConfigurationPanel Protected	the configuration panel tab.	*Default:*
configurationSubPanel javax.swing.JTabbedPane Protected		*Default:*
configurationTree javax.swing.JTree		*Default:*

Attribute	Notes	Constraints and tags
Protected		
infoPanel InfoLayerPanel Protected	the InfoLayerPanel tab.	*Default:*
infoTestCasePanel InfoLayerTestCasePanel Protected	the InfoLayer Test Case Panel tab.	*Default:*
jPanel1 javax.swing.JPanel Private		*Default:*
jScrollPane1 javax.swing.JScrollPane Private		*Default:*
jSplitPane1 javax.swing.JSplitPane Private		*Default:*
navigationPane javax.swing.JScrollPane Protected		*Default:*
okButton javax.swing.JButton Protected		*Default:*
saveAsButton javax.swing.JButton Protected		*Default:*
saveButton javax.swing.JButton Protected		*Default:*
sutPanel SUTPanel Protected	the SUT panel tab.	*Default:*

Attribute	Notes	Constraints and tags
validateConfigurationButton javax.swing.JButton Protected		*Default:*

Operations

Method	Notes	Parameters
ConfigurationBasePanel() Public	Creates new form ConfigurationBasePanel. First initialize the ConfigurationBasePanel. Then create new instances of the config, log, sut, info and app panels. Add the new panels to the ConfigurationBasePanel subPanel as Tabs. Set the SubPanel to be visible, but make teh ConfigurationBasePanel invisible.	
initComponents() void Private	This method is called from within the constructor to initialize the panel. WARNING: Do NOT modify this code. The content of this method is always regenerated by the Form Editor.	
valueChanged() void Public	(non-Javadoc) @see javax.swing.event.TreeSelectionListener#valueChanged(javax.swing.event.TreeSelectionEvent)	TreeSelectionEvent [in] e

5.1.2.2.2 AppLayerPanel

Type: **Class** javax.swing.JPanel *Package:* gui

The panel which allows the user to set Application Layer RTM and parameter settings. This class provides a view of the panel. Control of the panel is handled by a seperate controlling (coordinator) class.

Connections

Connector	Source	Target	Notes
Association Source -> Destination	Public ConfigurationBasePanel	Protected appPanel AppLayerPanel	
Realization SRS2 to AppLayerPanel Source -> Destination	Public SRS2	Public AppLayerPanel	

Connector	Source	Target	Notes
Realization SRS18 to AppLayerPanel Source -> Destination	Public SRS18	Public AppLayerPanel	
Realization SRS18.2 to AppLayerPanel Source -> Destination	Public SRS18.2	Public AppLayerPanel	

Attributes

Attribute	Notes	Constraints and tags
jPanel3 javax.swing.JPanel Private	</editor-fold>//GEN-END:initComponents Variables declaration - do not modify//GEN-BEGIN:variables	*Default:*
jPanel4 javax.swing.JPanel Private		*Default:*
needsClearjButton javax.swing.JButton Protected		*Default:*
needsPanel javax.swing.JPanel Private		*Default:*
needsScrollPane javax.swing.JScrollPane Private		*Default:*
parametersPanel javax.swing.JPanel Private		*Default:*
parametersScrollPane javax.swing.JScrollPane Protected		*Default:*
parametersTable javax.swing.JTable Protected		*Default:*

Attribute	Notes	Constraints and tags
requirementsClearButton javax.swing.JButton Protected		*Default:*
requirementsPanel javax.swing.JPanel Private		*Default:*
requirementsScrollPane javax.swing.JScrollPane Private		*Default:*
standardNeedsTable javax.swing.JTable Protected		*Default:*
standardRequirementsTable javax.swing.JTable Protected		*Default:*

Operations

Method	Notes	Parameters
AppLayerPanel() Public	Creates new JPanel AppLayerPanel by calling initComponents.	
initComponents() void Private	This method is called from within the constructor to define and initialize all elements of the panel. WARNING: Do NOT manually modify this code. The content of this method is always regenerated by the Form Editor.	

5.1.2.2.3 AppLayerTestCasePanel

Type: **Class** javax.swing.JPanel *Package:* gui

The panel used by the user to specify Information Level parameters to be used for a test. This class provides the view of the panel only. Control of the panel is handled by an external coordinator (control) class.

Connections

Connector	Source	Target	Notes
Realization SRS16.1 to AppLayerTestCasePanel Source -> Destination	Public SRS16.1	Public AppLayerTestCasePanel	
Association Source -> Destination	Public ConfigurationBasePanel	Protected appTestCasePanel AppLayerTestCasePanel	
Realization SRS2 to AppLayerTestCasePanel Source -> Destination	Public SRS2	Public AppLayerTestCasePanel	

Attributes

Attribute	Notes	Constraints and tags
selectNeedsScrollPane javax.swing.JScrollPane Private	Variables declaration - do not modify//GEN-BEGIN:variables	*Default:*
testCasesPanel javax.swing.JPanel Private		*Default:*
testCasesTable javax.swing.JTable Protected		*Default:*

Operations

Method	Notes	Parameters
AppLayerTestCasePanel() Public	Creates new form InfoLayerPanel by calling initComponents.	
initComponents() void Private	This method is called from within the constructor to initialize the panel. WARNING: Do NOT modify this code. The content of this method is always regenerated by the Form Editor.	

5.1.2.2.4 InfoLayerPanel

Type: **Class** javax.swing.JPanel *Package:* gui

The panel used by the user to specify Information Level parameters to be used for a test. This class provides the view of the panel only. Control of the panel is handled by an external coordinator (control) class.

Connections

Connector	Source	Target	Notes
Realization SRS16.1 to InfoLayerPanel Source -> Destination	Public SRS16.1	Public InfoLayerPanel	
Realization SRS2 to InfoLayerPanel Source -> Destination	Public SRS2	Public InfoLayerPanel	
Realization SRS11.3 to InfoLayerPanel Source -> Destination	Public SRS11.3	Public InfoLayerPanel	
Realization SRS1.3 to InfoLayerPanel Source -> Destination	Public SRS1.3	Public InfoLayerPanel	
Realization SRS18.1 to InfoLayerPanel Source -> Destination	Public SRS18.1	Public InfoLayerPanel	
Realization SRS1.2 to InfoLayerPanel Source -> Destination	Public SRS1.2	Public InfoLayerPanel	
Realization SRS18 to InfoLayerPanel Source -> Destination	Public SRS18	Public InfoLayerPanel	
Association Source -> Destination	Public ConfigurationBasePanel	Protected infoPanel InfoLayerPanel	

Attributes

Attribute	Notes	Constraints and tags
jPanel1 javax.swing.JPanel Private	Variables declaration - do not modify//GEN-BEGIN:variables	*Default:*
jPanel2 javax.swing.JPanel Private		*Default:*
needsClearOptionsButton javax.swing.JButton Protected		*Default:*

Attribute	Notes	Constraints and tags
parametersScrollPane javax.swing.JScrollPane Protected		*Default:*
parametersTable javax.swing.JTable Protected		*Default:*
requirementsClearAllButton javax.swing.JButton Protected		*Default:*
selectNeedPanel javax.swing.JPanel Private		*Default:*
selectNeedsScrollPane javax.swing.JScrollPane Private		*Default:*
selectRequirementsPanel javax.swing.JPanel Private		*Default:*
standardNeedsTable javax.swing.JTable Protected		*Default:*
standardRequirementsScroll Pane javax.swing.JScrollPane Private		*Default:*
standardRequirementsTable javax.swing.JTable Protected		*Default:*
testParametersPanel javax.swing.JPanel		*Default:*

Attribute	Notes	Constraints and tags
Private		

Operations

Method	Notes	Parameters
InfoLayerPanel() Public	Creates new form InfoLayerPanel by calling initComponents.	
initComponents() void Private	This method is called from within the constructor to initialize the panel. WARNING: Do NOT modify this code. The content of this method is always regenerated by the Form Editor.	

5.1.2.2.5 InfoLayerTestCasePanel

Type: **Class** javax.swing.JPanel *Package:* gui

The panel used by the user to specify Information Level parameters to be used for a test. This class provides the view of the panel only. Control of the panel is handled by an external coordinator (control) class.

Connections

Connector	Source	Target	Notes
Realization SRS1.3 to InfoLayerTestCasePanel Source -> Destination	Public SRS1.3	Public InfoLayerTestCasePanel	
Association Source -> Destination	Public ConfigurationBasePanel	Protected infoTestCasePanel InfoLayerTestCasePanel	
Realization SRS1.2 to InfoLayerTestCasePanel Source -> Destination	Public SRS1.2	Public InfoLayerTestCasePanel	
Realization SRS2 to InfoLayerTestCasePanel Source -> Destination	Public SRS2	Public InfoLayerTestCasePanel	
Realization SRS11.3 to InfoLayerTestCasePanel Source -> Destination	Public SRS11.3	Public InfoLayerTestCasePanel	

Attributes

Attribute	Notes	Constraints and tags
selectNeedsScrollPane javax.swing.JScrollPane Private	Variables declaration - do not modify//GEN-BEGIN:variables	*Default:*
testCasesPanel javax.swing.JPanel Private		*Default:*
testCasesTable javax.swing.JTable Protected		*Default:*

Operations

Method	Notes	Parameters
InfoLayerTestCasePanel() Public	Creates new form InfoLayerPanel by calling initComponents.	
initComponents() void Private	This method is called from within the constructor to initialize the panel. WARNING: Do NOT modify this code. The content of this method is always regenerated by the Form Editor.	

5.1.2.2.6 ConfigurationPanel

Type: **Class** javax.swing.JPanel *Package:* gui

Defines the components present on the Config Tab.

Connections

Connector	Source	Target	Notes
Realization SRS11.3 to ConfigurationPanel Source -> Destination	Public SRS11.3	Public ConfigurationPanel	
Realization SRS16.5.3 to ConfigurationPanel Source -> Destination	Public SRS16.5.3	Public ConfigurationPanel	
Realization SRS2 to ConfigurationPanel	Public	Public	

Connector	Source	Target	Notes
Source -> Destination	SRS2	ConfigurationPanel	
Association	Public	Protected configPanel	
Source -> Destination	ConfigurationBasePanel	ConfigurationPanel	
Realization SRS91 to ConfigurationPanel Source -> Destination	Public SRS91	Public ConfigurationPanel	
Realization SRS16.7 to ConfigurationPanel Source -> Destination	Public SRS16.7	Public ConfigurationPanel	
Realization SRS16.5.2 to ConfigurationPanel Source -> Destination	Public SRS16.5.2	Public ConfigurationPanel	
Realization SRS16.5.1 to ConfigurationPanel Source -> Destination	Public SRS16.5.1	Public ConfigurationPanel	
Realization SRS16.5 to ConfigurationPanel Source -> Destination	Public SRS16.5	Public ConfigurationPanel	
Realization SRS1.2 to ConfigurationPanel Source -> Destination	Public SRS1.2	Public ConfigurationPanel	
Realization SRS1.3 to ConfigurationPanel Source -> Destination	Public SRS1.3	Public ConfigurationPanel	

Attributes

Attribute	Notes	Constraints and tags
appLayerAuthenticationLabel javax.swing.JLabel Protected	Variables declaration - do not modify//GEN-BEGIN:variables	*Default:*
appLayerStandardLabel javax.swing.JLabel Private		*Default:*
appLayerStandardPanel javax.swing.JPanel Private		*Default:*

Attribute	Notes	Constraints and tags
appLayerTestSuiteDescriptionTextArea javax.swing.JTextArea Protected		*Default:*
appLayerTestSuiteTextField javax.swing.JTextField Protected		*Default:*
appLayerTextField javax.swing.JTextField Protected		*Default:*
checksumTextArea javax.swing.JTextArea Public		*Default:*
configDescriptionScrollPanel javax.swing.JScrollPane Private		*Default:*
configDescriptionTextArea javax.swing.JTextArea Protected		*Default:*
configNameTextField javax.swing.JTextField Protected		*Default:*
configPanel javax.swing.JPanel Private		*Default:*
configurationNamePanel javax.swing.JPanel Private		*Default:*
externalModeRadioButton javax.swing.JRadioButton		*Default:*

Attribute	Notes	Constraints and tags
Protected		
infoLayerAuthenticationLabel javax.swing.JLabel Protected		*Default:*
infoLayerStandardLabel javax.swing.JLabel Private		*Default:*
infoLayerStandardPanel javax.swing.JPanel Private		*Default:*
infoLayerTestSuiteDescriptionTextArea javax.swing.JTextArea Protected		*Default:*
infoLayerTestSuiteTextField javax.swing.JTextField Protected		*Default:*
infoLayerTextField javax.swing.JTextField Protected		*Default:*
jLabel1 javax.swing.JLabel Private		*Default:*
jScrollPane1 javax.swing.JScrollPane Private		*Default:*
ownerModeRadioButton javax.swing.JRadioButton Protected		*Default:*
riModeButtonGroup		*Default:*

Attribute	Notes	Constraints and tags
javax.swing.ButtonGroup Private		
riModePanel javax.swing.JPanel Private		*Default:*
testConfigDescriptionLabel javax.swing.JLabel Private		*Default:*
testConfigNameLabel javax.swing.JLabel Private		*Default:*
testSuiteDescriptionLabel javax.swing.JLabel Private		*Default:*
testSuiteDescriptionLabel1 javax.swing.JLabel Private		*Default:*
testSuiteDescriptionScrollPane javax.swing.JScrollPane Private		*Default:*
testSuiteDescriptionScrollPane1 javax.swing.JScrollPane Private		*Default:*
testSuiteNameLabel javax.swing.JLabel Private		*Default:*
testSuiteNameLabel1 javax.swing.JLabel Private		*Default:*

Attribute	Notes	Constraints and tags
testSuitePanel javax.swing.JPanel Private		*Default:*

Operations

Method	Notes	Parameters
ConfigurationPanel() Public	Creates new form ConfigurationPanel.	
initComponents() void Private	This method is called from within the constructor to initialize the panel. This method also sets the default values for the fields. WARNING: Do NOT modify this code. The content of this method is always regenerated by the Form Editor.	

5.1.2.2.7 EntityEmulationPanel

Type: **Class** javax.swing.JPanel *Package:* gui

The panel which allows the user to set Entity Emulation parameter settings. This class provides a view of the panel. Control of the panel is handled by a seperate controlling (coordinator) class.

Connections

Connector	Source	Target	Notes
Realization	Public	Public	
Source -> Destination	SRS97	EntityEmulationPanel	
Realization	Public	Public	
Source -> Destination	SRS96	EntityEmulationPanel	

Attributes

Attribute	Notes	Constraints and tags
commandQueueLengthLabel javax.swing.JLabel Private		*Default:*
commandQueueLengthTextBox javax.swing.JTextBox Protected		*Default:*

Attribute	Notes	Constraints and tags
entityDataSelectionsTable javax.swing.JTable Protected		*Default:*
entityEmulationPanel javax.swing.JPanel Private		*Default:*
entityEmulationScrollPane javax.swing.JScrollPane Private		*Default:*
jPanel3 javax.swing.JPanel Private	</editor-fold>//GEN-END:initComponents Variables declaration - do not modify//GEN-BEGIN:variables	*Default:*
jPanel4 javax.swing.JPanel Private		*Default:*
updatejButton javax.swing.JButton Protected		*Default:*
viewButton javax.swing.JButton Protected		*Default:*

Operations

Method	Notes	Parameters
AppLayerPanel() Public	Creates new JPanel AppLayerPanel by calling initComponents.	
initComponents() void Private	This method is called from within the constructor to define and initialize all elements of the panel. WARNING: Do NOT manually modify this code. The content of this method is always regenerated by the Form Editor.	

5.1.2.2.8 EntityDataViewerUI

Type: **Class** javax.swing.JDialog, javax.swing.event.ListSelectionListener *Package:* gui

The Class EntityDataViewerUI provides the user with the ability for the user to view the entity data for a selected entity data element.

Connections

Connector	Source	Target	Notes
Realization	Public	Public	
Source -> Destination	SRS96	EntityDataViewerUI	
Realization	Public	Public	
Source -> Destination	SRS104	EntityDataViewerUI	
Realization	Public	Public	
Source -> Destination	SRS97	EntityDataViewerUI	

Attributes

Attribute	Notes	Constraints and tags
EntityDataTextArea javax.swing.JTextArea Private		*Default:*
okButton javax.swing.JButton Protected		*Default:*

Operations

Method	Notes	Parameters
EntityDataViewerUI() Public	Creates new form EntityDataViewerUI.	java.awt.Frame [in] parent the parentboolean [in] modal the modal
initComponents() void Private	This method is called from within the constructor to initialize the form. WARNING: Do NOT modify this code. The content of this method is always regenerated by the Form Editor.	
Static **main()** void Public	The main method. Pre-Conditions: N/A Post-Conditions: N/A	String[] [in] args the command line arguments
okButtonActionPerformed()	OK button action performed. Pre-	java.awt.event.ActionEvent [in] evt

Method	Notes	Parameters
void Private	Conditions: N/A Post-Conditions: N/A	the evt
showDialog() boolean Public	Show dialog. Pre-Conditions: N/A Post-Conditions: N/A @return true, if successful	

5.1.2.2.9 SUTPanel

Type: **Class** javax.swing.JPanel *Package:* gui

Test Configuration Panel Class for obtaining SUT information.

Connections

Connector	Source	Target	Notes
Realization SRS2 to SUTPanel Source -> Destination	Public SRS2	Public SUTPanel	
Realization SRS16.2.6 to SUTPanel Source -> Destination	Public SRS16.2.6	Public SUTPanel	
Realization SRS16.2 to SUTPanel Source -> Destination	Public SRS16.2	Public SUTPanel	
Realization SRS1.3 to SUTPanel Source -> Destination	Public SRS1.3	Public SUTPanel	
Realization SRS16.2.2 to SUTPanel Source -> Destination	Public SRS16.2.2	Public SUTPanel	
Realization SRS16.2.1 to SUTPanel Source -> Destination	Public SRS16.2.1	Public SUTPanel	
Realization SRS16.2.3 to SUTPanel Source -> Destination	Public SRS16.2.3	Public SUTPanel	
Realization SRS16.2.4 to SUTPanel Source -> Destination	Public SRS16.2.4	Public SUTPanel	

Connector	Source	Target	Notes
Realization SRS11.3 to SUTPanel Source -> Destination	Public SRS11.3	Public SUTPanel	
Realization SRS1.2 to SUTPanel Source -> Destination	Public SRS1.2	Public SUTPanel	
Association Source -> Destination	Public ConfigurationBasePanel	Protected sutPanel SUTPanel	
Realization SRS16.2.5 to SUTPanel Source -> Destination	Public SRS16.2.5	Public SUTPanel	

Attributes

Attribute	Notes	Constraints and tags
hostNameLabel javax.swing.JLabel Private	Variables declaration - do not modify//GEN-BEGIN:variables	*Default:*
hostNameTextField javax.swing.JTextField Protected		*Default:*
ipAddressLabel javax.swing.JLabel Private		*Default:*
ipAddressTextField javax.swing.JTextField Protected		*Default:*
jpasswordCheckBox javax.swing.JCheckBox Public		*Default:*
juserNameCheckBox javax.swing.JCheckBox Public		*Default:*
passwordLabel		*Default:*

Attribute	Notes	Constraints and tags
javax.swing.JLabel Private		
passwordTextField javax.swing.JTextField Protected		*Default:*
portLabel javax.swing.JLabel Private		*Default:*
portTextField javax.swing.JTextField Protected		*Default:*
userNameLabel javax.swing.JLabel Private		*Default:*
userNameTextField javax.swing.JTextField Protected		*Default:*
webServiceURLLabel javax.swing.JLabel Private		*Default:*
webServiceURLTextField javax.swing.JTextField Protected		*Default:*

Operations

Method	Notes	Parameters
initComponents() void Private	This method is called from within the constructor to initialize the panel. WARNING: Do NOT modify this code. The content of this method is always regenerated by the Form Editor.	
jpasswordCheckBoxActionPe	Jpassword check box action performed. Pre-	java.awt.event.ActionEvent [in] evt

Method	Notes	Parameters
rformed() void Private	Conditions: N/A Post-Conditions: N/A	the evt
juserNameCheckBoxActionPerformed() void Private	Juser name check box action performed. Pre-Conditions: N/A Post-Conditions: N/A	java.awt.event.ActionEvent [in] evt the evt
SUTPanel() Public	Creates new panel SUTPanel. It calls initComponents to create all panel elements.	

5.1.2.2.10 TestConfigCreateAction

Type: **Class** SwingWorker *Package:* gui

The Class TestConfigCreateAction.

Connections

Connector	Source	Target	Notes
Association Source -> Destination	Public TestConfigCreateAction	Private testConfiguration TestConfiguration	
Association Source -> Destination	Public TestConfigCreateAction	Private testConfigUI TestConfigCreateUI	
Realization SRS10.2 to TestConfigCreateAction Source -> Destination	Public SRS10.2	Public TestConfigCreateAction	
Association Source -> Destination	Public TestConfigCreateUI	Private configCreator TestConfigCreateAction	
Realization SRS10 to TestConfigCreateAction Source -> Destination	Public SRS10	Public TestConfigCreateAction	
Realization SRS10.1 to TestConfigCreateAction Source -> Destination	Public SRS10.1	Public TestConfigCreateAction	

Attributes

Attribute	Notes	Constraints and tags
testConfigUI TestConfigCreateUI Private	The test config ui.	*Default:*

Attribute	Notes	Constraints and tags
testConfiguration TestConfiguration Private	The test configuration.	*Default:*

Operations

Method	Notes	Parameters
doInBackground() TestConfiguration Protected	(non-Javadoc) @see javax.swing.SwingWorker#doInBackground ()	
TestConfigCreateAction() Public	Instantiates a new test config create action. Pre-Conditions: N/A Post-Conditions: N/A	TestConfigCreateUI [in] testConfigUI the test config ui
TestConfigCreateAction() Private	Instantiates a new test config create action. Pre-Conditions: N/A Post-Conditions: N/A	

5.1.2.2.11 TestConfigCreateUI

Type: **Class** javax.swing.JDialog *Package:* gui

The Class TestConfigCreateUI.

Connections

Connector	Source	Target	Notes
Association Source -> Destination	Public TestConfigCreateUI	Private testConfiguration TestConfiguration	
Association Source -> Destination	Public TestConfigCreateUI	Private configCreator TestConfigCreateAction	
Association Source -> Destination	Public TestConfigCreateUI	Private reporterConfigFile ProgressReporter	
Association Source -> Destination	Public TestConfigCreateUI	Private listenerConfigFile ProgressListener	

Connector	Source	Target	Notes
Realization SRS11.2 to TestConfigCreateUI Source -> Destination	Public SRS11.2	Public TestConfigCreateUI	
Realization SRS1.3 to TestConfigCreateUI Source -> Destination	Public SRS1.3	Public TestConfigCreateUI	
Realization SRS10.4 to TestConfigCreateUI Source -> Destination	Public SRS10.4	Public TestConfigCreateUI	
Realization SRS11.1 to TestConfigCreateUI Source -> Destination	Public SRS11.1	Public TestConfigCreateUI	
Association Source -> Destination	Public TestConfigCreateAction	Private testConfigUI TestConfigCreateUI	
Realization SRS14 to TestConfigCreateUI Source -> Destination	Public SRS14	Public TestConfigCreateUI	
Realization Source -> Destination	Public SRS10.8	Public TestConfigCreateUI	
Realization SRS13.2 to TestConfigCreateUI Source -> Destination	Public SRS13.2	Public TestConfigCreateUI	
Realization Source -> Destination	Public SRS3.1	Public TestConfigCreateUI	
Realization SRS91 to TestConfigCreateUI Source -> Destination	Public SRS91	Public TestConfigCreateUI	
Realization Source -> Destination	Public SRS3.2	Public TestConfigCreateUI	
Realization SRS3 to TestConfigCreateUI Source -> Destination	Public SRS3	Public TestConfigCreateUI	
Realization SRS1.2 to TestConfigCreateUI Source -> Destination	Public SRS1.2	Public TestConfigCreateUI	
Realization SRS11 to TestConfigCreateUI Source -> Destination	Public SRS11	Public TestConfigCreateUI	
Realization SRS62 to	Public	Public	

Connector	Source	Target	Notes
TestConfigCreateUI Source -> Destination	SRS62	TestConfigCreateUI	
Realization Source -> Destination	Public SRS10.7	Public TestConfigCreateUI	
Realization SRS63 to TestConfigCreateUI Source -> Destination	Public SRS63	Public TestConfigCreateUI	
Realization SRS10.5 to TestConfigCreateUI Source -> Destination	Public SRS10.5	Public TestConfigCreateUI	

Attributes

Attribute	Notes	Constraints and tags
appLayerFieldjLabel javax.swing.JLabel Private	The app layer fieldj label.	*Default:*
appLayerTestSuiteComboBox javax.swing.JComboBox Private	The app layer test suite combo box.	*Default:*
appLayerTestSuiteDescriptionTextArea javax.swing.JTextArea Private	The app layer test suite description text area.	*Default:*
appLayerTestSuiteSelected String Private	The app layer test suite selected.	*Default:*
appLayerTextField javax.swing.JTextField Private	The app layer text field.	*Default:*
appStandardAuthenticationLabel javax.swing.JLabel Private	The app standard authentication label.	*Default:*
browseButton	The browse button.	*Default:*

Attribute	Notes	Constraints and tags
javax.swing.JButton Private		
cancelButton javax.swing.JButton Private	The cancel button.	*Default:*
configCreator TestConfigCreateAction Private	The config creator.	*Default:*
infoLayerStandardFieldLabel javax.swing.JLabel Private	The info layer standard field label.	*Default:*
infoLayerTestSuiteComboBox javax.swing.JComboBox Private	The info layer test suite combo box.	*Default:*
infoLayerTestSuiteDescriptionTextArea javax.swing.JTextArea Private	The info layer test suite description text area.	*Default:*
infoLayerTestSuiteSelected String Private	The info layer test suite selected.	*Default:*
infoLayerTextField javax.swing.JTextField Private	The info layer text field.	*Default:*
infoStandardAuthenticationLabel javax.swing.JLabel Private	The info standard authentication label.	*Default:*
jPanel1 javax.swing.JPanel Private	The j panel1.	*Default:*

Attribute	Notes	Constraints and tags
jPanel2 javax.swing.JPanel Private	The j panel2.	*Default:*
listenerConfigFile ProgressListener Private	The listener config file.	*Default:*
okButton javax.swing.JButton Private	The ok button.	*Default:*
okPressed boolean Private	The ok pressed.	*Default:*
pathFieldLabel javax.swing.JLabel Private	The path field label.	*Default:*
pathSelectedTextField javax.swing.JTextField Private	The path selected text field.	*Default:*
reporterConfigFile ProgressReporter Private	The reporter config file.	*Default:* new ProgressUI(null, true)
testConfigDescriptionFieldjLabel javax.swing.JLabel Private	The test config description fieldj label.	*Default:*
testConfigDescriptionTextArea javax.swing.JTextArea Private	The test config description text area.	*Default:*
testConfigFieldLabel javax.swing.JLabel Private	The test config field label.	*Default:*
testConfigName String	The test config name.	*Default:*

Attribute	Notes	Constraints and tags
Private		
testConfigPath String Private	The test config path.	*Default:*
testConfigTextScrollPane javax.swing.JScrollPane Private	The test config text scroll pane.	*Default:*
testConfiguration TestConfiguration Private	The test configuration.	*Default:*
testConfigurationFilePanel javax.swing.JPanel Private	The test configuration file panel.	*Default:*
testConfigurationTextField javax.swing.JTextField Private	The test configuration text field.	*Default:*
testDescription String Private	The test description.	*Default:*
testSuiteDescriptionFieldLabel javax.swing.JLabel Private	The test suite description field label.	*Default:*
testSuiteDescriptionFieldLabel1 javax.swing.JLabel Private	The test suite description field label1.	*Default:*
testSuiteDescriptionScrollPane javax.swing.JScrollPane Private	The test suite description scroll pane.	*Default:*
testSuiteDescriptionScrollPane1 javax.swing.JScrollPane	The test suite description scroll pane1.	*Default:*

Attribute	Notes	Constraints and tags
Private		
testSuiteFieldjLabel javax.swing.JLabel Private	The test suite fieldj label.	*Default:*
testSuiteFieldjLabel1 javax.swing.JLabel Private	The test suite fieldj label1.	*Default:*
testSuiteSelectionPanel javax.swing.JPanel Private	The test suite selection panel.	*Default:*

Operations

Method	Notes	Parameters
appLayerTestSuiteComboBoxItemStateChanged() void Private	App layer test suite combo box item state changed. Pre-Conditions: N/A Post-Conditions: N/A	java.awt.event.ItemEvent [in] evt the evt
browseButtonActionPerformed() void Private	Browse button action performed. Pre-Conditions: N/A Post-Conditions: N/A	java.awt.event.ActionEvent [in] evt the evt
cancelButtonActionPerformed() void Private	Cancel button action performed. Pre-Conditions: N/A Post-Conditions: N/A	java.awt.event.ActionEvent [in] evt the evt
checkPathUpdate() void Private	Check path update. Pre-Conditions: N/A Post-Conditions: N/A	
getAppLayerTestSuiteSelected() String Public	Gets the app layer test suite selected. @return the Application Layer Test Suite Selected	
getInfoLayerTestSuiteSelected() String Public	Gets the info layer test suite selected. @return the Information Layer Test Suite Selected	

Method	Notes	Parameters
getTestConfigName() String Public	Gets the test config name. @return the testConfigName	
getTestConfigPath() String Public	Gets the test config path. @return the testConfigPath	
getTestConfiguration() TestConfiguration Public	Gets the test configuration. @return the Test Configuration Created	
getTestDescription() String Public	Gets the test description. @return the Test Description	
infoLayerTestSuiteComboBoxItemStateChanged() void Private	Info layer test suite combo box item state changed. Pre-Conditions: N/A Post-Conditions: N/A	java.awt.event.ItemEvent [in] evt the evt
initComponents() void Private	This method is called from within the constructor to initialize the form. WARNING: Do NOT modify this code. The content of this method is always regenerated by the Form Editor.	
Static **main()** void Public	The main method. Pre-Conditions: N/A Post-Conditions: N/A	String[] [in] args the command line arguments
okButtonActionPerformed() void Private	Ok button action performed. Pre-Conditions: N/A Post-Conditions: N/A	java.awt.event.ActionEvent [in] evt the evt
pathSelectedTextFieldActionPerformed() void Private	Path selected text field action performed. Pre-Conditions: N/A Post-Conditions: N/A	java.awt.event.ActionEvent [in] evt the evt
pathSelectedTextFieldFocusLost() void Private	Path selected text field focus lost. Pre-Conditions: N/A Post-Conditions: N/A	java.awt.event.FocusEvent [in] evt the evt
setAppAuthenticationLabel() void	Sets the app authentication label. Pre-Conditions: N/A Post-Conditions: N/A	

Method	Notes	Parameters
Private		
setInfoAuthenticationLabel() void	Sets the info authentication label. Pre-Conditions: N/A Post-Conditions: N/A	
Private		
showDialog() boolean	Show dialog. Pre-Conditions: N/A Post-Conditions: N/A	
Public	@return flag indicating whether the test configuration whether the user entered valid values and clicked ok (true) or canceled (false)	
TestConfigCreateUI()	Creates new form TestConfigCreateUI.	java.awt.Frame [in] parent
Public		the parent frame for this objectboolean [in] modal
		indicates whether the dialog should be modal

5.1.2.2.12 Test Case Editor

5.1.2.2.12.1 Primary Presentation

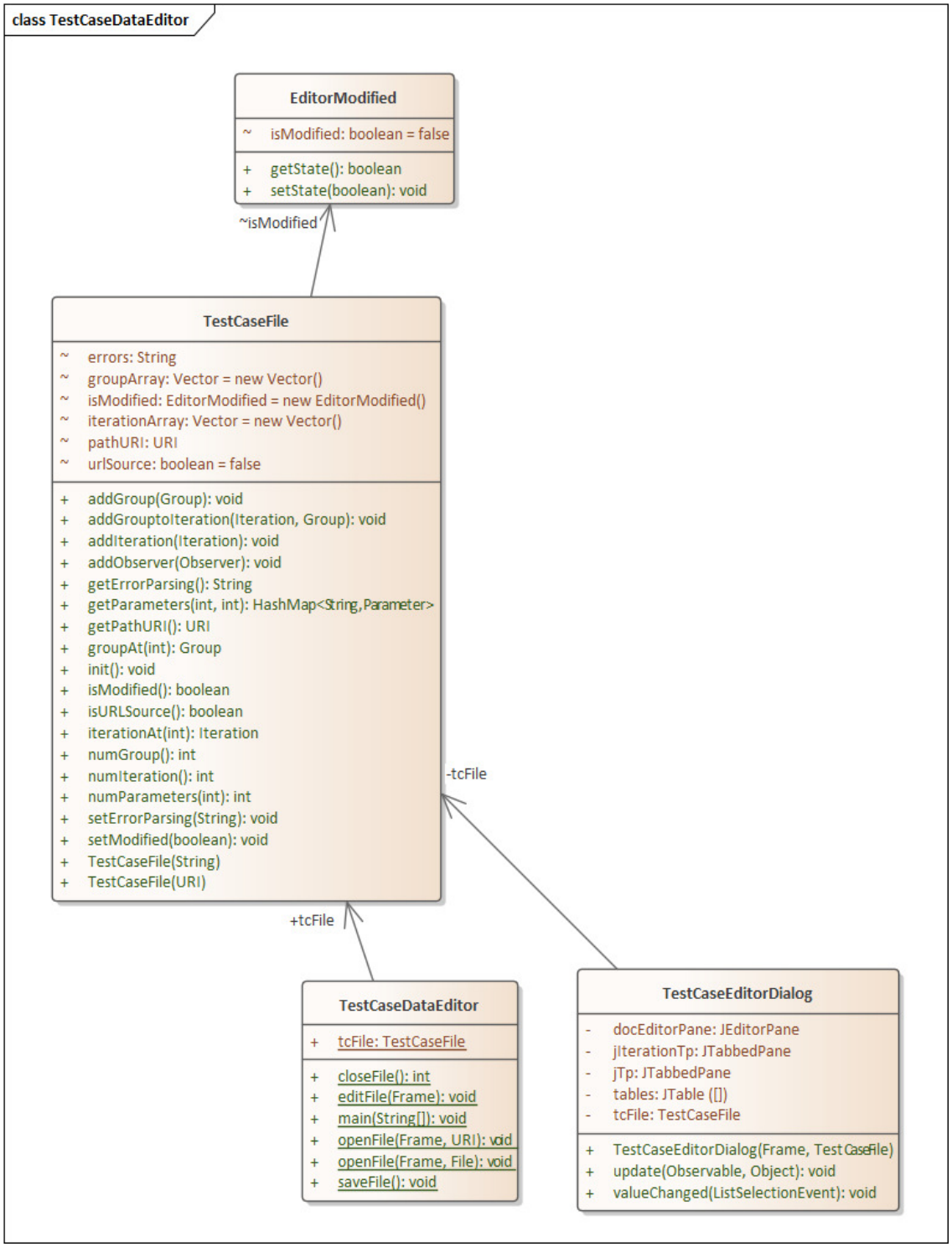

Figure 5-3 Test Case Data Editor Modules

The Test Data Editor feature allows users to edit the test configuration data from within the C2C RI and also allows users to see the impact of the changes to the configuration files.

5.1.2.2.12.2 Element Catalog

5.1.2.2.12.2.1 TestCaseEditorDialog

This module allows the user to select a Custom Test Case file to be loaded into the editor. If the user hasn't previously selected/defined a custom test case data file, then a file selection dialog will be presented to allow the user to select a data file that is currently on disk, or specficy the name for a new file.

5.1.2.2.12.2.2 TestCaseDataEditor

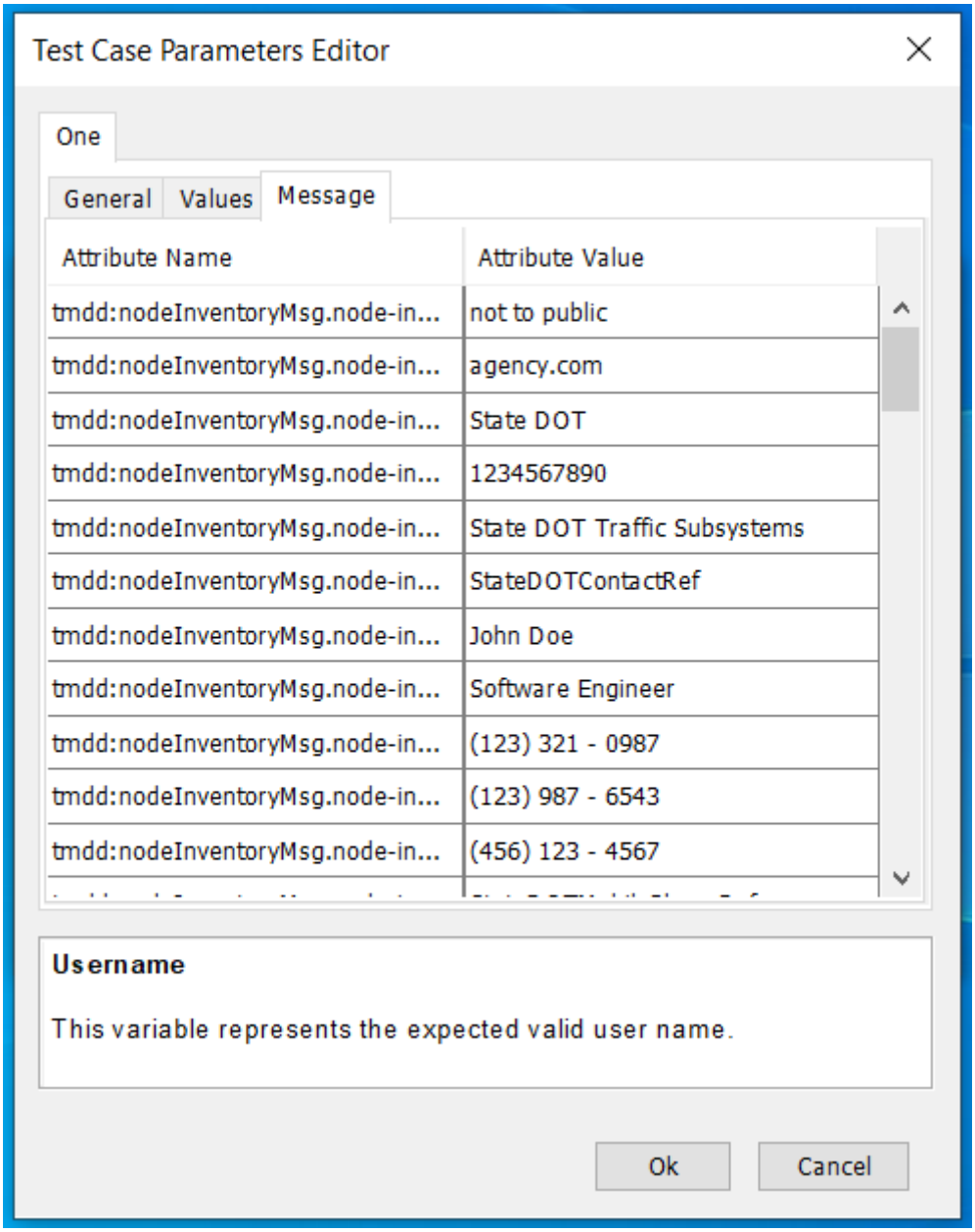

Figure 5-4 Test Case Data Editor

This module provides the user with the ability to create, edit and delete parameters from a custom test configuration file. The parameters will include iterations, groups and individual parameters. Each parameter includes a type. The editor will be sensitive to certain types and render the Attribute Value

column appropriately. For instance, if the parameter type is boolean, then user will be presented with a true/false pull down option.

When the editor is used to show a merged Test Configuration data file, the editor will not allow changes to fields marked as non-editable.

5.1.2.2.12.2.3 TestCaseFile

This module represents the Test Case Data File. When the user is editing a test case data file, then this object only contains the parameter in the user's test case data file. However, if the user is viewing the data file, then this object contains a merged copy of the user's test case data file (if defined) and the test case data file included in the Test Suite.

5.1.2.2.12.2.4 EditorModified

This module maintains state for the Test Case Data file. If the use has edited the file the state is set to modified. This state will be used to prompt the user before the file is saved.

5.1.2.2.12.3 Context Diagram

N/A

5.1.2.2.12.4 Variability Guide

N/A

5.1.2.2.12.5 Rationale

The design for this editor is very similar to the editor available through the currently available C2C RI Options window. This allows for consistency in editors in the program. Additionally, the format of the Test Case Data file and the c2cri.properties file are identical.

5.1.2.2.13 C2C RI Wizard

5.1.2.2.13.1 Primary Presentation

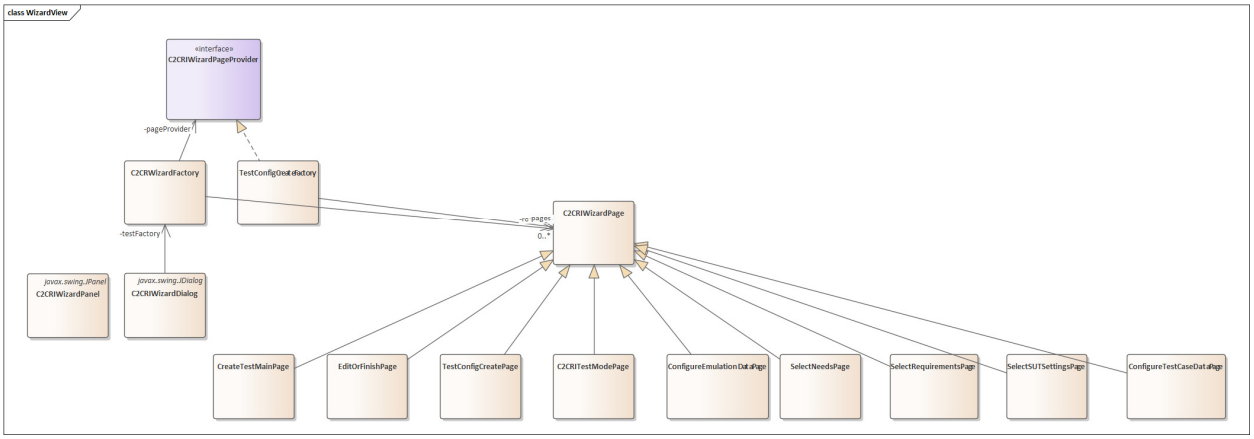

Figure 5-5: C2C RI Wizard Modules

This enhancement to the C2C RI guides users within the tool to correctly create and execute conformance tests and generate reports. This includes an interactive 'wizard' within the tool to guide users in the functionality of the software.

5.1.2.2.13.2 Element Catalog

5.1.2.2.13.2.1 C2CRIWizardPanel

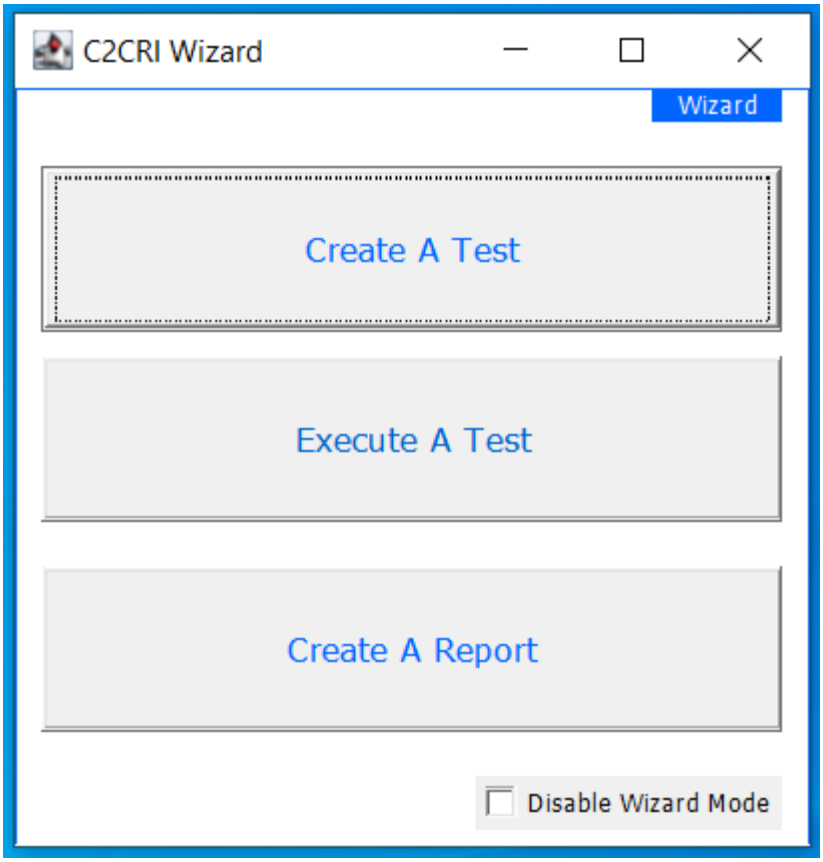

Figure 5-6 C2C RI Wizard Initial Selection Panel

The entry into the C2C RI Guided mode is provided through this module. The user interface lets the user select whether to create a new test configuration file, execute a test or create a report. Additionally the interface prsents the user the option to disable the Wizard mode, so that the user doesn't see the wizard panel the next time the C2C RI is started. The panel is implemented using standard java swing components. If a new test configuration is requested then control is passed over to the Test Configuration Wizard pages. Otherwise, control is passed back to the RIUICoordinator for further processing.

5.1.2.2.13.2.2 C2CRIWizardDialog

The main java swing JDialog that contains a JTree navigation panel and the currently presented C2CRI Wizard Page.

5.1.2.2.13.2.3 C2CRIWizardPageProvider

This interface defines the operations that must be provided by each implemented C2C RI Wizard Page Provider. The provide returns the next available page after a user selects the next button. Additionally, it provides a list of pages being managed. Currently, the design only includes a provider for Test Configuration definition, but in the future providers can be developed for other C2C RI work flows. The provider is contained in the C2CRIWizardFactory.

5.1.2.2.13.2.4 C2CRIWizardFactory

This module provides the C2CRIWizardDialog with the next page results and the current set of defined pages. This information allows the C2CRIWizardDialogs navigation panel to provide a status of the users progress in defining a test configuration.

5.1.2.2.13.2.5 TestConfigCreateFactory

This module is responsible for creating and managing the set of available Test Configuration pages presented to the user. The options available will vary depending on whether the user is creating a new test configuration, editing an existing configuration and the information and application layer needs selected.

5.1.2.2.13.2.6 C2CRIWizardPage

This module is an abstract class which provides the basic functionality that must exist in all pages used in the wizard.

5.1.2.2.13.2.7 CreateTestMainPage

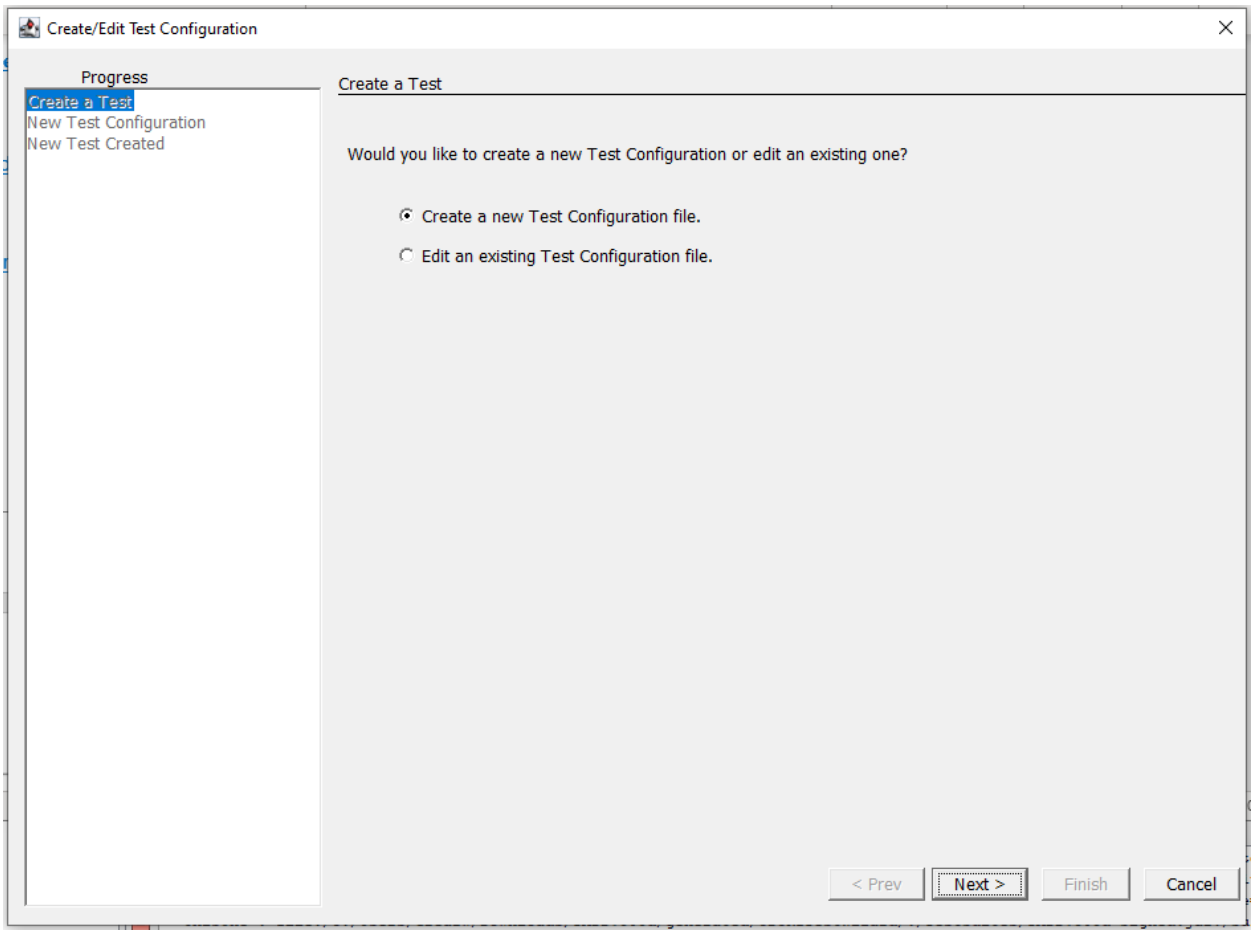

Figure 5-7 Create Test Configuration Main Page

This module provides the initial Test Configuration Wizard Page which allows the user to select whether a new test configuration is being developed, or an existing page will be edited.

5.1.2.2.13.2.8 EditOrFinishPage

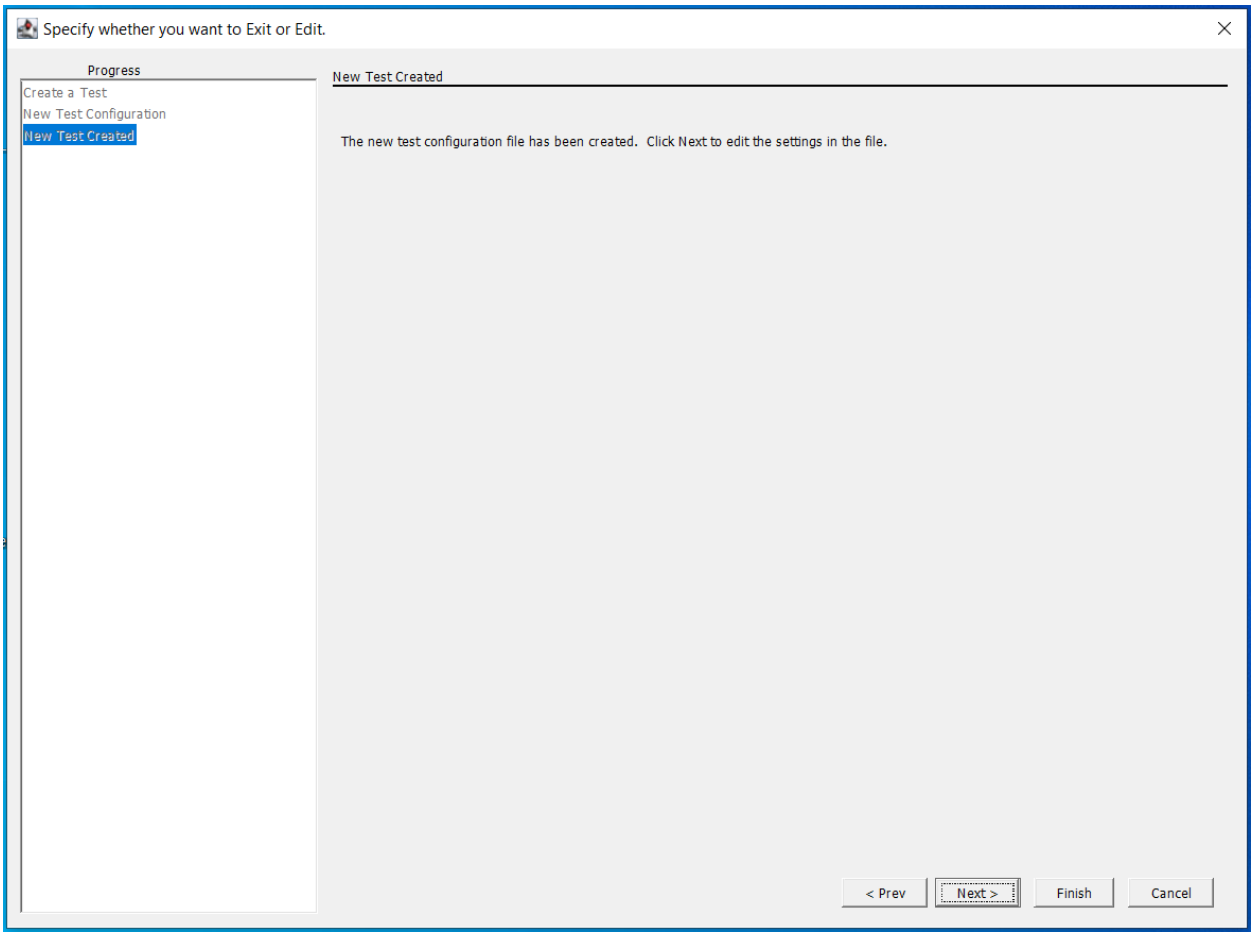

Figure 5-8 Exit or Edit Page

This module is presented after a user has created a new Test Configuration page. The user may continue on to edit the configuration file or finish.

5.1.2.2.13.2.9 TestConfigCreatePage

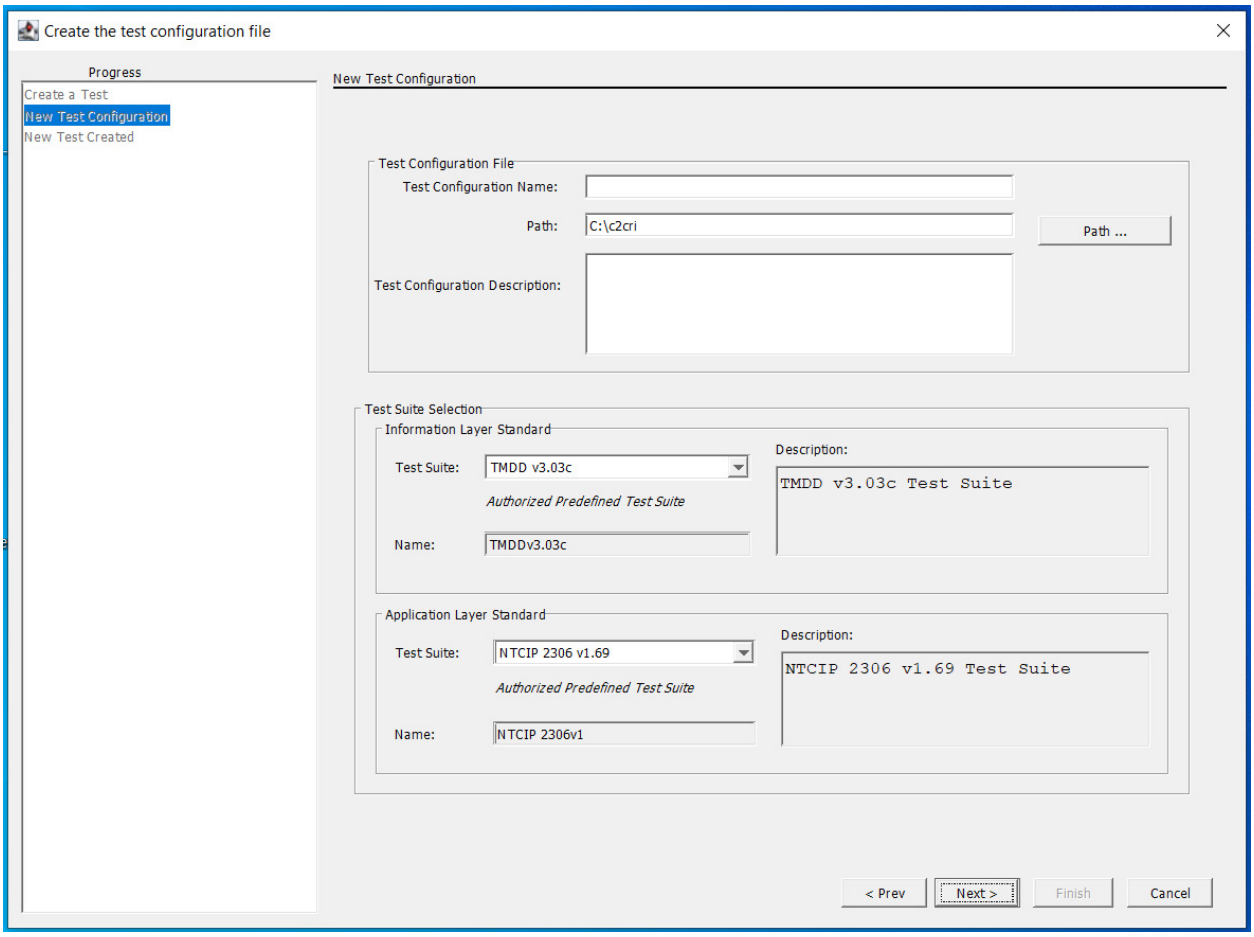

Figure 5-9 New Test Configuration Page

This module provides the initial Test Configuration Creation Page which prompts the user for the required test configuration name and applicable information and application layer test suites. Upon clicking the next button, the Test Configuration file will be created.

5.1.2.2.13.2.10 C2CRITestModePage

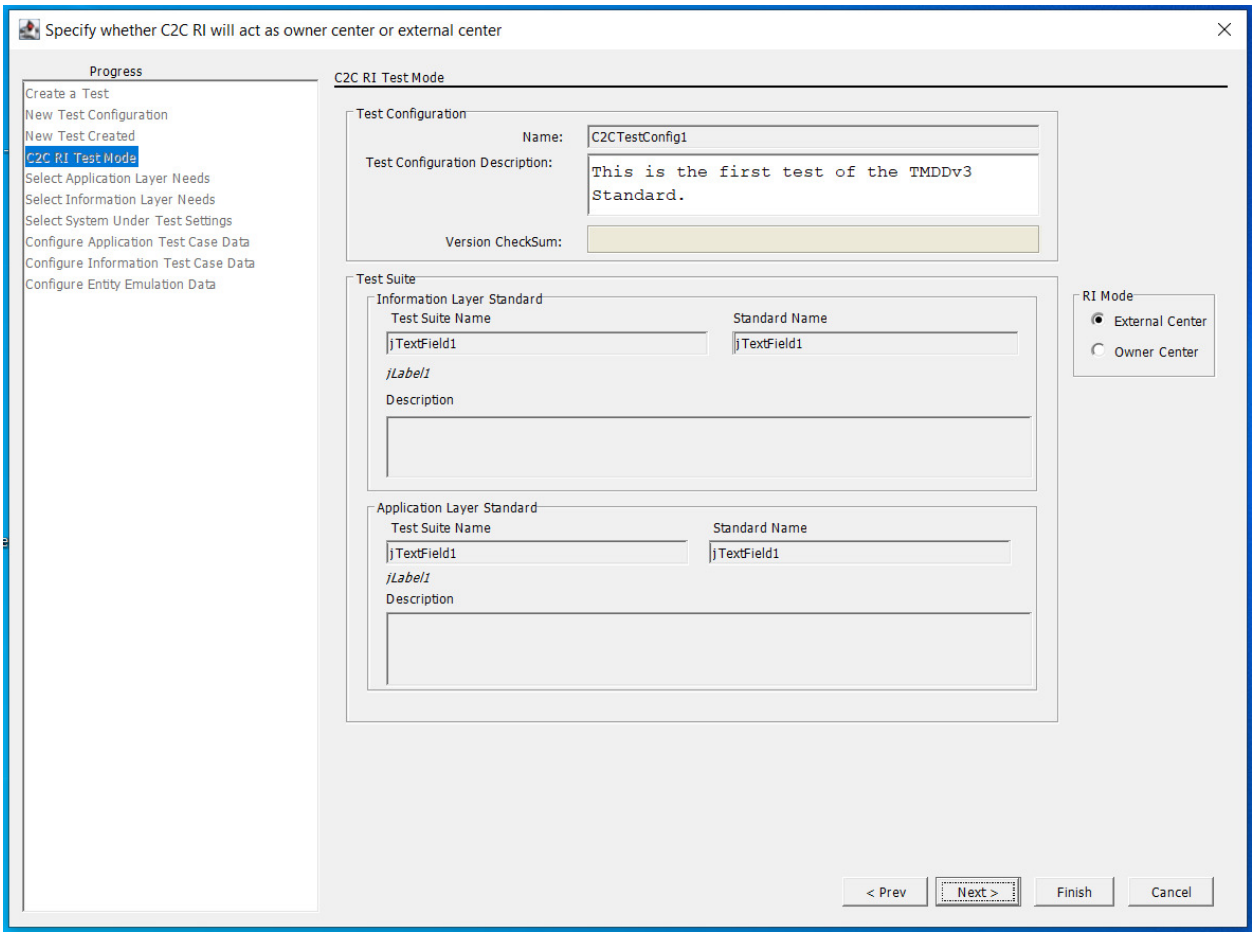

Figure 5-10 Test Mode Selection Page

This module provides the initial Test Configuration editing page which allows the user to select whether the RI will be acting as the owner center or external center in the test. The other fields on the screen will present the information defined in the Test Configuration File. The Finish button is enabled because the user should be able to save a partially edited configuration file.

5.1.2.2.13.2.11 ConfigureEmulationDataPage

This module provides the user with the ability to edit emulation data files and command queue lengths. The options available will be consistent with the existing Emulation Data File editing tag. The Finish button is enabled because the user should be able to save a partially edited (or completed) configuration file.

5.1.2.2.13.2.12 SelectNeedsPage

This module will be used for the selection of both Information and Application Layer Needs. At least one of each need must be selected. The page will provide an indication as to whether it is being used for the Information or Application Layers. The Finish button is enabled because the user should be able to save a partially edited configuration file.

Once the next button is selected the set of navigable pages will be changed to reflect the addition/removal of requirements selection pages as applicable.

5.1.2.2.13.2.13 SelectRequirementsPage

This module will be used for the selection of both Information and Application Layer Requirements. At least one of each requirement must be selected. The page will provide an indication as to whether it is being used for the Information or Application Layers as well as the selected need. The Finish button is enabled because the user should be able to save a partially edited configuration file.

5.1.2.2.13.2.14 SelectSUTSettingsPage

This module will be used for the selection of SUT settings to include the WSDL file name. The page will include a hint that specifies that the WSDL file name is needed for the NTCIP 2306 Test Suite to operate properly. The Finish button is enabled because the user should be able to save a partially edited configuration file.

5.1.2.2.13.2.15 ConfigureTestCaseDataPage

This module will be used for the selection and editing of both Information and Application Layer Test Cases. The set of test cases presented will be limited to the test cases applicable to the currently selected needs, requirements and C2C RI Mode. The test case editing feature will be available from this page as well. The Finish button is enabled because the user should be able to save a partially edited configuration file.

5.1.2.2.13.3 Context Diagram

NA

5.1.2.2.13.4 Variability Guide

The current design includes a provisions for a wizard to guide the creation and editing of Test Configuration Files. Additional wizards can be added in the future by creating wizards that implement the C2CRIWizardProvider interface.

5.1.2.2.13.5 Rationale

The modules within this section are intended to address the SRS1.4, SRS1.4.1, SRS1.4.2 and SRS1.4.3 requirements related to the new Wizard Mode. Additionally, the implementation needs to also satisfy the Section508 requirements already allocated to the user interface.

5.1.2.3 Context Diagram

N/A

5.1.2.4 Variability Guide

N/A

5.1.2.5 Rationale

N/A

5.1.3 Reports GUI

5.1.3.1 Primary Presentation

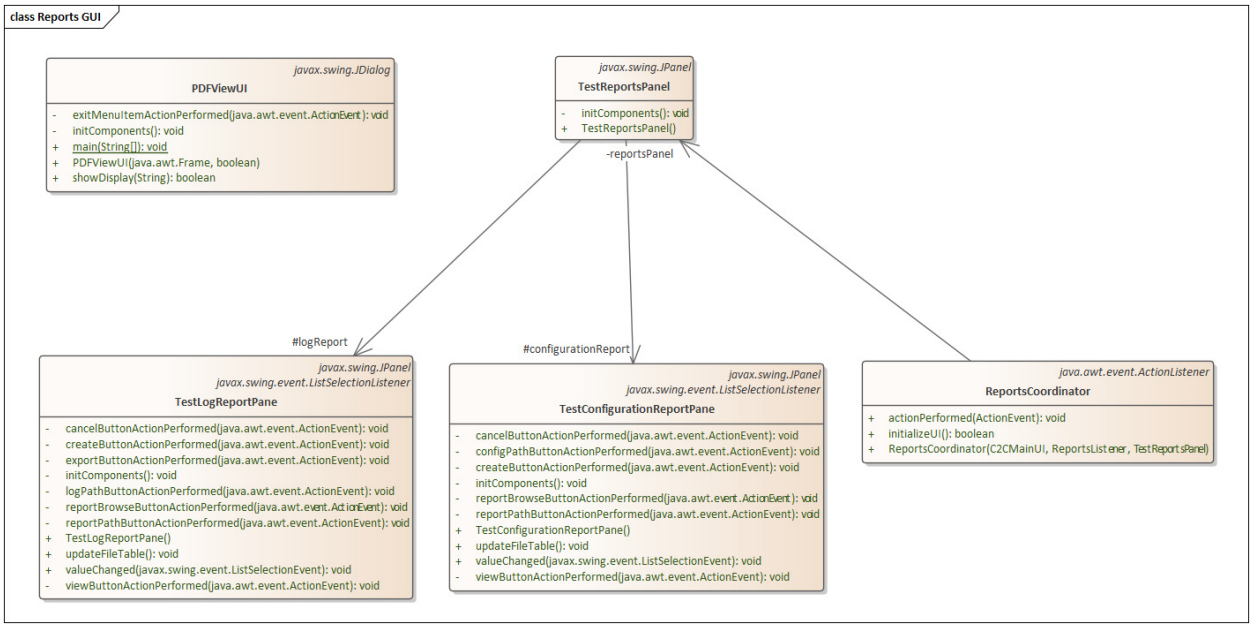

Figure 5-11 Reports GUI Modules

This module provides the user with access to create and view a variety of user generated reports. The reports are stored in PDF format and are displayed via a pdf viewer.

5.1.3.2 Element Catalog

5.1.3.2.1 ReportsCoordinator

Type: **Class** java.awt.event.ActionListener *Package:* gui

The Class ReportsCoordinator coordinates control of the Report GUIs.

Connections

Connector	Source	Target	Notes
Association	Public	Private mainUI	

Connector	Source	Target	Notes
Source -> Destination	ReportsCoordinator	C2CMainUI	
Association	Public	Private parentObject	
Source -> Destination	ReportsCoordinator	ReportsListener	
Association	Public	Private reportsPanel	
Source -> Destination	ReportsCoordinator	TestReportsPanel	
Association	Public	Private reporterReport	
Source -> Destination	ReportsCoordinator	ProgressReporter	
Association	Public	Private listenerReport	
Source -> Destination	ReportsCoordinator	ProgressListener	
Realization SRS49 to ReportsCoordinator	Public	Public	
Source -> Destination	SRS49	ReportsCoordinator	
Realization SRS48 to ReportsCoordinator	Public	Public	
Source -> Destination	SRS48	ReportsCoordinator	
Realization	Public	Public	
Source -> Destination	SRS55	ReportsCoordinator	
Realization SRS52 to ReportsCoordinator	Public	Public	
Source -> Destination	SRS52	ReportsCoordinator	
Realization SRS50 to ReportsCoordinator	Public	Public	
Source -> Destination	SRS50	ReportsCoordinator	
Association	Public	Private reportsUICoordinator	
Source -> Destination	RIUICoordinator	ReportsCoordinator	
Realization SRS63 to ReportsCoordinator	Public	Public	
Source -> Destination	SRS63	ReportsCoordinator	
Realization SRS62 to ReportsCoordinator	Public	Public	
Source -> Destination	SRS62	ReportsCoordinator	

Attributes

Attribute	Notes	Constraints and tags
listenerReport ProgressListener Private	The listener report.	*Default:*

Attribute	Notes	Constraints and tags
mainUI C2CMainUI Private	The main ui.	*Default:*
parentObject ReportsListener Private	The parent object.	*Default:*
reporterReport ProgressReporter Private	The reporter report.	*Default:*
reportsPanel TestReportsPanel Private Static	The reports panel.	*Default:*

Operations

Method	Notes	Parameters
actionPerformed() void Public	Handles actions from the ReportsPanel.	ActionEvent [in] e the action event received
initializeUI() boolean Public	Initialize the Reports UI. Set the default buttons and make this class the listener for the panel buttons. @return flag indicating whether the initialization was successful.	
ReportsCoordinator() Public	Constructor for the ReportsCoordinator Class.	C2CMainUI [in] parentFrame - the parent frame for the applicationReportsListener [in] parentObject - the calling object which implements ReportsListenerTestReportsPanel [in] reportsPanel - the reportsPanel that will be managed.

5.1.3.2.2 TestConfigurationReportPane

Type: **Class** javax.swing.JPanel, javax.swing.event.ListSelectionListener *Package:* gui

Panel for specifying and initiatiating a test configuration file report.

Connections

Connector	Source	Target	Notes
Realization SRS48.1 to TestConfigurationReportPane Source -> Destination	Public SRS48.1	Public TestConfigurationReportPane	
Realization SRS48.5 to TestConfigurationReportPane Source -> Destination	Public SRS48.5	Public TestConfigurationReportPane	
Realization SRS48.7 to TestConfigurationReportPane Source -> Destination	Public SRS48.7	Public TestConfigurationReportPane	
Association Source -> Destination	Public TestReportsPanel	Protected configurationReport TestConfigurationReportPane	

Attributes

Attribute	Notes	Constraints and tags
allRadioButton javax.swing.JRadioButton Protected	Variables declaration - do not modify//GEN-BEGIN:variables	*Default:*
cancelButton javax.swing.JButton Protected		*Default:*
configFilePathTextField javax.swing.JTextField Protected		*Default:*
configFileScrollPane javax.swing.JScrollPane Private		*Default:*
configFileSelectionTable		*Default:*

Attribute	Notes	Constraints and tags
javax.swing.JTable Private		
configName String Private	The config name.	*Default:*
configNameTextField javax.swing.JTextField Protected		*Default:*
configPath String Private	The config path.	*Default:*
configPathButton javax.swing.JButton Private		*Default:*
createButton javax.swing.JButton Protected		*Default:*
reportBrowseButton javax.swing.JButton Private		*Default:*
reportButtonGroup javax.swing.ButtonGroup Protected		*Default:*
reportDetailsPanel javax.swing.JPanel Private		*Default:*
reportName String Private	The report name.	*Default:*
reportNameLabel1 javax.swing.JLabel		*Default:*

Attribute	Notes	Constraints and tags
Private		
reportNameLabel2 javax.swing.JLabel Private		*Default:*
reportNameLabel3 javax.swing.JLabel Private		*Default:*
reportNameLabel4 javax.swing.JLabel Private		*Default:*
reportNameTextField javax.swing.JTextField Protected		*Default:*
reportPanel javax.swing.JPanel Private		*Default:*
reportPath String Private	The report path.	*Default:*
reportPathButton javax.swing.JButton Private		*Default:*
reportPathTextField javax.swing.JTextField Protected		*Default:*
scriptsRadioButton javax.swing.JRadioButton Protected		*Default:*
testConfigPanel javax.swing.JPanel		*Default:*

Attribute	Notes	Constraints and tags
Private		
viewButton javax.swing.JButton Protected		*Default:*

Operations

Method	Notes	Parameters
cancelButtonActionPerformed() void Private	Method called when the user presses the cancel button. The TestReportsPanel is notified so that this panel can be removed from the users view.	java.awt.event.ActionEvent [in] evt the evt
configPathButtonActionPerformed() void Private	Config path button action performed. Pre-Conditions: N/A Post-Conditions: N/A	java.awt.event.ActionEvent [in] evt the evt
createButtonActionPerformed() void Private	Method called when the user presses the create button. The RIReporter is notified of the user settings so that a report can be generated. At the conclusion of this method the View button is enabled.	java.awt.event.ActionEvent [in] evt the evt
initComponents() void Private	This method is called from within the constructor to initialize the form. WARNING: Do NOT modify this code. The content of this method is always regenerated by the Form Editor.	
reportBrowseButtonActionPerformed() void Private	Report browse button action performed. Pre-Conditions: N/A Post-Conditions: N/A	java.awt.event.ActionEvent [in] evt the evt
reportPathButtonActionPerformed() void Private	Report path button action performed. Pre-Conditions: N/A Post-Conditions: N/A	java.awt.event.ActionEvent [in] evt the evt
TestConfigurationReportPane() Public	Creates new form TestConfigurationReportPane.	
updateFileTable() void Public	Update file table. Pre-Conditions: N/A Post-Conditions: N/A	
valueChanged() void	(non-Javadoc) @see	javax.swing.event.ListSelectionEvent [in] e

Method	Notes	Parameters
Public	javax.swing.event.ListSelectionListener#valueChanged(javax.swing.event.ListSelectionEvent)	
viewButtonActionPerformed( ) void Private	Method called when the user presses the View button. The report that was created is displayed using the JasperReport Viewer.	java.awt.event.ActionEvent [in] evt the evt

5.1.3.2.3 TestReportsPanel

Type: **Class** javax.swing.JPanel *Package:* gui

The Class TestReportsPanel.

Connections

Connector	Source	Target	Notes
Association	Public	Protected logReport	
Source -> Destination	TestReportsPanel	TestLogReportPane	
Association	Public	Protected configurationReport	
Source -> Destination	TestReportsPanel	TestConfigurationReportPane	
Association	Public	Private reportsPanel	
Source -> Destination	ReportsCoordinator	TestReportsPanel	
Realization SRS48.11 to TestReportsPanel	Public	Public	
Source -> Destination	SRS48.11	TestReportsPanel	
Association	Public	Private reportsPanel	
Source -> Destination	RIUICoordinator	TestReportsPanel	
Realization SRS48.10 to TestReportsPanel	Public	Public	
Source -> Destination	SRS48.10	TestReportsPanel	
Realization SRS48.8 to TestReportsPanel	Public	Public	
Source -> Destination	SRS48.8	TestReportsPanel	
Realization	Public	Public	
Source -> Destination	SRS55	TestReportsPanel	
Realization SRS3 to	Public	Public	

Connector	Source	Target	Notes
TestReportsPanel Source -> Destination	SRS3	TestReportsPanel	
Realization	Public	Public	
Source -> Destination	SRS3.4	TestReportsPanel	
Realization	Public	Public	
Source -> Destination	SRS3.1	TestReportsPanel	
Realization SRS48 to TestReportsPanel Source -> Destination	Public SRS48	Public TestReportsPanel	
Realization	Public	Public	
Source -> Destination	SRS3.2	TestReportsPanel	
Realization SRS58.1 to TestReportsPanel Source -> Destination	Public SRS58.1	Public TestReportsPanel	
Realization	Public	Public	
Source -> Destination	SRS3.3	TestReportsPanel	
Realization SRS48.9 to TestReportsPanel Source -> Destination	Public SRS48.9	Public TestReportsPanel	

Attributes

Attribute	Notes	Constraints and tags
configurationReport TestConfigurationReportPane Protected	reference to the Configuration Report Pane.	*Default:*
logReport TestLogReportPane Protected	reference to the Test Log Report Pane.	*Default:*
reportsSubPanel javax.swing.JTabbedPane Private	Variables declaration - do not modify//GEN-BEGIN:variables	*Default:*

Operations

Method	Notes	Parameters
initComponents() void Private	This method is called from within the constructor to initialize the form. WARNING: Do NOT modify this code. The	

Method	Notes	Parameters
	content of this method is always regenerated by the Form Editor.	
TestReportsPanel() Public	Creates new form TestReportsPanel.	

5.1.3.2.4 TestLogReportPane

Type: **Class** javax.swing.JPanel, javax.swing.event.ListSelectionListener *Package:* gui

The Class TestLogReportPane.

Connections

Connector	Source	Target	Notes
Realization SRS58.3 to TestLogReportPane Source -> Destination	Public SRS58.3	Public TestLogReportPane	
Realization SRS48.2 to TestLogReportPane Source -> Destination	Public SRS48.2	Public TestLogReportPane	
Realization SRS58.2 to TestLogReportPane Source -> Destination	Public SRS58.2	Public TestLogReportPane	
Association Source -> Destination	Public TestReportsPanel	Protected logReport TestLogReportPane	
Realization SRS48.6 to TestLogReportPane Source -> Destination	Public SRS48.6	Public TestLogReportPane	
Realization SRS58.5 to TestLogReportPane Source -> Destination	Public SRS58.5	Public TestLogReportPane	
Realization SRS58 to TestLogReportPane Source -> Destination	Public SRS58	Public TestLogReportPane	
Realization SRS48.3 to TestLogReportPane Source -> Destination	Public SRS48.3	Public TestLogReportPane	
Realization SRS60 to TestLogReportPane Source -> Destination	Public SRS60	Public TestLogReportPane	

Connector	Source	Target	Notes
Realization SRS48.7 to TestLogReportPane Source -> Destination	Public SRS48.7	Public TestLogReportPane	

Attributes

Attribute	Notes	Constraints and tags
cancelButton javax.swing.JButton Protected	GEN-LAST:event_logPathButtonActionPerformed Variables declaration - do not modify//GEN-BEGIN:variables	*Default:*
conformanceRadioButton javax.swing.JRadioButton Protected		*Default:*
createButton javax.swing.JButton Protected		*Default:*
exportButton javax.swing.JButton Protected		*Default:*
logFilePathTextField javax.swing.JTextField Protected		*Default:*
logFileScrollPane javax.swing.JScrollPane Private		*Default:*
logFileSelectionTable javax.swing.JTable Private		*Default:*
logName String Private	The log name.	*Default:*
logNameTextField javax.swing.JTextField		*Default:*

Attribute	Notes	Constraints and tags
Protected		
logPath String Private	The log path.	*Default:*
logPathButton javax.swing.JButton Private		*Default:*
msgDetailRadioButton javax.swing.JRadioButton Protected		*Default:*
msgSummaryRadioButton javax.swing.JRadioButton Protected		*Default:*
reportBrowseButton javax.swing.JButton Private		*Default:*
reportButtonGroup javax.swing.ButtonGroup Protected		*Default:*
reportDetailsPanel javax.swing.JPanel Private		*Default:*
reportName String Private	The report name.	*Default:*
reportNameLabel javax.swing.JLabel Private		*Default:*
reportNameLabel1 javax.swing.JLabel		*Default:*

Attribute	Notes	Constraints and tags
Private		
reportNameLabel2 javax.swing.JLabel Private		*Default:*
reportNameLabel3 javax.swing.JLabel Private		*Default:*
reportNameTextField javax.swing.JTextField Protected		*Default:*
reportPanel javax.swing.JPanel Private		*Default:*
reportPath String Private	The report path.	*Default:*
reportPathButton javax.swing.JButton Private		*Default:*
reportPathTextField javax.swing.JTextField Protected		*Default:*
scriptLogRadioButton javax.swing.JRadioButton Protected		*Default:*
testCaseDetailRadioButton javax.swing.JRadioButton Protected		*Default:*
testCaseSummaryRadioButto n javax.swing.JRadioButton		*Default:*

Attribute	Notes	Constraints and tags
Protected		
testLogPanel javax.swing.JPanel Private		*Default:*
viewButton javax.swing.JButton Protected		*Default:*

Operations

Method	Notes	Parameters
cancelButtonActionPerformed() void Private	Method called when the user presses the cancel button. The TestReportsPanel is notified so that this panel can be removed from the users view.	java.awt.event.ActionEvent [in] evt the evt
createButtonActionPerformed() void Private	Method called when the user presses the create button. The RIReporter is notified of the user settings so that a report can be generated. At the conclusion of this method the View button is enabled.	java.awt.event.ActionEvent [in] evt the evt
exportButtonActionPerformed() void Private	Method called when the user presses the export button. The RIReporter is notified of the user settings so that a report can be exported.	java.awt.event.ActionEvent [in] evt the evt
initComponents() void Private	This method is called from within the constructor to initialize the form. WARNING: Do NOT modify this code. The content of this method is always regenerated by the Form Editor.	
logPathButtonActionPerformed() void Private	Log path button action performed. Pre-Conditions: N/A Post-Conditions: N/A	java.awt.event.ActionEvent [in] evt the evt
reportBrowseButtonActionPerformed() void Private	Report browse button action performed. Pre-Conditions: N/A Post-Conditions: N/A	java.awt.event.ActionEvent [in] evt the evt
reportPathButtonActionPerformed() void	Report path button action performed. Pre-Conditions: N/A Post-Conditions: N/A	java.awt.event.ActionEvent [in] evt the evt

Method	Notes	Parameters
Private **TestLogReportPane()** Public	Creates new form TestLogReportPane.	
updateFileTable() void Public	Update file table. Pre-Conditions: N/A Post-Conditions: N/A	
valueChanged() void Public	(non-Javadoc) @see javax.swing.event.ListSelectionListener#valueChanged(javax.swing.event.ListSelectionEvent)	javax.swing.event.ListSelectionEvent [in] e
viewButtonActionPerformed() void Private	Method called when the user presses the View button. The report that was created is displayed using the JasperReport Viewer.	java.awt.event.ActionEvent [in] evt the evt

5.1.3.2.5 PDFViewUI

Type: **Class** javax.swing.JDialog *Package:* gui

The Class PDFViewUI provides the PDV View capability.

Connections

Connector	Source	Target	Notes
Realization SRS1.3 to PDFViewUI Source -> Destination	Public SRS1.3	Public PDFViewUI	
Realization SRS56 to PDFViewUI Source -> Destination	Public SRS56	Public PDFViewUI	
Realization SRS1.2 to PDFViewUI Source -> Destination	Public SRS1.2	Public PDFViewUI	
Realization SRS56.1 to PDFViewUI Source -> Destination	Public SRS56.1	Public PDFViewUI	

Attributes

Attribute	Notes	Constraints and tags
controller SwingController Private	The controller.	*Default:*
exitMenuItem javax.swing.JMenuItem Private	The exit menu item.	*Default:*
jMenu1 javax.swing.JMenu Private	The j menu1.	*Default:*
jMenuBar1 javax.swing.JMenuBar Private	The j menu bar1.	*Default:*

Operations

Method	Notes	Parameters
exitMenuItemActionPerformed() void Private	Exit menu item action performed. Pre-Conditions: N/A Post-Conditions: N/A	java.awt.event.ActionEvent [in] evt the evt
initComponents() void Private	This method is called from within the constructor to initialize the form. WARNING: Do NOT modify this code. The content of this method is always regenerated by the Form Editor.	
Static **main()** void Public	The main method. Pre-Conditions: N/A Post-Conditions: N/A	String[] [in] args the command line arguments
PDFViewUI() Public	Creates new form PDFViewUI.	java.awt.Frame [in] parent the parentboolean [in] modal the modal
showDisplay() boolean Public	Show display. Pre-Conditions: N/A Post-Conditions: N/A @return true, if successful	String [in] report the report

5.1.3.3 Context Diagram

N/A

5.1.3.4 Variability Guide

The system can be updated to include additional reports in the GUI. All reports are visible through the same PDF viewer interface.

5.1.3.5 Rationale

PDF file formats are very common and accessible by a number of viewing applications.

5.1.4 Test GUI

5.1.4.1 Primary Presentation

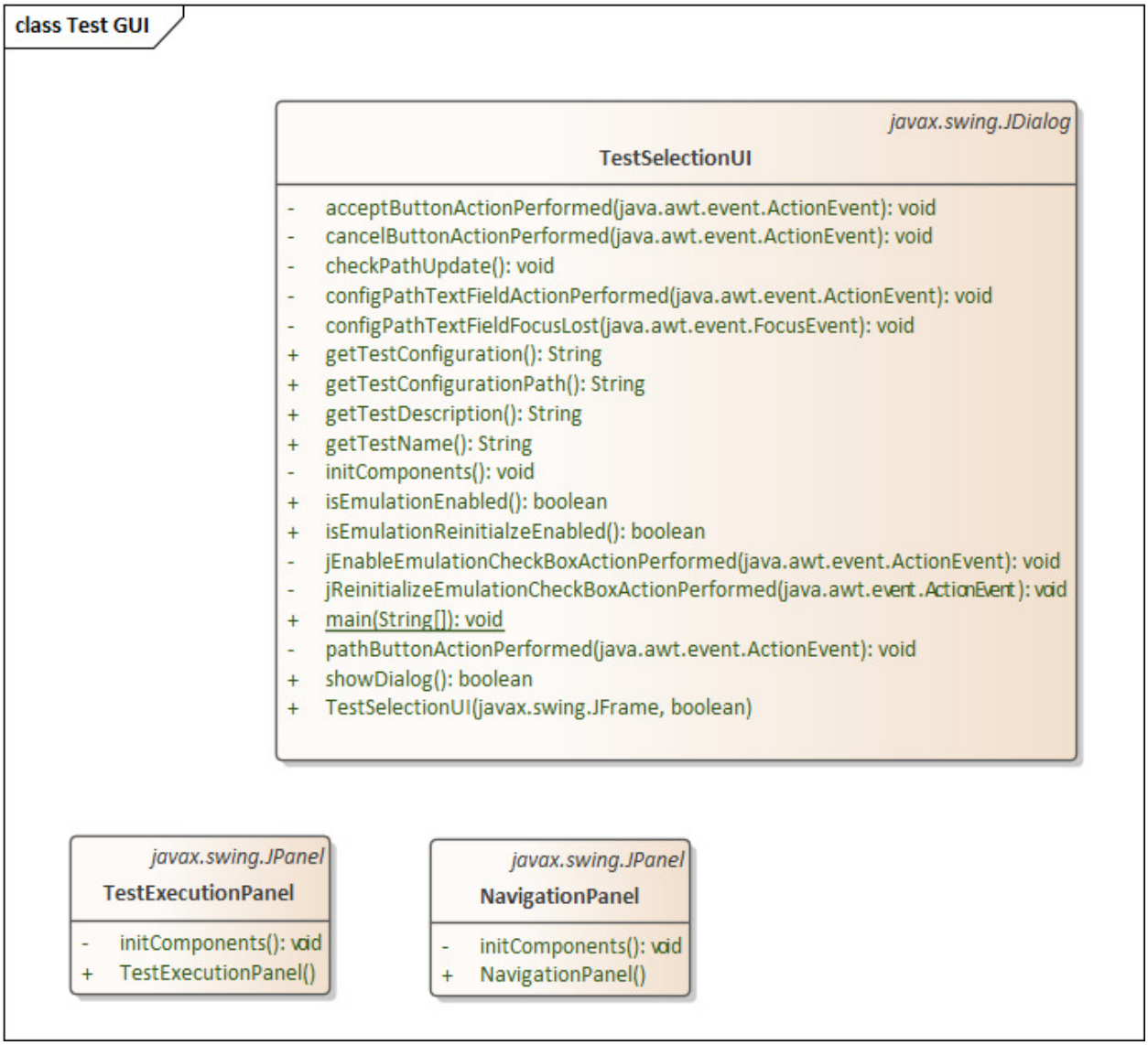

Figure 5-12 Test GUI Modules

The Test GUI provides the user with the ability to select a test configuration file for testing. Once initialized the user may select and execute test cases.

5.1.4.2 Element Catalog

5.1.4.2.1 TestSelectionUI

Type: **Class** javax.swing.JDialog *Package:* gui

The TestSelectionUI Dialog is used to specify a new test. It provides the user the ability to specify the name for the test, a description and the test configuration file to be used for the test. It extends the javax.swing.JDialog Class.

Connections

Connector	Source	Target	Notes
Realization SRS31.3 to TestSelectionUI Source -> Destination	Public SRS31.3	Public TestSelectionUI	
Realization SRS31.6 to TestSelectionUI Source -> Destination	Public SRS31.6	Public TestSelectionUI	
Realization SRS31 to TestSelectionUI Source -> Destination	Public SRS31	Public TestSelectionUI	
Realization SRS31.5 to TestSelectionUI Source -> Destination	Public SRS31.5	Public TestSelectionUI	
Realization SRS3 to TestSelectionUI Source -> Destination	Public SRS3	Public TestSelectionUI	
Realization Source -> Destination	Public SRS3.4	Public TestSelectionUI	
Realization Source -> Destination	Public SRS3.3	Public TestSelectionUI	
Realization Source -> Destination	Public SRS3.1	Public TestSelectionUI	
Realization Source -> Destination	Public SRS31.7	Public TestSelectionUI	
Realization Source -> Destination	Public SRS93	Public TestSelectionUI	
Realization Source -> Destination	Public SRS3.2	Public TestSelectionUI	
Realization Source -> Destination	Public SRS94	Public TestSelectionUI	
Realization SRS31.4 to TestSelectionUI Source -> Destination	Public SRS31.4	Public TestSelectionUI	
Realization SRS31.1 to TestSelectionUI	Public SRS31.1	Public TestSelectionUI	

Connector	Source	Target	Notes
Source -> Destination			

Attributes

Attribute	Notes	Constraints and tags
acceptButton javax.swing.JButton Private	Variables declaration - do not modify//GEN-BEGIN:variables	*Default:*
cancelButton javax.swing.JButton Private		*Default:*
configPathTextField javax.swing.JTextField Private		*Default:*
enableEntityEmulationCheckBox javax.swing.JCheckBox Private	The check box that allows the user to specify whether entity emulation will/will not be used for the test.	*Default:*
initializeEmulationDataCheckBox javax.swing.JCheckBox Private	Checkbox that allows the user to specify whether entity emulation will be initialized after each test case.	*Default:*
okPressed boolean Private	Flag used to indicate that the dialog was closed due to the user pressing the ok button.	*Default:*
parentFrame javax.swing.JFrame Private	The parent frame.	*Default:*
pathButton javax.swing.JButton Private		*Default:*
selectedFile File Private	The selected file.	*Default:*

Attribute	Notes	Constraints and tags
testConfiguration String Private	The name of the configuration file that the user specifies for a test.	*Default:*
testConfigurationLabel javax.swing.JLabel Private		*Default:*
testConfigurationPath String Private	The path to the configuration file that the user specifies for a test.	*Default:*
testDescription String Private	The description that the user specifies for a test.	*Default:*
testName String Private	The name that the user specifies for a test.	*Default:*
testNameLabel javax.swing.JLabel Private		*Default:*
testNameTextField javax.swing.JTextField Private		*Default:*
testNotesLabel javax.swing.JLabel Private		*Default:*
testNotesLabel1 javax.swing.JLabel Private		*Default:*
testNotesScrollPane javax.swing.JScrollPane Private		*Default:*
testNotesTextArea javax.swing.JTextArea		*Default:*

Attribute	Notes	Constraints and tags
Private		
testSelectionConfigFileTable javax.swing.JTable Private		*Default:*
testSelectionScrollPane javax.swing.JScrollPane Private		*Default:*

Operations

Method	Notes	Parameters
acceptButtonActionPerformed() void Private	The user has requested that the test be run with the specified parameters. Then notify the C2CMainUI that this dialog is complete. Note: The selected Configuration file name is presumed valid since it was selected from a list of available file names. In the future perform a check to ensure that associated test suite exists.	java.awt.event.ActionEvent [in] evt the evt
cancelButtonActionPerformed() void Private	The user has requested to cancel the operation. Set the okPressed flag to false, and make the dialog invisible.	java.awt.event.ActionEvent [in] evt the evt
checkPathUpdate() void Private	Check path update. Pre-Conditions: N/A Post-Conditions: N/A	
configPathTextFieldActionPerformed() void Private	Config path text field action performed. Pre-Conditions: N/A Post-Conditions: N/A	java.awt.event.ActionEvent [in] evt the evt
configPathTextFieldFocusLost() void Private	Config path text field focus lost. Pre-Conditions: N/A Post-Conditions: N/A	java.awt.event.FocusEvent [in] evt the evt
getTestConfiguration() String Public	Returns the user provided Test Configuration. @return - the selected Test Configuration file name	
getTestConfigurationPath()	Returns the user provided Test	

Method	Notes	Parameters
String Public	Configuration Path. @return - the path for the configuration file	
getTestDescription() String Public	Returns the user provided test description. @return - the test description	
getTestName() String Public	Returns the user provided test name. @return - the test name	
initComponents() void Private	This method is called from within the constructor to initialize the dialog window. WARNING: Do NOT modify this code. The content of this method is always regenerated by the Form Editor.	
Static **main()** void Public	This method is only used for testing purposes and is not used as part of the C2CRI application.	String[] [in] args the command line arguments
pathButtonActionPerformed() void Private	The user has clicked on the path button. Present the user with a JFileChooser window which was set to only show directories. If the user selected a path then set the configPathTextField to the specified path. Create a ConfigFileTableModel object using the file returned from JFileChooser. Finally, set the testSelectionConfigTable model to the new ConfigFileTableModel object.	java.awt.event.ActionEvent [in] evt - the pathButton Action Event
showDialog() boolean Public	This method presents the TestSelectionUI dialog to the user, and processes the input after the user has either selected OK or Cancel options. Create a ConfigFileTableModel object set to the default path for Config Files. Finally, set the testSelectionConfigTable model to the new ConfigFileTableModel object. Then make the dialog visible to the user. When the user closes out the dialog, store the user entered information and return true if the user pressed OK. If the user pressed cancel, just return false.	

Method	Notes	Parameters
	@return - flag indicating true if the user provided information, and false if the user canceled.	
TestSelectionUI() Public	Creates new window TestSelectionUI. It first calls the constructor for the java.awt.Frame class setting its modal parameter. Then it calls initComponents to place the screen components on the display and to establish the listeners. Events are handled local to this class.	javax.swing.JFrame [in] parent - the parent frame that this dialog is associated withboolean [in] modal - flag indicating whether the dialog will be modal or not.

TestSelectionUI - *(User Interface diagram)*

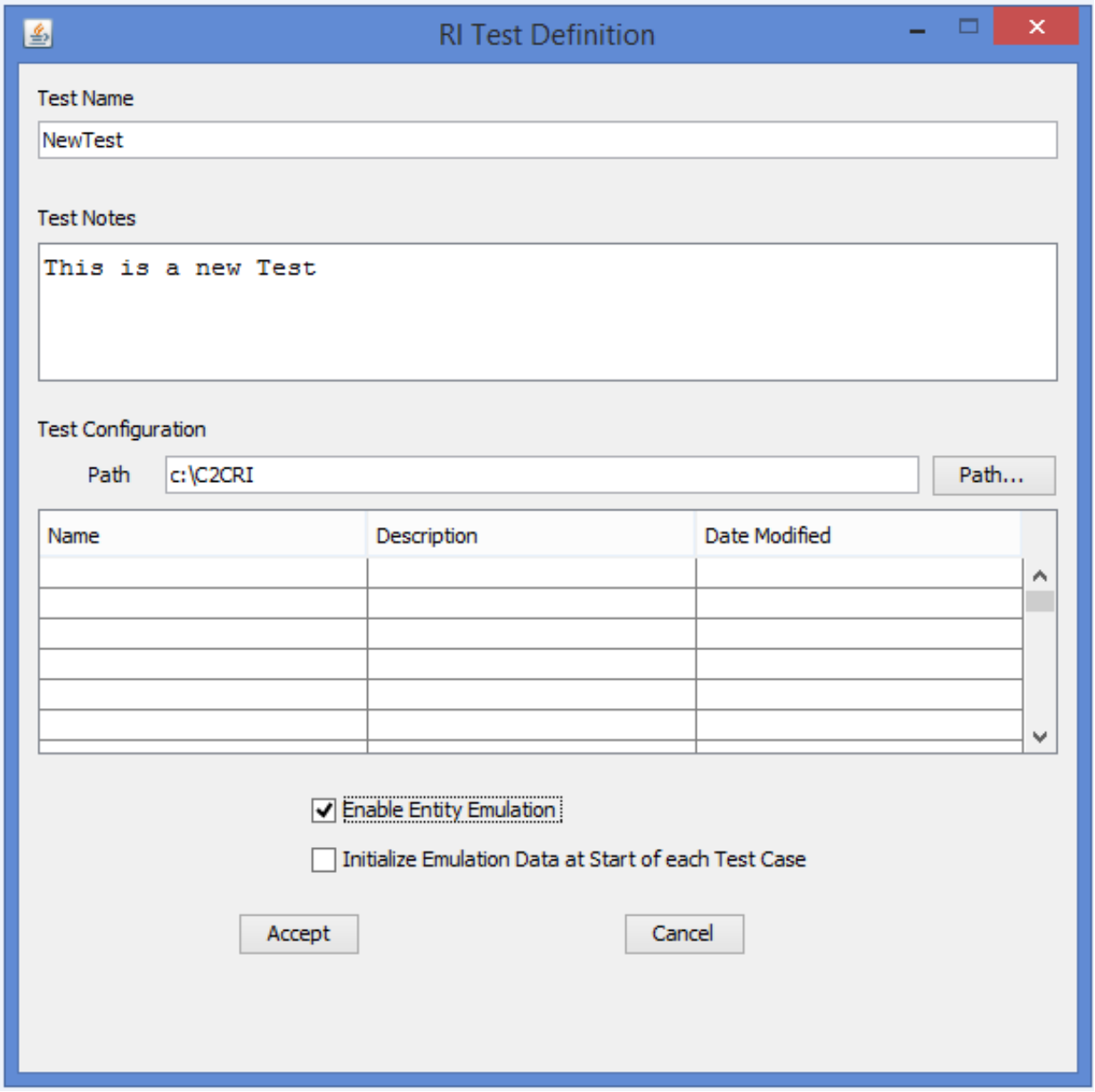

Figure 5-13: Test Definition Screen

5.1.4.2.2 TestExecutionPanel

Type: **Class** javax.swing.JPanel *Package:* gui

This class represents the TestExecutionPanel which the user controls the test with.

Connections

Connector	Source	Target	Notes
Sequence actionPerformed(e) Source -> Destination	Public TestExecutionPanel	Public TestExecutionCoordinator	
Sequence TestExecutionCoordinator (parentFrame, parentObject, executionPanel) Source -> Destination	Public TestExecutionPanel	Public TestExecutionCoordinator	
Realization SRS34.2 to TestExecutionPanel Source -> Destination	Public SRS34.2	Public TestExecutionPanel	
Realization SRS1.2 to TestExecutionPanel Source -> Destination	Public SRS1.2	Public TestExecutionPanel	
Realization Source -> Destination	Public SRS3.4	Public TestExecutionPanel	
Realization Source -> Destination	Public SRS3.2	Public TestExecutionPanel	
Realization SRS36.1 to TestExecutionPanel Source -> Destination	Public SRS36.1	Public TestExecutionPanel	
Realization SRS1.3 to TestExecutionPanel Source -> Destination	Public SRS1.3	Public TestExecutionPanel	
Association Source -> Destination	Public RIUICoordinator	Private executionPanel TestExecutionPanel	
Realization SRS34 to TestExecutionPanel Source -> Destination	Public SRS34	Public TestExecutionPanel	
Realization SRS33.1 to TestExecutionPanel Source -> Destination	Public SRS33.1	Public TestExecutionPanel	
Realization SRS33.2 to TestExecutionPanel Source -> Destination	Public SRS33.2	Public TestExecutionPanel	
Realization SRS35.3 to TestExecutionPanel Source -> Destination	Public SRS35.3	Public TestExecutionPanel	

Connector	Source	Target	Notes
Sequence Source -> Destination	Public TestExecutionCoordinator	Public TestExecutionPanel	
Realization SRS34.3 to TestExecutionPanel Source -> Destination	Public SRS34.3	Public TestExecutionPanel	
Realization Source -> Destination	Public SRS3.1	Public TestExecutionPanel	
Sequence Source -> Destination	Public TestExecutionCoordinator	Public TestExecutionPanel	
Realization SRS36.2 to TestExecutionPanel Source -> Destination	Public SRS36.2	Public TestExecutionPanel	
Association Source -> Destination	Public TestExecutionCoordinator	Private executionPanel TestExecutionPanel	
Realization SRS33.3 to TestExecutionPanel Source -> Destination	Public SRS33.3	Public TestExecutionPanel	
Realization SRS35 to TestExecutionPanel Source -> Destination	Public SRS35	Public TestExecutionPanel	
Realization SRS35.2 to TestExecutionPanel Source -> Destination	Public SRS35.2	Public TestExecutionPanel	
Realization Source -> Destination	Public SRS3.3	Public TestExecutionPanel	
Realization SRS3 to TestExecutionPanel Source -> Destination	Public SRS3	Public TestExecutionPanel	
Realization SRS36.3 to TestExecutionPanel Source -> Destination	Public SRS36.3	Public TestExecutionPanel	
Realization SRS32 to TestExecutionPanel Source -> Destination	Public SRS32	Public TestExecutionPanel	

Attributes

Attribute	Notes	Constraints and tags
actionTextArea javax.swing.JTextArea Protected	</editor-fold>//GEN-END:initComponents Variables declaration - do not modify//GEN-BEGIN:variables	*Default:*
actionTextScrollPane javax.swing.JScrollPane Private		*Default:*
appStandardTree javax.swing.JTree Protected		*Default:*
infoStandardTree javax.swing.JTree Protected		*Default:*
jLabel1 javax.swing.JLabel Private		*Default:*
jLabel2 javax.swing.JLabel Private		*Default:*
jPanel1 javax.swing.JPanel Private		*Default:*
jRunningLabel javax.swing.JLabel Protected		*Default:*
jScrollPane1 javax.swing.JScrollPane Private		*Default:*
jScrollPane2 javax.swing.JScrollPane Private		*Default:*

Attribute	Notes	Constraints and tags
jScrollPane3 javax.swing.JScrollPane Private		*Default:*
jScrollPane4 javax.swing.JScrollPane Private		*Default:*
jSplitPane1 javax.swing.JSplitPane Private		*Default:*
jTabbedPane1 javax.swing.JTabbedPane Private		*Default:*
jTestExecutionProgressBar javax.swing.JProgressBar Protected		*Default:*
pauseButton javax.swing.JButton Protected		*Default:*
reportPanel javax.swing.JPanel Protected		*Default:*
resumeButton javax.swing.JButton Protected		*Default:*
runButton javax.swing.JButton Protected		*Default:*
statusTabbedPane javax.swing.JTabbedPane Protected		*Default:*

Attribute	Notes	Constraints and tags
statusTextArea javax.swing.JTextArea Protected		*Default:*
statusTextScrollPane javax.swing.JScrollPane Private		*Default:*
terminateButton javax.swing.JButton Protected		*Default:*
testCaseTable javax.swing.JTable Protected		*Default:*
testNameLabel javax.swing.JLabel Private		*Default:*
testNameTextField javax.swing.JTextField Protected		*Default:*
testResultsPanel javax.swing.JPanel Protected		*Default:*
testResultsTable javax.swing.JTable Protected		*Default:*
testScriptLogLabel javax.swing.JLabel Private		*Default:*
testStatusLabel javax.swing.JLabel		*Default:*

Attribute	Notes	Constraints and tags
Private		
testStatusPanel javax.swing.JPanel Protected		*Default:*

Operations

Method	Notes	Parameters
initComponents() void Private	This method is called from within the constructor to initialize the panel. WARNING: Do NOT modify this code. The content of this method is always regenerated by the Form Editor.	
TestExecutionPanel() Public	Creates new form TestExecutionPanel. All panel components are initialized by calling initComponents.	

5.1.4.2.3 NavigationPanel

Type: **Class** javax.swing.JPanel *Package:* gui

The Class NavigationPanel.

Connections

Connector	Source	Target	Notes
Realization SRS1.2 to NavigationPanel Source -> Destination	Public SRS1.2	Public NavigationPanel	
Realization SRS1.3 to NavigationPanel Source -> Destination	Public SRS1.3	Public NavigationPanel	

Attributes

Attribute	Notes	Constraints and tags
jScrollPane1 javax.swing.JScrollPane Private	Variables declaration - do not modify//GEN-BEGIN:variables	*Default:*

Attribute	Notes	Constraints and tags
jTree1 javax.swing.JTree Private		*Default:*

Operations

Method	Notes	Parameters
initComponents() void Private	This method is called from within the constructor to initialize the form. WARNING: Do NOT modify this code. The content of this method is always regenerated by the Form Editor.	
NavigationPanel() Public	Creates new form NavigationPanel.	

5.1.4.3 Context Diagram

N/A

5.1.4.4 Variability Guide

N/A

5.1.4.5 Rationale

N/A

5.1.5 Test File Management GUI

5.1.5.1 Primary Presentation

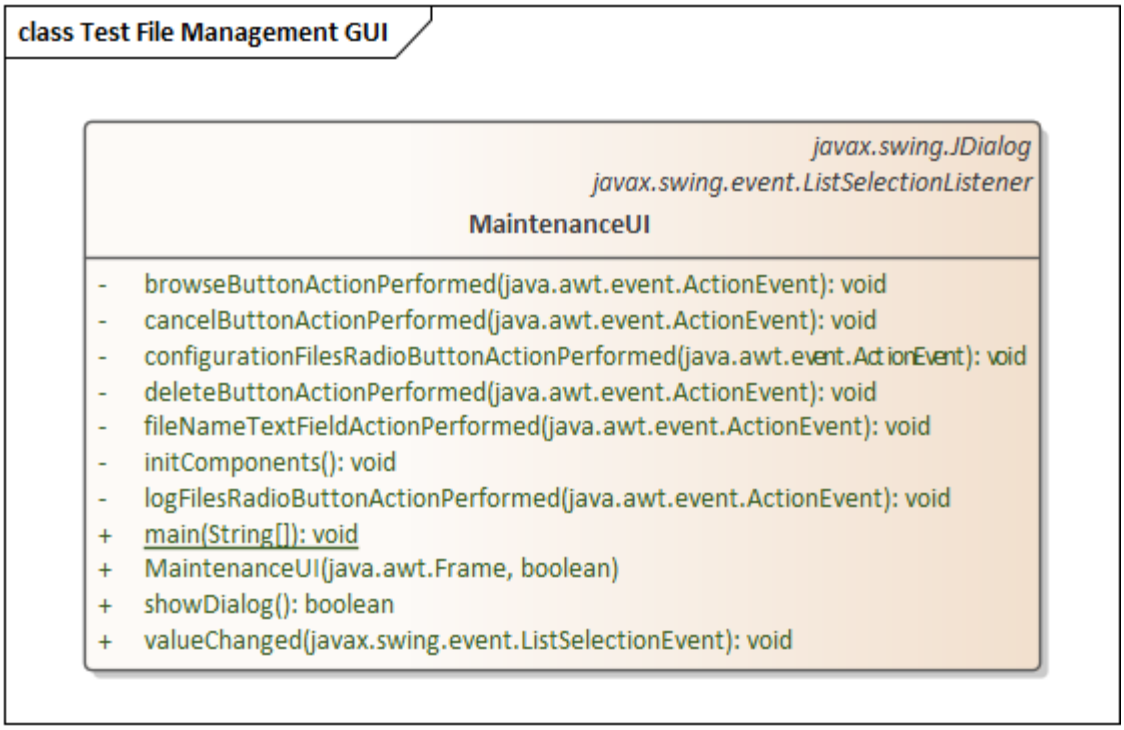

Figure 5-14 Test File Management GUI Modules

The Maintenance UI modulce provides the user with access to tools to remove Test Configuration and Test Log files.

5.1.5.2 Element Catalog

5.1.5.2.1 MaintenanceUI

Type: **Class** javax.swing.JDialog, javax.swing.event.ListSelectionListener *Package:* gui

The Class MaintenanceUI provides the file maintenace function to the user.

Connections

Connector	Source	Target	Notes
Realization SRS1.2 to MaintenanceUI Source -> Destination	Public SRS1.2	Public MaintenanceUI	
Realization SRS29.1 to MaintenanceUI Source -> Destination	Public SRS29.1	Public MaintenanceUI	

Connector	Source	Target	Notes
Realization SRS62 to MaintenanceUI Source -> Destination	Public SRS62	Public MaintenanceUI	
Realization SRS15 to MaintenanceUI Source -> Destination	Public SRS15	Public MaintenanceUI	
Realization SRS15.1 to MaintenanceUI Source -> Destination	Public SRS15.1	Public MaintenanceUI	
Realization SRS29 to MaintenanceUI Source -> Destination	Public SRS29	Public MaintenanceUI	
Realization SRS28.1 to MaintenanceUI Source -> Destination	Public SRS28.1	Public MaintenanceUI	
Realization SRS29.2 to MaintenanceUI Source -> Destination	Public SRS29.2	Public MaintenanceUI	
Realization SRS29.3 to MaintenanceUI Source -> Destination	Public SRS29.3	Public MaintenanceUI	
Realization SRS1.3 to MaintenanceUI Source -> Destination	Public SRS1.3	Public MaintenanceUI	
Realization SRS28.3 to MaintenanceUI Source -> Destination	Public SRS28.3	Public MaintenanceUI	
Realization SRS15.3 to MaintenanceUI Source -> Destination	Public SRS15.3	Public MaintenanceUI	
Realization SRS15.2 to MaintenanceUI Source -> Destination	Public SRS15.2	Public MaintenanceUI	
Realization SRS28.2 to MaintenanceUI Source -> Destination	Public SRS28.2	Public MaintenanceUI	
Realization SRS28 to MaintenanceUI Source -> Destination	Public SRS28	Public MaintenanceUI	
Realization SRS63 to	Public	Public	

Connector	Source	Target	Notes
MaintenanceUI Source -> Destination	SRS63	MaintenanceUI	
Association	Public	Private maintenanceUI	
Source -> Destination	RIUICoordinator	MaintenanceUI	

Attributes

Attribute	Notes	Constraints and tags
browseButton javax.swing.JButton Private	Variables declaration - do not modify//GEN-BEGIN:variables	*Default:*
cancelButton javax.swing.JButton Protected		*Default:*
configPathTextField javax.swing.JTextField Private		*Default:*
configurationFilesRadioButton javax.swing.JRadioButton Private		*Default:*
deleteButton javax.swing.JButton Protected		*Default:*
fileDeletionScrollPane javax.swing.JScrollPane Private		*Default:*
fileDeletionTable javax.swing.JTable Private		*Default:*
fileNameLabel javax.swing.JLabel Private		*Default:*

Attribute	Notes	Constraints and tags
fileNameTextField javax.swing.JTextField Private		*Default:*
filePath String Private	The file path.	*Default:*
fileTypeButtonGroup javax.swing.ButtonGroup Private		*Default:*
fileTypeButtonPanel javax.swing.JPanel Private		*Default:*
logFilesRadioButton javax.swing.JRadioButton Private		*Default:*

Operations

Method	Notes	Parameters
browseButtonActionPerformed() void Private	Browse button action performed. Pre-Conditions: N/A Post-Conditions: N/A	java.awt.event.ActionEvent [in] evt the evt
cancelButtonActionPerformed() void Private	Cancel button action performed. Pre-Conditions: N/A Post-Conditions: N/A	java.awt.event.ActionEvent [in] evt the evt
configurationFilesRadioButtonActionPerformed() void Private	Configuration files radio button action performed. Pre-Conditions: N/A Post-Conditions: N/A	java.awt.event.ActionEvent [in] evt the evt
deleteButtonActionPerformed() void Private	Delete button action performed. Pre-Conditions: N/A Post-Conditions: N/A	java.awt.event.ActionEvent [in] evt the evt
fileNameTextFieldActionPerformed() void Private	File name text field action performed. Pre-Conditions: N/A Post-Conditions: N/A	java.awt.event.ActionEvent [in] evt the evt
initComponents() void Private	This method is called from within the constructor to initialize the form. WARNING: Do NOT modify this code. The	

Method	Notes	Parameters
	content of this method is always regenerated by the Form Editor.	
logFilesRadioButtonActionPerformed() void Private	Log files radio button action performed. Pre-Conditions: N/A Post-Conditions: N/A	java.awt.event.ActionEvent [in] evt the evt
Static **main()** void Public	The main method. Pre-Conditions: N/A Post-Conditions: N/A	String[] [in] args the command line arguments
MaintenanceUI() Public	Creates new form MaintenanceUI.	java.awt.Frame [in] parent the parentboolean [in] modal the modal
showDialog() boolean Public	Show dialog. Pre-Conditions: N/A Post-Conditions: N/A @return true, if successful	
valueChanged() void Public	(non-Javadoc) @see javax.swing.event.ListSelectionListener#valueChanged(javax.swing.event.ListSelectionEvent)	javax.swing.event.ListSelectionEvent [in] e

5.1.5.3 Context Diagram

N/A

5.1.5.4 Variability Guide

N/A

5.1.5.5 Rationale

The C2C RI needs a mechanism for removing the files that it creates.

5.2 Test Model

5.2.1 Test Control

5.2.1.1 Test Controller

5.2.1.1.1 Primary Presentation

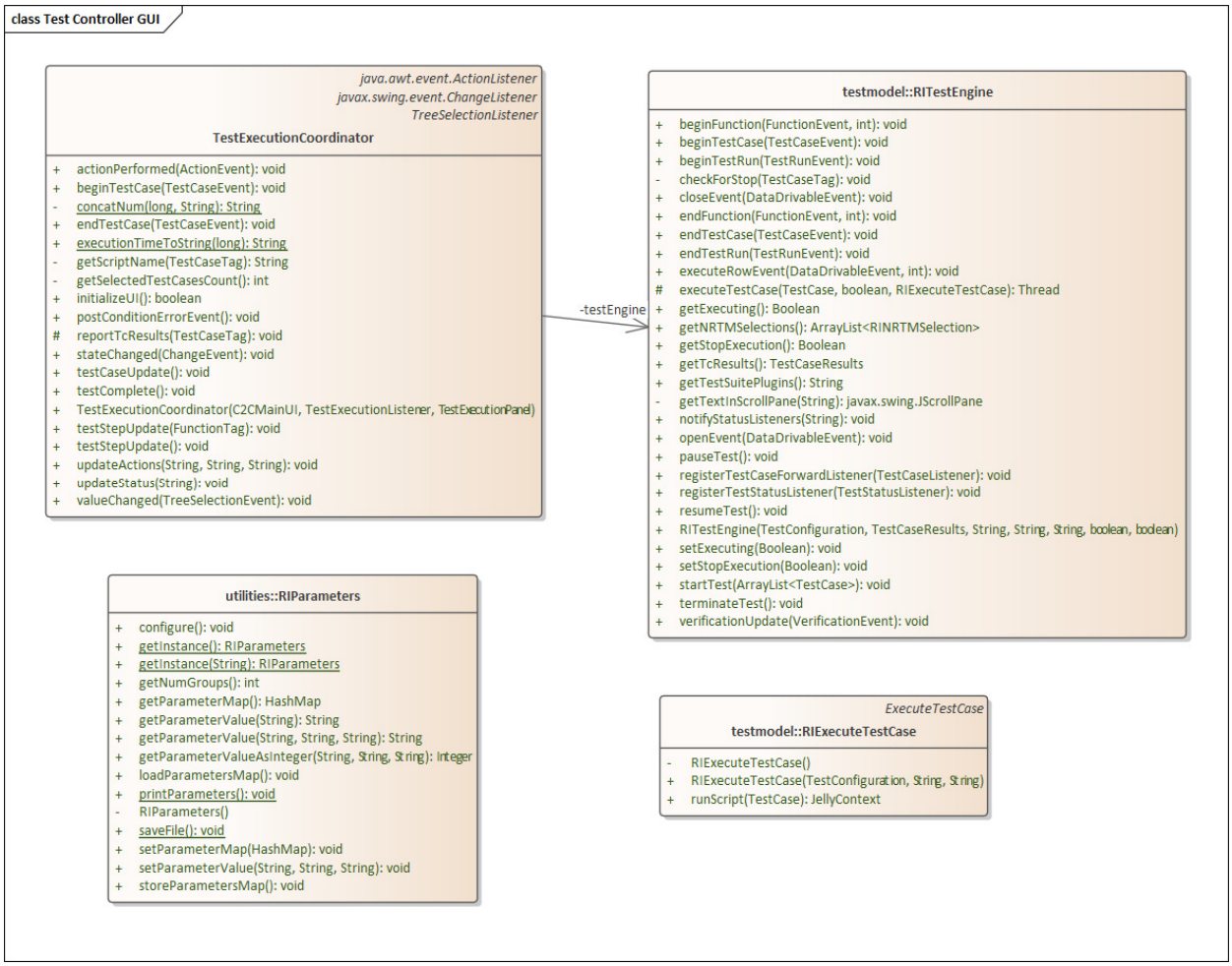

Figure 5-15 Test Controller Modules

The Test Controller modules provide mechanisms for executing test scripts when the user selects a test case to be run.

5.2.1.1.2 Element Catalog

5.2.1.1.2.1 RIExecuteTestCase

Type: **Class** ExecuteTestCase *Package:* testmodel

This class extends the Jameleon ExecuteTestCase class in order to be able to pass a URL for a script file to the underlying Jelly framework. It already had a method that accepts the URL, but Jameleon did not provide a means to utilize it.

Connections

Connector	Source	Target	Notes
Sequence JellyContext()	Public	Public	
Source -> Destination	RIExecuteTestCase	Jelly	
Association	Public	Private testConfiguration	
Source -> Destination	RIExecuteTestCase	TestConfiguration	
Generalization	Public	Public	
Source -> Destination	RIExecuteTestCase	ExecuteTestCase	
Sequence runScript()	Public	Public	
Source -> Destination	RIExecuteTestCase	Jelly	
Realization SRS32.9 to RIExecuteTestCase	Public	Public	
Source -> Destination	SRS32.9	RIExecuteTestCase	
Sequence runScript(tc)	Public	Public	
Source -> Destination	RITestEngine	RIExecuteTestCase	
Sequence	Public	Public	
Source -> Destination	Jelly	RIExecuteTestCase	
Sequence	Public	Public	
Source -> Destination	Jelly	RIExecuteTestCase	
Sequence RIExecuteTestCase(theConfig)	Public	Public	
Source -> Destination	RITestEngine	RIExecuteTestCase	

Attributes

Attribute	Notes	Constraints and tags
testCaseScriptURL URL Private	The test case script url.	*Default:* RIExecuteTestCase.class.getResource("/org/fhwa/c2cri/testmodel/TestCaseLauncher.xml")
testConfiguration TestConfiguration Private	The test configuration.	*Default:*

Operations

Method	Notes	Parameters
RIExecuteTestCase() Public	Instantiates a new rI execute test case. Pre-Conditions: N/A Post-Conditions: N/A	TestConfiguration [in] theConfig the the config
RIExecuteTestCase() Private	Instantiates a new rI execute test case. Pre-Conditions: N/A Post-Conditions: N/A	
runScript() JellyContext Public	This method is added to allow for the running of a script referenced as a URL instead of a File object. It is basically the same method as the runScrpt method included in ExecuteTestCase, with that one exception. @return - JellyContext	TestCase [in] tc the tc

5.2.1.1.2.2 RITestEngine

Type: **Class** *Package:* testmodel

Primary RI class for initiating and maintaining user initiated tests. It receives messages from the operator via the TestExecutionPanel and forwards the requests (start, pause, resume, terminate) to Jameleon.

Connections

Connector	Source	Target	Notes
Association	Public	Private testConfig	
Source -> Destination	RITestEngine	TestConfiguration	
Sequence	Public	Public	
Source -> Destination	RITestEngine	TestExecutionCoordinator	
Association	Public	Private testCaseListener	
Source -> Destination	RITestEngine	TestCaseListener	
Sequence runScript(tc)	Public	Public	
Source -> Destination	RITestEngine	RIExecuteTestCase	
Realization	Public	Public	
Source -> Destination	RITestEngine	DataDrivableListener	
Realization	Public	Public	
Source -> Destination	RITestEngine	TestRunListener	
Association	Public	Private	

Connector	Source	Target	Notes
Source -> Destination	RITestEngine	testCaseForwardListener	
		TestCaseListener	
Association	Public	Private functionListener	
Source -> Destination	RITestEngine	FunctionListener	
Association	Public	Private testRunListener	
Source -> Destination	RITestEngine	TestRunListener	
Realization	Public	Public	
Source -> Destination	RITestEngine	VerificationListener	
Association	Public	Private tcResults	
Source -> Destination	RITestEngine	TestCaseResults	
Association	Public	Private dataListener	
Source -> Destination	RITestEngine	DataDrivableListener	
Association	Public	Private verificationListener	
Source -> Destination	RITestEngine	VerificationListener	
Sequence	Public	Public	
Source -> Destination	RITestEngine	TestExecutionCoordinator	
Sequence RIExecuteTestCase(theConfig)	Public	Public	
Source -> Destination	RITestEngine	RIExecuteTestCase	
Association	Public	Private logger	
Source -> Destination	RITestEngine	RILogging	
Realization	Public	Public	
Source -> Destination	RITestEngine	TestCaseListener	
Realization	Public	Public	
Source -> Destination	RITestEngine	FunctionListener	
Association	Public	Private statusListener	
Source -> Destination	RITestEngine	TestStatusListener	
Association	Public	Private actionListener	
Source -> Destination	RITestEngine	TestActionListener	
Sequence startTest(testCaseList)	Public	Public	
Source -> Destination	TestExecutionCoordinator	RITestEngine	
Realization SRS34 to RITestEngine	Public	Public	
Source -> Destination	SRS34	RITestEngine	

Connector	Source	Target	Notes
Realization SRS89 to RITestEngine Source -> Destination	Public SRS89	Public RITestEngine	
Association Source -> Destination	Public TestExecutionCoordinator	Private testEngine RITestEngine	
Realization SRS33 to RITestEngine Source -> Destination	Public SRS33	Public RITestEngine	
Realization SRS34.1 to RITestEngine Source -> Destination	Public SRS34.1	Public RITestEngine	
Realization SRS47 to RITestEngine Source -> Destination	Public SRS47	Public RITestEngine	
Realization SRS33.1 to RITestEngine Source -> Destination	Public SRS33.1	Public RITestEngine	
Realization SRS35.3 to RITestEngine Source -> Destination	Public SRS35.3	Public RITestEngine	
Sequence RITestEngine(testConfig, tcResults, testName, testDescription, testConfigName) Source -> Destination	Public TestExecutionCoordinator	Public RITestEngine	
Realization SRS32.2 to RITestEngine Source -> Destination	Public SRS32.2	Public RITestEngine	
Realization SRS32.9 to RITestEngine Source -> Destination	Public SRS32.9	Public RITestEngine	
Realization SRS32.1 to RITestEngine Source -> Destination	Public SRS32.1	Public RITestEngine	
Realization SRS36 to RITestEngine Source -> Destination	Public SRS36	Public RITestEngine	

Attributes

Attribute	Notes	Constraints and tags
actionListener TestActionListener Private Static	The action listener.	*Default:*
dataListener DataDrivableListener Private Static	The data listener.	*Default:*
debug boolean Package Const	The debug.	*Default:* true
etcThread1 Thread Private Static	The etc thread1.	*Default:*
executing Boolean Private Static	The executing.	*Default:* false
functionListener FunctionListener Private Static	The function listener.	*Default:*
logger RILogging Private	The logger.	*Default:*
pauseExecution boolean Private	The pause execution.	*Default:* false
statusListener TestStatusListener Private Static	The status listener.	*Default:*
stopExecution Boolean Protected Static	The stop execution.	*Default:* true
t Thread Private	The t.	*Default:*

Attribute	Notes	Constraints and tags
Static		
tcResults TestCaseResults Private	The tc results.	*Default:*
testCaseForwardListener TestCaseListener Private Static	The test case forward listener.	*Default:*
testCaseListener TestCaseListener Private Static	The test case listener.	*Default:*
testConfig TestConfiguration Private	The test config.	*Default:*
testRunListener TestRunListener Private Static	The test run listener.	*Default:*
testStepFailure boolean Private	The test step failure.	*Default:* false
testStepMap HashMap<FunctionTag, Integer> Private Static	The test step map.	*Default:* new HashMap()
verificationListener VerificationListener Private Static	The verification listener.	*Default:*

Operations

Method	Notes	Parameters
beginFunction() void Public	Begin function. Pre-Conditions: N/A Post-Conditions: N/A	FunctionEvent [in] event the eventint [in] rowNum the row num
beginTestCase() void	Begin test case. Pre-Conditions: N/A Post-Conditions: N/A	TestCaseEvent [in] event

Method	Notes	Parameters
Public		the event
beginTestRun() void Public	Begin test run. Pre-Conditions: N/A Post-Conditions: N/A	TestRunEvent [in] event the event
checkForStop() void Private	Check for stop. Pre-Conditions: N/A Post-Conditions: N/A	TestCaseTag [in] tct the tct
closeEvent() void Public	Gets called before the close method of a DataDrivable.	DataDrivableEvent [in] event - a DataDrivableEvent Object
endFunction() void Public	End function. Pre-Conditions: N/A Post-Conditions: N/A	FunctionEvent [in] event the eventint [in] rowNum the row num
endTestCase() void Public	End test case. Pre-Conditions: N/A Post-Conditions: N/A	TestCaseEvent [in] event the event
endTestRun() void Public	End test run. Pre-Conditions: N/A Post-Conditions: N/A	TestRunEvent [in] event the event
executeRowEvent() void Public	Gets called before the executeDrivableRow.	DataDrivableEvent [in] event - a DataDrivableEvent Objectint [in] rowNum - the current row number being executed from the data source.
executeTestCase() Thread Protected	Execute test case. Pre-Conditions: N/A Post-Conditions: N/A @return the thread	TestCase [in] fn the fnboolean [in] debug the debugRIExecuteTestCase [in] executor the executor
getExecuting() Boolean Public	Gets the executing. @return the executing	
getStopExecution() Boolean Public	Gets the stop execution. @return the stop execution	
getTcResults() TestCaseResults	Gets the tc results. @return the tc results	

Method	Notes	Parameters
Public		
getTestSuitePlugins() String Public	Gets the test suite plugins. @return the test suite plugins	
getTextInScrollPane() javax.swing.JScrollPane Private	Gets the text in scroll pane. @return the text in scroll pane	String [in] text the text
notifyActionListeners() void Public	Notify action listeners. Pre-Conditions: N/A Post-Conditions: N/A	String [in] notification the notification
notifyStatusListeners() void Public	Notify status listeners. Pre-Conditions: N/A Post-Conditions: N/A	String [in] notification the notification
openEvent() void Public	Gets called before the open method of a DataDrivable.	DataDrivableEvent [in] event - a DataDrivableEvent Object
pauseTest() void Public	pause the previously started test being executed by Jameleon. Log this action.	
postConditionError() void Public	This method will be called from within the endFunction callback method when an error occurs within a post-condition tag. The RITestEngine will pause the Test Engine and notify the user of the condition.	
registerTestActionListener() void Public	method for registering a listener to receive notification of test actions.	TestActionListener [in] theListener the the listener
registerTestCaseForwardListener() void Public	method for registering a listener to receive notification of test actions.	TestCaseListener [in] theListener the the listener
registerTestStatusListener() void Public	method for registering a listener to receive notification of test status.	TestStatusListener [in] theListener the the listener
resumeTest() void Public	method to restart the previously paused test. Log this action.	

Method	Notes	Parameters
RITestEngine() Public	Constructor method for initializing the test engine for a test. It performs all test setup scripts available in the Test Suite. It sends test information to RILogging to initialize the test log.	TestConfiguration [in] testConfig the test configTestCaseResults [in] tcResults the tc resultsString [in] testName the test nameString [in] testDescription the test descriptionString [in] testConfigName the test config name
setExecuting() void Public	Sets the executing.	Boolean [in] executing the new executing
setStopExecution() void Public	Sets the stop execution.	Boolean [in] stopExecution the new stop execution
startTest() void Public	Notify Jameleon to start running the specified test suite, and provide its global context with user defined app and info layer parameters. Log this test action. For each main script available in the Test Suite, provide each to Jameleon to be run.	ArrayList<TestCase> [in] testCaseList the test case list
terminateTest() void Public	Notify Jameleon to terminate the currently running test. Log this action.	
testStepUpdate() void Public	This method will be called from within the endFunction callback method when an error occurs within a post-condition tag. The RITestEngine will pause the Test Engine and notify the user of the condition.	FunctionTag [in] ft the ft
verificationUpdate() void Public	Verification update. Pre-Conditions: N/A Post-Conditions: N/A	VerificationEvent [in] event the event

5.2.1.1.2.3 TestExecutionCoordinator

Type: **Class** java.awt.event.ActionListener, TreeSelectionListener *Package:* gui

The Class TestExecutionCoordinator.

Connections

Connector	Source	Target	Notes
Association	Public	Private parentObject	

Connector	Source	Target	Notes
Source -> Destination	TestExecutionCoordinator	TestExecutionListener	
Sequence startTest(testCaseList) Source -> Destination	Public TestExecutionCoordinator	Public RITestEngine	
Realization	Public	Public	
Source -> Destination	TestExecutionCoordinator	TestStatusListener	
Association	Public	Private tcModel	
Source -> Destination	TestExecutionCoordinator	TestCaseResultsTableModel	
Association	Public	Private tcReasonModel	
Source -> Destination	TestExecutionCoordinator	TestStepResultsTableModel	
Association	Public	Private testCaseResults	
Source -> Destination	TestExecutionCoordinator	TestCaseResults	
Association	Public	Private testEngine	
Source -> Destination	TestExecutionCoordinator	RITestEngine	
Realization	Public	Public	
Source -> Destination	TestExecutionCoordinator	TestCaseResultsListener	
Realization	Public	Public	
Source -> Destination	TestExecutionCoordinator	TestCaseListener	
Sequence	Public	Public	
Source -> Destination	TestExecutionCoordinator	TestExecutionPanel	
Sequence RITestEngine(testConfig, tcResults, testName, testDescription, testConfigName) Source -> Destination	Public TestExecutionCoordinator	Public RITestEngine	
Sequence	Public	Public	
Source -> Destination	TestExecutionCoordinator	TestExecutionPanel	
Association	Public	Private executionPanel	
Source -> Destination	TestExecutionCoordinator	TestExecutionPanel	

Connector	Source	Target	Notes
Realization	Public	Public	
Source -> Destination	TestExecutionCoordinator	TestActionListener	
Association	Public	Private mainUI	
Source -> Destination	TestExecutionCoordinator	C2CMainUI	
Realization SRS33.3 to TestExecutionCoordinator	Public	Public	
Source -> Destination	SRS33.3	TestExecutionCoordinator	
Realization SRS35.2.1.3 to TestExecutionCoordinator	Public	Public	
Source -> Destination	SRS35.2.1.3	TestExecutionCoordinator	
Sequence	Public	Public	
Source -> Destination	RITestEngine	TestExecutionCoordinator	
Realization SRS33.1 to TestExecutionCoordinator	Public	Public	
Source -> Destination	SRS33.1	TestExecutionCoordinator	
Realization SRS36.3 to TestExecutionCoordinator	Public	Public	
Source -> Destination	SRS36.3	TestExecutionCoordinator	
Realization SRS35.2 to TestExecutionCoordinator	Public	Public	
Source -> Destination	SRS35.2	TestExecutionCoordinator	
Realization SRS32.1 to TestExecutionCoordinator	Public	Public	
Source -> Destination	SRS32.1	TestExecutionCoordinator	
Realization SRS89 to TestExecutionCoordinator	Public	Public	
Source -> Destination	SRS89	TestExecutionCoordinator	
Realization SRS36.1 to TestExecutionCoordinator	Public	Public	
Source -> Destination	SRS36.1	TestExecutionCoordinator	
Sequence actionPerformed(e)	Public	Public	
Source -> Destination	TestExecutionPanel	TestExecutionCoordinator	
Realization SRS34.3 to TestExecutionCoordinator	Public	Public	
Source -> Destination	SRS34.3	TestExecutionCoordinator	

Connector	Source	Target	Notes
Sequence TestExecutionCoordinator (parentFrame, parentObject, executionPanel) Source -> Destination	Public TestExecutionPanel	Public TestExecutionCoordinator	
Realization SRS33 to TestExecutionCoordinator Source -> Destination	Public SRS33	Public TestExecutionCoordinator	
Realization SRS32.7 to TestExecutionCoordinator Source -> Destination	Public SRS32.7	Public TestExecutionCoordinator	
Realization SRS47 to TestExecutionCoordinator Source -> Destination	Public SRS47	Public TestExecutionCoordinator	
Realization SRS32.8 to TestExecutionCoordinator Source -> Destination	Public SRS32.8	Public TestExecutionCoordinator	
Realization SRS34.2 to TestExecutionCoordinator Source -> Destination	Public SRS34.2	Public TestExecutionCoordinator	
Realization Source -> Destination	Public SRS35.1	Public TestExecutionCoordinator	
Realization SRS35.2.1.2 to TestExecutionCoordinator Source -> Destination	Public SRS35.2.1.2	Public TestExecutionCoordinator	
Realization SRS35.2.1.1 to TestExecutionCoordinator Source -> Destination	Public SRS35.2.1.1	Public TestExecutionCoordinator	
Association Source -> Destination	Public RIUICoordinator	Private executionUICoordinator TestExecutionCoordinator	
Realization SRS32 to TestExecutionCoordinator Source -> Destination	Public SRS32	Public TestExecutionCoordinator	
Realization SRS35.2.1 to TestExecutionCoordinator Source -> Destination	Public SRS35.2.1	Public TestExecutionCoordinator	

Connector	Source	Target	Notes
Realization SRS36.2 to TestExecutionCoordinator Source -> Destination	Public SRS36.2	Public TestExecutionCoordinator	
Realization SRS33.2 to TestExecutionCoordinator Source -> Destination	Public SRS33.2	Public TestExecutionCoordinator	
Sequence Source -> Destination	Public RITestEngine	Public TestExecutionCoordinator	
Realization SRS32.9 to TestExecutionCoordinator Source -> Destination	Public SRS32.9	Public TestExecutionCoordinator	

Attributes

Attribute	Notes	Constraints and tags
EXEC_TIME int Private Static Const	The Constant EXEC_TIME.	*Default:* 5
executionPanel TestExecutionPanel Private Static	Reference to the TestExecutionPanel.	*Default:*
FAIL int Private Static Const	The Constant FAIL.	*Default:* 4
initializationResults boolean Private	The initialization results.	*Default:* false
initialRun boolean Private	The initial run.	*Default:* true
mainUI C2CMainUI Private	Reference to the C2CMainUI object.	*Default:*
parentObject TestExecutionListener Private	Reference to the parent Object that implements the TestExecutionListener interface.	*Default:*

Attribute	Notes	Constraints and tags
PASS_FAIL int Private Static Const	The Constant PASS_FAIL.	*Default:* 3
results List Private	The results.	*Default:* new ArrayList()
RI_DEFINE_STATE_NAME String Protected Static	The ri define state name.	*Default:* "Running"
RI_EXECUTE_STATE_NAME String Protected Static	The ri execute state name.	*Default:* "Paused"
RI_INIT_STATE_NAME String Protected Static	The ri init state name.	*Default:* "Initial"
RI_OPTIONS_STATE_NAME String Protected Static	The ri options state name.	*Default:* "Terminating"
RI_REPORT_STATE_NAME String Protected Static	The ri report state name.	*Default:* "Resuming"
RUN int Private Static Const	The Constant RUN.	*Default:* 3
STATUS int Private Static Const	The Constant STATUS.	*Default:* 2
STEP_NUMBER int Private Static Const	The Constant STEP_NUMBER.	*Default:* 2
TC_NUM int Private	The Constant TC_NUM.	*Default:* 0

Attribute	Notes	Constraints and tags
Static Const		
tcModel TestCaseResultsTableModel Private	The tc model.	*Default:*
tcReasonModel TestStepResultsTableModel Private	The tc reason model.	*Default:*
TEST_CASE int Private Static Const	The Constant TEST_CASE.	*Default:* 1
TEST_PROCEDURE int Private Static Const	The Constant TEST_PROCEDURE.	*Default:* 1

TestCaseComparator Comparator<TestCase>
Public
Static
	The Test case comparator.	*Default:* new Comparator<TestCase>(){
		public int compare(TestCase testCase1, TestCase testCase2){
		String testCaseName1 = testCase1.getName().toUpperCase();
		String testCaseName2 = testCase2.getName().toUpperCase();
		
		return testCaseName1.compareTo(testCaseName2);
		}
		}

testCaseResults TestCaseResults Private	The test case results.	*Default:*
testCases Map Protected	The test cases.	*Default:* new HashMap()
testComplete boolean Private	The test complete.	*Default:*

Attribute	Notes	Constraints and tags
testEngine RITestEngine Private	Reference to the RITestEnging Object.	*Default:*
testExecutionUIState String Private	Not used. To be removed.	*Default:*
TIME_STAMP int Private Static Const	The Constant TIME_STAMP.	*Default:* 0

Operations

Method	Notes	Parameters
actionPerformed() void Public	Handles actions from the TestExecutionPanel If the Close button was pressed: remove the execution panel from the users view call the testEngine terminateTest method remove this object as an actionlistener for each of the panel's buttons notify the parentObject listener that the testExecution is completed If the run button was pressed: notify the test engine that the test was started notify the parent TestExecutionListener that the test was started disable the run and resume buttons enable the pause button If the pause button was pressed: notify the test engine that the test was paused disable the run and pause buttons enable the resume button If the resume button was pressed: notify the test engine that the test should be resumed disable the resume button (the run button is already disabled) enable the pause button.	ActionEvent [in] e the action event received
beginTestCase() void Public	Begin test case. Pre-Conditions: N/A Post-Conditions: N/A	TestCaseEvent [in] event the event
Static **concatNum()** String Private	Concat num. Pre-Conditions: N/A Post-Conditions: N/A @return the string	long [in] num the numString [in] postfix the postfix
endTestCase() void Public	End test case. Pre-Conditions: N/A Post-Conditions: N/A	TestCaseEvent [in] event the event

Method	Notes	Parameters
Static **executionTimeToString()** String Public	Execution time to string. Pre-Conditions: N/A Post-Conditions: N/A @return the string	long [in] time the time
getScriptName() String Private	Gets the script name. @return the script name	TestCaseTag [in] tct the tct
getSelectedTestCasesCount() int Private	Gets the selected test cases count. @return the selected test cases count	
initializeUI() boolean Public	This method first prompts the user to complete the information requested by the TestSelectionUI dialog. Set the testComplete flag to false If the user selects OK, then an instance of the RITestEngine is created using the TestConfiguration and other parameters selected by the user. This TestExecutionCoordinator instance then registers itself as an action and status listener for the RITestEngine. Next it registers this object as a listener for button events on the TestExecutionPanel. Finally, it clears the test status and test action text areas. If the user cancels, then return false indicating that the UI was not initialized. @return - true if the intialization was successful	
postConditionErrorEvent() void Public	Post condition error event. Pre-Conditions: N/A Post-Conditions: N/A	
reportTcResults() void Protected	Report tc results. Pre-Conditions: N/A Post-Conditions: N/A	TestCaseTag [in] tct the tct
testCaseUpdate() void Public	Test case update. Pre-Conditions: N/A Post-Conditions: N/A	
testComplete() void	This method is called by the RITestEngine	

Method	Notes	Parameters
Public	when the test has completed its execution. Set the terminateButton label to Close. Disable the user's ability to press the pause and resume buttons.	
TestExecutionCoordinator() Public	Main constructor method for the the TestExecutionCoordinator Class. This class manages the TestExecutionPanel and the responses to the user's interaction with it. It also updates the panel based on events reported from the RITestEngine object. It initializes the applicable class parameters to the values provided in the parameter list.	C2CMainUI [in] parentFrame - reference to the C2CMainUI frameTestExecutionListener [in] parentObject - reference to the monitor/creating object for this class that implements the TestExecutionListener interface.TestExecutionPanel [in] executionPanel - reference to the TestExecutionPanel
testStepUpdate() void Public	Test step update. Pre-Conditions: N/A Post-Conditions: N/A	
testStepUpdate() void Public	Test step update. Pre-Conditions: N/A Post-Conditions: N/A	FunctionTag [in] ft the ft
updateActions() void Public	This method is called by the RITestEngine when there is an update in test action information. When called, append the provide update information to the actionTextArea of the executionPanel.	String [in] theUpdate - the test action update information received
updateStatus() void Public	This method is called by the RITestEngine when there is a test status update to report. Add the received status information to the statusTextArea of the TestExecutionPanel.	String [in] theUpdate - the status information
valueChanged() void Public	(non-Javadoc) @see javax.swing.event.TreeSelectionListener#valueChanged(javax.swing.event.TreeSelectionEvent)	TreeSelectionEvent [in] e

5.2.1.1.2.4 RIParameters

Type: **Class** *Package:* utilities

This class stores and manages the user designated parameters that apply to the RI. Thes parameters include the location to find user-defined test suites, locations to find user-defined tag libraries, locations to store log and configuration files, etc.

Parameters for TMDDv303c entity emulation will also be included.

Connections

Connector	Source	Target	Notes
Association	Public	Public eFile	
Source -> Destination	RIParameters	EditorFile	
Realization	Public	Public	
Source -> Destination	SRS93	RIParameters	
Realization	Public	Public	
Source -> Destination	SRS104	RIParameters	
Realization	Public	Public	
Source -> Destination	SRS94	RIParameters	
Realization	Public	Public	
Source -> Destination	SRS97	RIParameters	

Attributes

Attribute	Notes	Constraints and tags
configName String Private Static	The config name.	*Default:* DEFAULT_CONFIG_NAME
DEFAULT_CONFIG_NAME String Public Static Const	The Constant DEFAULT_CONFIG_NAME.	*Default:* "c2cri.properties"
DEFAULT_RI_USER_PARAMETER_VALUE String Public Static Const	The Constant DEFAULT_RI_USER_PARAMETER_VALUE.	*Default:* ""
eFile EditorFile Public Static	The e file.	*Default:*
RI_USER_PARAMETER String Public Static Const	The Constant RI_USER_PARAMETER.	*Default:* "C2CRIUserName"

Attribute	Notes	Constraints and tags
RI_USER_PARAMETER_GROUP String Public Static Const	The Constant RI_USER_PARAMETER_GROUP.	*Default:* "General"
RI_VERSION_NUMBER String Public Static Const	The Constant RI_VERSION_NUMBER.	*Default:* "Version 1.0"
riParameters RIParameters Private Static	The ri parameters.	*Default:*
TEST_TARGET String Public Static	The test target.	*Default:* "TestTarget"

Operations

Method	Notes	Parameters
configure() void Public	Set everything up by reading the configuration file. This method is called by {@link #getValue} and is only executed if things aren't already set up. In other words, once a file has been read in, this method will no longer do anything. To read in a new file, call {@link #clearInstance} first.	
Static **getInstance()** RIParameters Public	Gets an instance Configurator if one has already been created or creates a new one if not. @return An instance Configurator if one has already been created or creates a new one if not.	
Static **getInstance()** RIParameters Public	Gets an instance Configurator if one has already been created or creates a new one using the specified file name if not. @return An instance Configurator if one has already been created or creates a new one if not.	String [in] configFile the config file
getNumGroups() int	Gets the num groups.	

Method	Notes	Parameters
Public	@return the num groups	
getParameterMap() HashMap Public	Returns the parent map to be used on the OptionsUI. @return the parameter map	
getParameterValue() String Public	Gets the parameter value. @return the parameter value	String [in] parameterName the parameter name
getParameterValue() String Public	Gets the parameter value. @return the parameter value	String [in] group the groupString [in] parameterName the parameter nameString [in] defaultValue the default value
getParameterValueAsInteger() Integer Public	Gets the parameter value as integer. @return the parameter value as integer	String [in] group the groupString [in] parameterName the parameter nameString [in] defaultValue the default value
loadParametersMap() void Public	Loads the user selected parameters from the parameters file.	
Static **printParameters()** void Public	Prints the parameters. Pre-Conditions: N/A Post-Conditions: N/A	
RIParameters() Private	Instantiates a new rI parameters. Pre-Conditions: N/A Post-Conditions: N/A	
Static **saveFile()** void Public	Save file. Pre-Conditions: N/A Post-Conditions: N/A	
setParameterMap() void Public	Sets the Parameter Map based on the users settings on the OptionsUI.	HashMap [in] parameterMap the new parameter map
storeParametersMap() void Public	Stores the user selected parameters into a parameters file.	

Method	Notes	Parameters

5.2.1.1.3 Context Diagram

N/A

5.2.1.1.4 Variability Guide

N/A

5.2.1.1.5 Rationale

N/A

5.2.1.2 Test Sequencer

The test sequencer function is handled by the user selection of the test cases which will be executed, and the execution steps defined within the test script files.

5.3 Test Configuration

5.3.1 Test Script

5.3.1.1 Primary Presentation

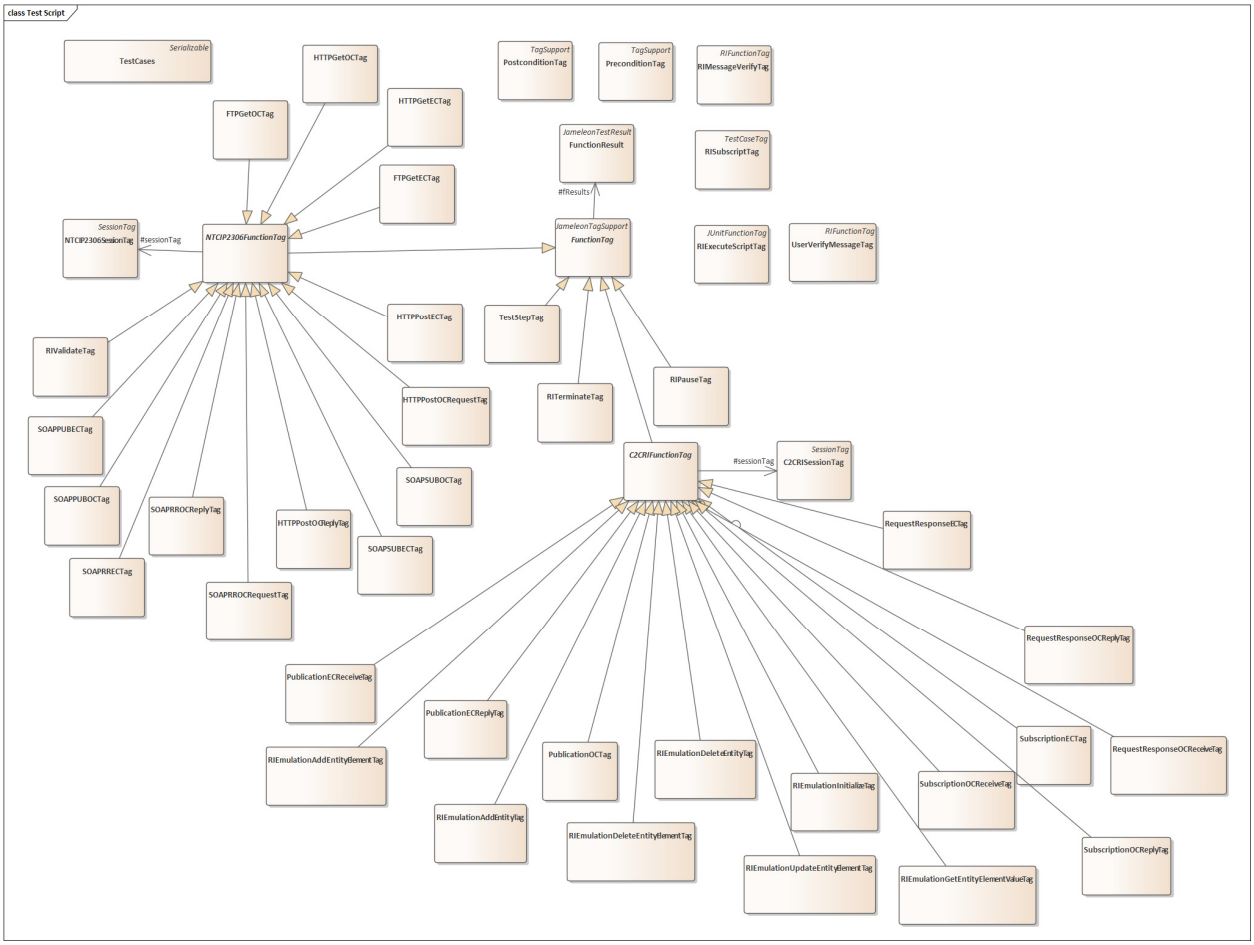

Figure 5-16 Test Script Modules

The Test Script Module includes a variety of classes that were created to support test actions specified in the test script files deployed with the standards test suites.

5.3.1.2 Element Catalog

5.3.1.2.1 TestCases

Type: **Class** Serializable *Package:* testmodel

The Class TestCases contains a collection of the test cases defined within a configuration file.

Connections

Connector	Source	Target	Notes
Association	Public	Public testCases	
Source -> Destination	TestCases	TestCase	
Association	Public	Private testCases	
Source -> Destination	TestCasesTableModel	TestCases	
Association	Public	Private testCases	
Source -> Destination	DefaultLayerParameters	TestCases	
Realization SRS30 to TestCases	Public	Public	
Source -> Destination	SRS30	TestCases	

Attributes

Attribute	Notes	Constraints and tags
baselineTestSuite String Private	The baseline test suite.	*Default:* ""
customTestSuite String Private	The custom test suite.	*Default:* ""
dataFile_Header String Public Static Const	The Constant dataFile_Header.	*Default:* "DataFile"
dataFile_Index int Private	The data file_ index.	*Default:*
description_Header String Public Static Const	The Constant description_Header.	*Default:* "Description"
description_Index int Private	The description_ index.	*Default:*
extendedTestSuite boolean Private	The extended test suite.	*Default:*
headerList ArrayList<String> Public	The header list.	*Default:*

Attribute	Notes	Constraints and tags
name_Header String Public Static Const	The Constant name_Header.	*Default:* "Name"
name_Index int Private	The name_ index.	*Default:*
scriptFile_Header String Public Static Const	The Constant scriptFile_Header.	*Default:* "ScriptFile"
scriptFile_Index int Private	The script file_ index.	*Default:*
testCases List<TestCase> Public	The test cases.	*Default:* new ArrayList<TestCase>()
testCasesMap HashMap Public	The test cases map.	*Default:*
type_Header String Public Static Const	The Constant type_Header.	*Default:* "Type"
type_Index int Private	The type_ index.	*Default:*

Operations

Method	Notes	Parameters
getName_Index() int Public	Gets the name_ index. @return the name_ index	
getText_Index() int Public	Gets the text_ index. @return the text_ index	

Method	Notes	Parameters
getTitle_Index() int Public	Gets the title_ index. @return the title_ index	
getType_Index() int Public	Gets the type_ index. @return the type_ index	
loadTestCases() List<TestCase> Private	Load test cases. Pre-Conditions: N/A Post-Conditions: N/A @return the list	URL [in] testSuiteTestCasesPath the test suite test cases pathboolean [in] customSuite the custom suite
TestCases() Private	Instantiates a new test cases. Pre-Conditions: N/A Post-Conditions: N/A	
TestCases() Public	Instantiates a new test cases. Pre-Conditions: N/A Post-Conditions: N/A	String [in] selectedTestSuite the selected test suite
TestCases() Public	Instantiates a new test cases. Pre-Conditions: N/A Post-Conditions: N/A	String [in] predefinedTestSuite the predefined test suiteString [in] customTestSuite the custom test suite
TestCases() Public	Instantiates a new test cases. Pre-Conditions: N/A Post-Conditions: N/A	URL [in] selectedTestSuite the selected test suite
TestCases() Public	Instantiates a new test cases. Pre-Conditions: N/A Post-Conditions: N/A	URL [in] predefinedTestSuite the predefined test suiteURL [in] customTestSuite the custom test suite

5.3.1.2.2 PostconditionTag

Type: **Class** TagSupport *Package:* jameleon

This tag is used to mark a group of tags as postconditions. Marking a tag as a postcondition tag guarantees the execution of the tag. If the test case fails at a tag marked as a postcondition, then the results are marked as a postcondition failure.

@jameleon.function name="postcondition"

Connections

Connector	Source	Target	Notes
Realization SRS88.1 to PostconditionTag Source -> Destination	Public SRS88.1	Public PostconditionTag	
Realization SRS89 to PostconditionTag Source -> Destination	Public SRS89	Public PostconditionTag	
Realization SRS46.2 to PostconditionTag Source -> Destination	Public SRS46.2	Public PostconditionTag	
Realization SRS88.2 to PostconditionTag Source -> Destination	Public SRS88.2	Public PostconditionTag	
Realization SRS46.1 to PostconditionTag Source -> Destination	Public SRS46.1	Public PostconditionTag	
Realization SRS47 to PostconditionTag Source -> Destination	Public SRS47	Public PostconditionTag	
Realization SRS46 to PostconditionTag Source -> Destination	Public SRS46	Public PostconditionTag	
Realization SRS88 to PostconditionTag Source -> Destination	Public SRS88	Public PostconditionTag	

Operations

Method	Notes	Parameters
doTag() void Public		XMLOutput [in] out

5.3.1.2.3 PreconditionTag

Type: **Class** TagSupport *Package:* jameleon

This tag is used to mark a group of tags as preconditions. Marking a tag as a precondition tag doesn't change the functionality of the test case. If the test case fails at a tag marked as a precondition, then the results are marked as a precondition failure.

@jameleon.function name="precondition"

Connections

Connector	Source	Target	Notes

Connector	Source	Target	Notes
Realization SRS42 to PreconditionTag Source -> Destination	Public SRS42	Public PreconditionTag	
Realization SRS45 to PreconditionTag Source -> Destination	Public SRS45	Public PreconditionTag	
Realization SRS43 to PreconditionTag Source -> Destination	Public SRS43	Public PreconditionTag	

Operations

Method	Notes	Parameters
doTag() void Public		XMLOutput [in] out

5.3.1.2.4 FunctionTag

Type: **Class** JameleonTagSupport *Package:* function

Represents an abstract function point that does a lot behind the scenes for the function point developer. The following instance variables are described:

- fResults - Results for this function point. Basically, the same information is kept in the session results as are in these results, except the function results only keep information about this function.

- functionId - This is a brief description of what this function point is used for in the test case. This is a required attribute that is set in the macro tag of each function point. This is also used for logging and results.

- log - @see org.apache.log4j.Logger an easy way to send messages about the function point to the logs.

The steps to creating an abstract function tag for a plugin are as follows:

- Implement the optional {@link #store(java.lang.String,int)} method

- Override the default behavior of {@link #setupEnvironment}, being sure to call super first. The new behavior should be to get the corresponding {@link net.sf.jameleon.SessionTag} for the plug-in if a handle on the application is kept there. Use the Jelly-provided **findAncestorWithClass(class)** method to get a handle on the parent SessionTag. The abstract FunctionTag implementation should set the handle on the application being tested to a local instance variable for easy manipulation for FunctionTag extenders. If use of the plug-in FunctionTag requires a handle on the application, then it is good practice to throw a JameleonScriptException if the an ancestor of the plug-in SessionTag isn't found.

- Write your wrapper methods to make life easier for implementers of the plug-in.</l>

 Actually, to implement a FunctionTag for a plug-in is as simple as extending this base class and nothing more. The above list is provided as recommendation only.

To implement a FunctionTag from an existing plugin, the following methods must be implemented:

- {@link #testBlock()}

The following methods help define the behaviour of the function point and should be implemented not at the plugin level, but at the customization layer the user will do:

- {@link #setup()}
- {@link #validate()}
- {@link #tearDown()}

Connections

Connector	Source	Target	Notes
Realization	Public	Public	
Source -> Destination	FunctionTag	Storable	
Association	Public	Protected addt	
Source -> Destination	FunctionTag	AbstractDataDrivableTag	
Association	Public	Protected state	
Source -> Destination	FunctionTag	StateStorer	
Association	Public	Protected fResults	
Source -> Destination	FunctionTag	FunctionResult	
Association	Public	Protected vpt	
Source -> Destination	FunctionTag	ParamTypeValidatable	
Association	Public	Protected tct	
Source -> Destination	FunctionTag	TestCaseTag	
Realization	Public	Public	
Source -> Destination	FunctionTag	BreakPoint	
Association	Public	Package assertLevelManager	
Source -> Destination	FunctionTag	AssertLevel	
Association	Public	Protected st	
Source -> Destination	FunctionTag	SessionTag	
Generalization	Public	Public	
Source -> Destination	FunctionTag	JameleonTagSupport	

Connector	Source	Target	Notes
Generalization	Public	Public	
Source -> Destination	FunctionDocTag	FunctionTag	
Generalization	Public	Public	
Source -> Destination	HttpFunctionTag	FunctionTag	
Generalization	Public	Public	
Source -> Destination	WaitTag	FunctionTag	
Realization SRS46 to FunctionTag	Public	Public	
Source -> Destination	SRS46	FunctionTag	
Generalization	Public	Public	
Source -> Destination	C2CRIFunctionTag	FunctionTag	
Realization SRS37.5 to FunctionTag	Public	Public	
Source -> Destination	SRS37.5	FunctionTag	
Realization SRS46.1 to FunctionTag	Public	Public	
Source -> Destination	SRS46.1	FunctionTag	
Association	Public	Private functionTag	
Source -> Destination	FunctionDebugDialog	FunctionTag	
Generalization	Public	Public	
Source -> Destination	RIWSDLFunctionTag	FunctionTag	
Generalization	Public	Public	
Source -> Destination	NTCIP2306FunctionTag	FunctionTag	
Generalization	Public	Public	
Source -> Destination	JUnitFunctionTag	FunctionTag	
Generalization	Public	Public	
Source -> Destination	WatijFunctionTag	FunctionTag	
Realization SRS33.1 to FunctionTag	Public	Public	
Source -> Destination	SRS33.1	FunctionTag	
Generalization	Public	Public	
Source -> Destination	RITerminateTag	FunctionTag	
Generalization	Public	Public	
Source -> Destination	MockFunctionTag	FunctionTag	
Generalization	Public	Public	
Source -> Destination	SampleTag	FunctionTag	
Generalization	Public	Public	

Connector	Source	Target	Notes
Source -> Destination	HttpUnitFunctionTag	FunctionTag	
Generalization	Public	Public	
Source -> Destination	SeleniumFunctionTag	FunctionTag	
Realization SRS45 to FunctionTag	Public	Public	
Source -> Destination	SRS45	FunctionTag	
Generalization	Public	Public	
Source -> Destination	TestStepTag	FunctionTag	
Generalization	Public	Public	
Source -> Destination	HtmlUnitFunctionTag	FunctionTag	
Generalization	Public	Public	
Source -> Destination	RIUserPromptTag	FunctionTag	
Generalization	Public	Public	
Source -> Destination	JagacyFunctionTag	FunctionTag	
Generalization	Public	Public	
Source -> Destination	AbstractSqlTag	FunctionTag	
Generalization	Public	Public	
Source -> Destination	IEFunctionTag	FunctionTag	
Generalization	Public	Public	
Source -> Destination	RIPauseTag	FunctionTag	
Generalization	Public	Public	
Source -> Destination	RIFunctionTag	FunctionTag	
Realization SRS46.2 to FunctionTag	Public	Public	
Source -> Destination	SRS46.2	FunctionTag	

Attributes

Attribute	Notes	Constraints and tags
addt AbstractDataDrivableTag Protected		*Default:*
assertLevelManager AssertLevel Package		*Default:* AssertLevel.getInstance()
breakPoint boolean Protected	Tells the GUI to stop at this functional point @jameleon.attribute required="false"	*Default:*

Attribute	Notes	Constraints and tags
expectException boolean Protected	Specifies an exception is expected. @jameleon.attribute required="false"	*Default:*
expectFailure boolean Protected	Specifies a failure is expected. @jameleon.attribute required="false"	*Default:*
fResults FunctionResult Protected	The Results for this function	*Default:*
functionDelay long Protected	The delay time to use at the end of each functional point @jameleon.attribute default="-1"	*Default:* NO_DELAY
functionId String Protected	The id of this function. This is used in the macro as a short description of the function @jameleon.attribute required="true"	*Default:*
lastFileWritten File Protected	A list of required attributes for this function point. @deprecated use getFunctionResults().setErrorFile() instead.	*Default:*
log Logger Protected Static		*Default:* Logger.getLogger(FunctionTag.class .getName())
NO_DELAY long Protected Static Const		*Default:* -1
params ArrayList Protected	A list of parameters passed to this functional point via the <param> tag.	*Default:*
postcondition boolean Protected	Specifies this is a postcondition and will therefore get executed even if a previous functional point or session tag failed. @jameleon.attribute required="false"	*Default:*

Attribute	Notes	Constraints and tags
precondition boolean Protected	Specifies this is a precondition @jameleon.attribute required="false"	*Default:*
st SessionTag Protected	The parent session tag of this functional point	*Default:*
state StateStorer Protected	Stores the state of the application.	*Default:*
tct TestCaseTag Protected	The parent test case of this functional point	*Default:*
testStep boolean Protected	Added for the C2C Reference Implementation Specifies this is a testStep	*Default:*
vpt ParamTypeValidatable Protected		*Default:* new DefaultValidParamType()

Operations

Method	Notes	Parameters
addParam() void Public	Adds a ParamTag to the list of parameters	Object [in] param - the ParamTag to add
addRequiredAttribute() void Public	Adds a required attribute to the macro tag defined for this function point. This is used to enforce a certain attribute be set in the tag of the function point. The javadoc method should be used instead if possible.	String [in] attribute - The name of the attribute to add to the required list.
assertEquals() void Public	Asserts that two Strings are equal.	String [in] message - The message to print out if the assert failsString [in] expected - The expected value.String [in] actual - The value to compare against the <code>expected</code> or the actual value.

Method	Notes	Parameters
assertEquals() void Public	Asserts that two objects are equal.	Object [in] expected - The expected value.Object [in] actual - The value to compare against the <code>expected</code> or the actual value.
assertEquals() void Public	Asserts that two objects are equal.	String [in] message - The message to print out if the assert failsObject [in] expected - The expected value.Object [in] actual - The value to compare against the <code>expected</code> or the actual value.
assertEquals() void Public	Asserts that two objects are equal. @link net.sf.jameleon.util.AssertLevel}	Object [in] expected - The expected value.Object [in] actual - The value to compare against the <code>expected</code> or the actual value.int [in] assertLevel - Only run this test under the assertLevel as described in {
assertEquals() void Public	Asserts that two Strings are equal.	String [in] expected - The expected value.String [in] actual - The value to compare against the <code>expected</code> or the actual value.
assertEquals() void Public	Asserts that two Strings are equal. @link net.sf.jameleon.util.AssertLevel}	String [in] expected - The expected value.String [in] actual - The value to compare against the <code>expected</code> or the actual value.int [in] assertLevel - Only run this test under the assertLevel as described in {

Method	Notes	Parameters
assertEquals() void Public	Asserts that two Strings are equal. @link net.sf.jameleon.util.AssertLevel}	String [in] message - The message to print out if the assert failsString [in] expected - The expected value.String [in] actual - The value to compare against the <code>expected</code> or the actual value.int [in] assertLevel - Only run this test under the assertLevel as described in {
assertEquals() void Public	Asserts that two objects are equal. @link net.sf.jameleon.util.AssertLevel}	String [in] message - The message to print out if the assert failsObject [in] expected - The expected value.Object [in] actual - The value to compare against the <code>expected</code> or the actual value.int [in] assertLevel - Only run this test under the assertLevel as described in {
assertEquals() void Public	Asserts that two floats are equal concerning a delta. If the expected value is infinity then the delta value is ignored. @link net.sf.jameleon.util.AssertLevel}	String [in] message - The message to print out if the assert failsfloat [in] expected - The expected value.float [in] actual - The value to compare against the <code>expected</code> or the actual value.float [in] delta - The amount the expected and actual can be different from each other.int [in] assertLevel - Only run this test under the assertLevel as described in {
assertEquals() void Public	Asserts that two doubles are equal concerning a delta. If the expected value is infinity then the delta value is ignored.	double [in] expected - The expected value.double [in] actual - The value to compare against the <code>expected</code> or the actual value.double [in] delta - The amount the expected and actual can be different from each other.

Method	Notes	Parameters
assertEquals() void Public	Asserts that two ints are equal.	int [in] expected - The expected value.int [in] actual - The value to compare against the <code>expected</code> or the actual value.
assertEquals() void Public	Asserts that two chars are equal.	char [in] expected - The expected value.char [in] actual - The value to compare against the <code>expected</code> or the actual value.
assertEquals() void Public	Asserts that two chars are equal.	String [in] message - The message to print out if the assert failschar [in] expected - The expected value.char [in] actual - The value to compare against the <code>expected</code> or the actual value.
assertEquals() void Public	Asserts that two chars are equal. @link net.sf.jameleon.util.AssertLevel}	char [in] expected - The expected value.char [in] actual - The value to compare against the <code>expected</code> or the actual value.int [in] assertLevel - Only run this test under the assertLevel as described in {
assertEquals() void Public	Asserts that two chars are equal. @link net.sf.jameleon.util.AssertLevel}	String [in] message - The message to print out if the assert failschar [in] expected - The expected value.char [in] actual - The value to compare against the <code>expected</code> or the actual value.int [in] assertLevel - Only run this test under the assertLevel as described in {
assertEquals() void Public	Asserts that two shorts are equal.	short [in] expected - The expected value.short [in] actual - The value to compare against the <code>expected</code> or the actual value.

Method	Notes	Parameters
assertEquals() void Public	Asserts that two shorts are equal.	String [in] message - The message to print out if the assert failsshort [in] expected - The expected value.short [in] actual - The value to compare against the <code>expected</code> or the actual value.
assertEquals() void Public	Asserts that two floats are equal concerning a delta. If the expected value is infinity then the delta value is ignored.	String [in] message - The message to print out if the assert failsfloat [in] expected - The expected value.float [in] actual - The value to compare against the <code>expected</code> or the actual value.float [in] delta - The amount the expected and actual can be different from each other.
assertEquals() void Public	Asserts that two shorts are equal. @link net.sf.jameleon.util.AssertLevel}	String [in] message - The message to print out if the assert failsshort [in] expected - The expected value.short [in] actual - The value to compare against the <code>expected</code> or the actual value.int [in] assertLevel - Only run this test under the assertLevel as described in {
assertEquals() void Public	Asserts that two bytes are equal.	String [in] message - The message to print out if the assert failsbyte [in] expected - The expected value.byte [in] actual - The value to compare against the <code>expected</code> or the actual value.
assertEquals() void Public	Asserts that two ints are equal.	String [in] message - The message to print out if the assert failsint [in] expected - The expected value.int [in] actual - The value to compare against the <code>expected</code> or the actual value.

Method	Notes	Parameters
assertEquals() void Public	Asserts that two ints are equal. @link net.sf.jameleon.util.AssertLevel}	int [in] expected - The expected value.int [in] actual - The value to compare against the <code>expected</code> or the actual value.int [in] assertLevel - Only run this test under the assertLevel as described in {
assertEquals() void Public	Asserts that two ints are equal. @link net.sf.jameleon.util.AssertLevel}	String [in] message - The message to print out if the assert failsint [in] expected - The expected value.int [in] actual - The value to compare against the <code>expected</code> or the actual value.int [in] assertLevel - Only run this test under the assertLevel as described in {
assertEquals() void Public	Asserts that two shorts are equal. @link net.sf.jameleon.util.AssertLevel}	short [in] expected - The expected value.short [in] actual - The value to compare against the <code>expected</code> or the actual value.int [in] assertLevel - Only run this test under the assertLevel as described in {
assertEquals() void Public	Asserts that two booleans are equal.	boolean [in] expected - The expected value.boolean [in] actual - The value to compare against the <code>expected</code> or the actual value.
assertEquals() void Public	Asserts that two doubles are equal concerning a delta. If the expected value is infinity then the delta value is ignored. @link net.sf.jameleon.util.AssertLevel}	String [in] message - The message to print out if the assert failsdouble [in] expected - The expected value.double [in] actual - The value to compare against the <code>expected</code> or the actual value.double [in] delta - The amount the expected and actual can be different from each other.int [in] assertLevel - Only run this test under the assertLevel as described in {

Method	Notes	Parameters
assertEquals() void Public	Asserts that two floats are equal concerning a delta. If the expected value is infinity then the delta value is ignored. @link net.sf.jameleon.util.AssertLevel}	float [in] expected - The expected value.float [in] actual - The value to compare against the <code>expected</code> or the actual value.float [in] delta - The amount the expected and actual can be different from each other.int [in] assertLevel - Only run this test under the assertLevel as described in {
assertEquals() void Public	Asserts that two doubles are equal concerning a delta. If the expected value is infinity then the delta value is ignored. @link net.sf.jameleon.util.AssertLevel}	double [in] expected - The expected value.double [in] actual - The value to compare against the <code>expected</code> or the actual value.double [in] delta - The amount the expected and actual can be different from each other.int [in] assertLevel - Only run this test under the assertLevel as described in {
assertEquals() void Public	Asserts that two longs are equal.	long [in] expected - The expected value.long [in] actual - The value to compare against the <code>expected</code> or the actual value.
assertEquals() void Public	Asserts that two longs are equal.	String [in] message - The message to print out if the assert failslong [in] expected - The expected value.long [in] actual - The value to compare against the <code>expected</code> or the actual value.
assertEquals() void Public	Asserts that two bytes are equal. @link net.sf.jameleon.util.AssertLevel}	String [in] message - The message to print out if the assert failsbyte [in] expected - The expected value.byte [in] actual - The value to compare against the <code>expected</code> or the actual value.int [in] assertLevel - Only run this test under the assertLevel as described in {

Method	Notes	Parameters
assertEquals() void	Asserts that two longs are equal.	String [in] message
Public	@link net.sf.jameleon.util.AssertLevel}	- The message to print out if the assert failslong [in] expected
		- The expected value.long [in] actual
		- The value to compare against the <code>expected</code> or the actual value.int [in] assertLevel
		- Only run this test under the assertLevel as described in {
assertEquals() void	Asserts that two bytes are equal.	byte [in] expected
Public	@link net.sf.jameleon.util.AssertLevel}	- The expected value.byte [in] actual
		- The value to compare against the <code>expected</code> or the actual value.int [in] assertLevel
		- Only run this test under the assertLevel as described in {
assertEquals() void	Asserts that two booleans are equal.	String [in] message
Public		- The message to print out if the assert failsboolean [in] expected
		- The expected value.boolean [in] actual
		- The value to compare against the <code>expected</code> or the actual value.
assertEquals() void	Asserts that two booleans are equal.	boolean [in] expected
Public	@link net.sf.jameleon.util.AssertLevel}	- The expected value.boolean [in] actual
		- The value to compare against the <code>expected</code> or the actual value.int [in] assertLevel
		- Only run this test under the assertLevel as described in {
assertEquals() void	Asserts that two booleans are equal.	String [in] message
Public	@link net.sf.jameleon.util.AssertLevel}	- The message to print out if the assert failsboolean [in] expected
		- The expected value.boolean [in] actual
		- The value to compare against the <code>expected</code> or the actual value.int [in] assertLevel
		- Only run this test under the assertLevel as described in {

Method	Notes	Parameters
assertEquals() void Public	Asserts that two bytes are equal.	byte [in] expected - The expected value.byte [in] actual - The value to compare against the <code>expected</code> or the actual value.
assertEquals() void Public	Asserts that two doubles are equal concerning a delta. If the expected value is infinity then the delta value is ignored.	String [in] message - The message to print out if the assert failsdouble [in] expected - The expected value.double [in] actual - The value to compare against the <code>expected</code> or the actual value.double [in] delta - The amount the expected and actual can be different from each other.
assertEquals() void Public	Asserts that two longs are equal. @link net.sf.jameleon.util.AssertLevel}	long [in] expected - The expected value.long [in] actual - The value to compare against the <code>expected</code> or the actual value.int [in] assertLevel - Only run this test under the assertLevel as described in {
assertEquals() void Public	Asserts that two floats are equal concerning a delta. If the expected value is infinity then the delta value is ignored.	float [in] expected - The expected value.float [in] actual - The value to compare against the <code>expected</code> or the actual value.float [in] delta - The amount the expected and actual can be different from each other.
assertFalse() void Public	Asserts that a condition is false.	boolean [in] condition - The boolean condition to test.
assertFalse() void Public	Asserts that a condition is false.	String [in] message - The message to print out if the assert failsboolean [in] condition - The boolean condition to test.
assertFalse() void Public	Asserts that a condition is false. @link net.sf.jameleon.util.AssertLevel}	boolean [in] condition - The boolean condition to test.int [in] assertLevel - Only run this test under the assertLevel as described in {

Method	Notes	Parameters
assertFalse() void Public	Asserts that a condition is false. @link net.sf.jameleon.util.AssertLevel}	String [in] message - The message to print out if the assert failsboolean [in] condition - The boolean condition to test.int [in] assertLevel - Only run this test under the assertLevel as described in {
assertMethod() void Protected	Runs the run method of a Runnable and records the results appropriately. @deprecated This method will be removed in the next major release.	Runnable [in] r - The block of code you want executed (defined in the run() method).
assertMethodWithLevel() void Protected	Runs the run method of a Runnable and records the results appropriately.	Runnable [in] r - The block of code you want executed (defined in the run() method).int [in] assertLevel - The assert level of when to run the method
assertNotNull() void Public	Asserts that an object isn't null. @link net.sf.jameleon.util.AssertLevel}	String [in] message - The message to print out if the assert failsObject [in] object - The object to test for nullint [in] assertLevel - Only run this test under the assertLevel as described in {
assertNotNull() void Public	Asserts that an object isn't null. @link net.sf.jameleon.util.AssertLevel}	Object [in] object - The object to test for nullint [in] assertLevel - Only run this test under the assertLevel as described in {
assertNotNull() void Public	Asserts that an object isn't null.	String [in] message - The message to print out if the assert failsObject [in] object - The object to test for null
assertNotNull() void Public	Asserts that an object isn't null.	Object [in] object - The object to test for null

Method	Notes	Parameters
assertNotSame() void Public	Asserts that two objects refer to the same object.	String [in] message - The message to print out if the assert failsObject [in] expected - The expected value.Object [in] actual - The value to compare against the <code>expected</code> or the actual value.
assertNotSame() void Public	Asserts that two objects refer to the same object. @link net.sf.jameleon.util.AssertLevel}	String [in] message - The message to print out if the assert failsObject [in] expected - The expected value.Object [in] actual - The value to compare against the <code>expected</code> or the actual value.int [in] assertLevel - Only run this test under the assertLevel as described in {
assertNotSame() void Public	Asserts that two objects refer to the same object.	Object [in] expected - The expected value.Object [in] actual - The value to compare against the <code>expected</code> or the actual value.
assertNotSame() void Public	Asserts that two objects refer to the same object. @link net.sf.jameleon.util.AssertLevel}	Object [in] expected - The expected value.Object [in] actual - The value to compare against the <code>expected</code> or the actual value.int [in] assertLevel - Only run this test under the assertLevel as described in {
assertNull() void Public	Asserts that an object is null. @link net.sf.jameleon.util.AssertLevel}	String [in] message - The message to print out if the assert failsObject [in] object - The object to test for nullint [in] assertLevel - Only run this test under the assertLevel as described in {
assertNull() void Public	Asserts that an object is null.	Object [in] object - The object to test for null

Method	Notes	Parameters
assertNull() void Public	Asserts that an object is null. @link net.sf.jameleon.util.AssertLevel}	Object [in] object - The object to test for nullint [in] assertLevel - Only run this test under the assertLevel as described in {
assertNull() void Public	Asserts that an object is null.	String [in] message - The message to print out if the assert failsObject [in] object - The object to test for null
assertRegexMatches() void Public	Asserts that given regex matches the provided text	String [in] text - The text to match the regex againstString [in] regex - The regex to match.
assertRegexMatches() void Public	Asserts that given regex matches the provided text	String [in] message - The message to print out if the assert failsString [in] text - The text to match the regex againstString [in] regex - The regex to match.
assertRegexMatches() void Public	Asserts that given regex matches the provided text @link net.sf.jameleon.util.AssertLevel}	String [in] text - The text to match the regex againstString [in] regex - The regex to match.int [in] assertLevel - Only run this test under the assertLevel as described in {
assertRegexMatches() void Public	Asserts that given regex matches the provided text @link net.sf.jameleon.util.AssertLevel}	String [in] message - The message to print out if the assert failsString [in] text - The text to match the regex againstString [in] regex - The regex to match.int [in] assertLevel - Only run this test under the assertLevel as described in {

Method	Notes	Parameters
assertSame() void Public	Asserts that two objects refer to the same object. @link net.sf.jameleon.util.AssertLevel}	String [in] message - The message to print out if the assert failsObject [in] expected - The expected value.Object [in] actual - The value to compare against the <code>expected</code> or the actual value.int [in] assertLevel - Only run this test under the assertLevel as described in {
assertSame() void Public	Asserts that two objects refer to the same object. @link net.sf.jameleon.util.AssertLevel}	Object [in] expected - The expected value.Object [in] actual - The value to compare against the <code>expected</code> or the actual value.int [in] assertLevel - Only run this test under the assertLevel as described in {
assertSame() void Public	Asserts that two objects refer to the same object.	String [in] message - The message to print out if the assert failsObject [in] expected - The expected value.Object [in] actual - The value to compare against the <code>expected</code> or the actual value.
assertSame() void Public	Asserts that two objects refer to the same object.	Object [in] expected - The expected value.Object [in] actual - The value to compare against the <code>expected</code> or the actual value.
assertTextContains() void Public	Asserts that two objects are equal.	String [in] text - The text to be comparedString [in] textContained - the text to be contained
assertTextContains() void Public	Asserts that two objects are equal.	String [in] message - The message to print out if the assert failsString [in] text - The text to be comparedString [in] textContained - the text to be contained

Method	Notes	Parameters
assertTextContains() void Public	Asserts that two objects are equal. @link net.sf.jameleon.util.AssertLevel}	String [in] text - The text to be comparedString [in] textContained - the text to be containedint [in] assertLevel - Only run this test under the assertLevel as described in {
assertTextContains() void Public	Asserts text is contained into some given text. @link net.sf.jameleon.util.AssertLevel}	String [in] message - The message to print out if the assert failsString [in] text - The text to be comparedString [in] textContained - the text to be containedint [in] assertLevel - Only run this test under the assertLevel as described in {
assertTextEndsWith() void Public	Asserts that the given endingText is at the end of the given text.. @link net.sf.jameleon.util.AssertLevel}	String [in] text - The text to be comparedString [in] endingText - the text that must be at the end.int [in] assertLevel - Only run this test under the assertLevel as described in {
assertTextEndsWith() void Public	Asserts that the given endingText is at the end of the given text..	String [in] message - The message to print out if the assert failsString [in] text - The text to be comparedString [in] endingText - the text that must be at the end.
assertTextEndsWith() void Public	Asserts text is contained into some given text. @link net.sf.jameleon.util.AssertLevel}	String [in] message - The message to print out if the assert failsString [in] text - The text to be comparedString [in] endingText - the text that must be at the end.int [in] assertLevel - Only run this test under the assertLevel as described in {

Method	Notes	Parameters
assertTextEndsWith() void Public	Asserts that the given endingText is at the end of the given text..	String [in] text - The text to be comparedString [in] endingText - the text that must be at the end.
assertTextStartsWith() void Public	Asserts that the given startingText begins the given text..	String [in] text - The text to be comparedString [in] startingText - the text that must be at the beginning.
assertTextStartsWith() void Public	Asserts that the given startingText begins the given text..	String [in] message - The message to print out if the assert failsString [in] text - The text to be comparedString [in] startingText - the text that must be at the beginning.
assertTextStartsWith() void Public	Asserts that the given startingText begins the given text.. @link net.sf.jameleon.util.AssertLevel}	String [in] text - The text to be comparedString [in] startingText - the text that must be at the beginning.int [in] assertLevel - Only run this test under the assertLevel as described in {
assertTextStartsWith() void Public	Asserts that the given startingText begins the given text.. @link net.sf.jameleon.util.AssertLevel}	String [in] message - The message to print out if the assert failsString [in] text - The text to be comparedString [in] startingText - the text that must be at the beginning.int [in] assertLevel - Only run this test under the assertLevel as described in {
assertTrue() void Public	Asserts that a condition is true. @link net.sf.jameleon.util.AssertLevel}	boolean [in] condition - The boolean condition to test.int [in] assertLevel - Only run this test under the assertLevel as described in {
assertTrue() void Public	Asserts that a condition is true.	boolean [in] condition - The boolean condition to test.

Method	Notes	Parameters
assertTrue() void Public	Asserts that a condition is true.	String [in] message - The message to print out if the assert failsboolean [in] condition - The boolean condition to test.
assertTrue() void Public	Asserts that a condition is true. @link net.sf.jameleon.util.AssertLevel}	String [in] message - The message to print out if the assert failsboolean [in] condition - The boolean condition to test.int [in] assertLevel - Only run this test under the assertLevel as described in {
checkParamTypes() void Protected		
cleanUp() void Protected	If this method is overridden, it is highly suggested that the super method be called as well.	
cleanUpEnvironment() void Protected		
delay() void Public		
delay() void Public		long [in] delayTime
doTag() void Public	Simply calls <code>setupEnvironment()</code>, <code>validate()</code>, <code>setup()</code> and <code>testBlock(XMLOutput out)</code>. If this method is overridden, you will need to call super.doTag() or else none of the logging or error reporting will work!	XMLOutput [in] out
fail() void Public	Fails by throwing an AssertionFailedError.	
fail() void Public	Fails by throwing an AssertionFailedError with a <code>message</code>	String [in] message - The message to display on failure.

Method	Notes	Parameters
fail() void Public	Fails by throwing an AssertionFailedError with a <code>message</code> @link net.sf.jameleon.util.AssertLevel}	int [in] assertLevel - Only run this test under the assertLevel as described in {
fail() void Public	Fails by throwing an AssertionFailedError with a <code>message</code> @link net.sf.jameleon.util.AssertLevel}	String [in] message - The message to display on failure.int [in] assertLevel - Only run this test under the assertLevel as described in {
getFunctionId() String Public	Gets the id of the function being tested. This is usally a short description of how the function point.is used in the test case. @return functionId - The name of the function	
getFunctionResults() FunctionResult Public	@return The test results of this test step	
getMatchingRegexText() String Public	Gets the matching text found in the given string. @return The matching text in the searchString	String [in] text - The text to match againstString [in] regex - The regex pattern to match. This regex pattern must have a single grouping representing the desired text.int [in] matchingGroup - The group position to match against.
getParam() Object Public	Gets the nth ParamTag. @return the ParamTag located at index	int [in] index - the index location of the desired ParamTag.
getParamLength() int Public	Gets the number of ParamTags under this tag. @return the number of ParamTags under this tag.	

Method	Notes	Parameters
getParams() List Public	Gets the list of ParamTags under this tag. @return the list of ParamTags under this tag.	
getParentTags() void Protected		
getSessionTag() SessionTag Public		
getStoreToFileName() String Public	Gets the filename to store the state of the application to. The default implementation is to simply use timestamps. If this is not the desired behavior, override this method. @return the appropriate filename which starts with ERROR- if the StateStorer Event was an Error	int [in] event - the StateStorer Event
getStringOrDefault() String Public	Gets a String by its default value if an actual value isn't found. @return a String by its default value if an actual value isn't found..	String [in] value - the value to returnString [in] defaultValue - the value to return if the <code>value</code> parameter is null
getTestCaseTag() TestCaseTag Public	Gets the parent test case tag @return the parent test case tag of this tag.	
getVariable() Object Public	Gets the variable, <code>key</code> from the test case context. This is simply a wrapper class for Jelly's own context.getVariable(); @return the value of <code>key</key>	String [in] key - the name attribute to set
getVariableAsBoolean() boolean Protected	@return a variable with name <code>key</code> from the context as a (primitive) boolean	String [in] key The variable name

Method	Notes	Parameters
getVariableAsList() List Protected	Gets a variable from context and casts it to a list. If the value isn't a List, then it creates a ArrayList and adds the value to it. @return a variable with name <code>key</code> from the context as a List even if the value is null.	String [in] key The variable name
getVariableAsString() String Protected	@return a variable with name <code>key</code> from the context as a String	String [in] key The variable name
isBreakPoint() boolean Public	Checks whether this tag is a break point. Used mostly for the GUI. @return true if this tag is a break point.	
isContextVariableNull() boolean Protected	Checks to see if the variable, <code>key</code>, is null or hasn't been set. @return true If the variable is set to null or "" then.	String [in] key - The name of the variable to be checked for a null value. This variable must be a String.
pluginTearDown() void Public	Called when the function is done being executed. Use to clean up resources created in the setup method.	
recordFunctionResult() void Protected	Adds the result for this test step into the session's results and logs this information to the logger.	long [in] startTime - The time this test step started execution.
regexMatches() boolean Public	Validates that the provided regex matches the provided text @return True if the regex matches or false if it doesn't	String [in] text - The text to match againstString [in] regex - The regex pattern to match
removeFunctionResult() void Protected		

Method	Notes	Parameters
removeVariable() void Public	Removes the variable, <code>key</code> from the test case context. This is simply a wrapper class for Jelly's own context.removeVariable();	String [in] key - the name attribute to set
setDefaultVariableValue() void Public	Sets the variable to a default value in the context. It is recommended that all variables be set in the context to take advantage of all features provided by Jamaleon. This can be used when you want the function point to work with both a CSV file and setting the attributes in the macro. The rule is if the variable is already set (via testEnvironment-Applications.properties or the corresponding CSV dile or another funcationl point in the same test case), then this won't be set. If you want to set variable in the context, without regard as to the CSV file, then simply call context.setVariable(String key, String value) (through the Jelly API). For the order context variables are set in Jameleon, please see the class description of TestCaseTag. @see TestCaseTag net.sf.jameleon.TestCaseTag	String [in] attributeName - the name attribute to setObject [in] attributeValue - the value of the attribute to set.

Method	Notes	Parameters
setDefaultVariableValue() void Public	Sets the variable to a default value in the context. It is recommended that all variables be set in the context to take advantage of all features provided by Jamaleon. This can be used when you want the function point to work with both a CSV file and setting the attributes in the macro. The rule is if the variable is already set (via testEnvironment-Applications.properties or the corresponding CSV dile or another funcational point attribute), then this won't be set. If you want to set variable in the context, without regard as to the CSV file, then simply call context.setVariable(String key, String value) (through the Jelly API). For the order context variables are set in Jameleon, please see the class description of TestCaseTag. @see TestCaseTag net.sf.jameleon.TestCaseTag	String [in] attributeName - the name attribute to setString [in] attributeValue - the value of the attribute to set.
setExpectException() void Public	Sets the value for expectException. This attrubute defaults to false.	boolean [in] expectException - a value of "yes" or "true" indicates the function tag should only pass if there is an exception.
setExpectFailure() void Public	Sets the value for expectFailure. This attrubute defaults to false, so only set it if true is set.	boolean [in] expectFailure - set to true to expect a failure. To invoke this functionality must be 'yes' or 'true'.
setFunctionDelay() void Public	Sets the delay time to occur at the end of this functional point.	long [in] functionDelay - the time in millis.
setFunctionId() void Public	Sets the id of the function being tested.	String [in] functionId - the id of the function being tested
setup() void Public	Does setup for plugin-required intialization. Functional points that implement the plugin should call the super of method before implementing their own code.	
setupEnvironment() void Protected	This method is called before anything else is called.	
setUpFunctionResults() void Protected		

Method	Notes	Parameters
setVariable() void Public	Sets the variable, <code>key</code> to the value, <code>value</code> This is simply a wrapper class for Jelly's own context.setVariable();	String [in] key - the name attribute to setObject [in] value - the value of the attribute to set.
store() void Public	Stores the current state of the object to a given <code>File</code>. The default implementation of this method does nothing. Override this method to implement plug-in specific behavior. Some examples might be: - Saving the HTML from an HTTP plug-in during a state change or error - Taking a screen shot for a GUI application during a state change event A listener is already registered for each function tag. All that is required is implementing this method. @link net.sf.jameleon.util.StateStorer event} that occured (error, state change ...).	String [in] filename the name of the if the file to store the contents to.int [in] event The {
tearDown() void Public	Called when the function is done being executed. Use to clean up resources created in the setup method.	
testBlock() void Public	Contains the actual testing of the function point. Implement this method with any code that would make the FunctionTag unique. The default implementation does nothing. Implement this method to make your function tag useful.	
traceMsg() void Protected		String [in] msg

Method	Notes	Parameters
validate() void Protected	This method is called before the testBlock method is called. If you want to override this method, call super first, followed by the custom validations that need to occur in your TestStep.	

5.3.1.2.5 TestStepTag

Type: **Class** FunctionTag *Package:* function

Used to consilidate a set of test actions into a single logical test step.

@jameleon.function name="testStep"

Connections

Connector	Source	Target	Notes
Generalization Source -> Destination	Public TestStepTag	Public FunctionTag	
Realization SRS33.1 to TestStepTag Source -> Destination	Public SRS33.1	Public TestStepTag	

Attributes

Attribute	Notes	Constraints and tags
expectedResult String Protected	The result expected from this test step. @jameleon.attribute	*Default:*
needsVerified String Protected	The needs verified by the successful completion of this test step (separated by semi-colon). @jameleon.attribute	*Default:*
passfailResult String Protected	The requirements verified by the successful completion of this test step (separated by semi-colon). @jameleon.attribute	*Default:* "True"
requirementsVerified String Protected	The requirements verified by the successful completion of this test step (separated by semi-colon). @jameleon.attribute	*Default:*

Attribute	Notes	Constraints and tags
skipped boolean Private		*Default:* false
stepNumber String Protected	The number associated with this test step. @jameleon.attribute	*Default:*

Operations

Method	Notes	Parameters
doTag() void Public	An implementation of the <code>doTag</code> method provided by the <code>TagSupport</code> class.	XMLOutput [in] out
getNeedsVerified() List<String> Public		
getRequirementsVerified() List<String> Public		
isPassFailResult() boolean Public		
isSkipped() boolean Public		
setSkipped() void Public		boolean [in] skipped
setup() void Public		
setupEnvironment() void Protected	This method is called before anything else is called.	
store() void Public		File [in] f int [in] event
testBlock() void Public		XMLOutput [in] out

5.3.1.2.6 FunctionResult

Type: **Class** JameleonTestResult *Package:* result

An implementation of @see TestResult that represents the result of a function point

Connections

Connector	Source	Target	Notes
Generalization	Public	Public	
Source -> Destination	FunctionResult	JameleonTestResult	
Association	Public	Protected fResults	
Source -> Destination	FunctionTag	FunctionResult	
Association	Public	Private res4	
Source -> Destination	FunctionResultTest	FunctionResult	
Association	Public	Private res3	
Source -> Destination	FunctionResultTest	FunctionResult	
Association	Public	Private res2	
Source -> Destination	FunctionResultTest	FunctionResult	
Realization SRS37.5 to FunctionResult	Public	Public	
Source -> Destination	SRS37.5	FunctionResult	
Association	Public	Private res1	
Source -> Destination	FunctionResultTest	FunctionResult	
Generalization	Public	Public	
Source -> Destination	CountableFunctionResult	FunctionResult	

Attributes

Attribute	Notes	Constraints and tags
postcondition boolean Protected	If the function is a postcondition	*Default:*
precondition boolean Protected	If the function is a precondition	*Default:*
resultsFileName String Protected	The name of the file where the results of the display were written to if any	*Default:*
serialVersionUID long Private		*Default:* 1L

Attribute	Notes	Constraints and tags
Static Const		
testStep boolean	If the function is a testStep	*Default:*
Protected		

Operations

Method	Notes	Parameters
FunctionResult()		FunctionalPoint [in] tag
Public		- The functional point tied to the resultsHasChildResults [in] parentTestResult
		- The parent test results to update
FunctionResult()	The default constructor	
Public		
FunctionResult()		FunctionalPoint [in] tag
Public		- The functional point tied to the results
getErrorMsg() String		
Public		
getErrorMsgPrefix() String		
Protected		
hasChildren() boolean		
Public		
isDataDriven() boolean		
Public		
isParent() boolean		
Public		
isPostcondition() boolean	Tells whether the functional point is a postcondition	
Public	@return true if this is a postcondition, otherwise false	
isPrecondition() boolean	Tells whether the functional point is a precondition	
Public	@return true if this is a precondition, otherwise false	

Method	Notes	Parameters
isTestStep() boolean Public	Tells whether the functional point is a testStep @return true if this is a testStep, otherwise false	
setPostcondition() void Public	Sets the FunctionalPoint as a postcondition	boolean [in] postcondition - set to true to make this a postcondition
setPrecondition() void Public	Sets the FunctionalPoint as a precondition	boolean [in] precondition - set to true to make this a precondition
setTestStep() void Public	Sets the FunctionalPoint as a testStep @param testSTep - set to true to make this a testStep	boolean [in] testStep
toString() String Public	@return a String representation of the function point results	
toXML() String Public	@return an XML representation of the function point results	

5.3.1.2.7 NTCIP2306FunctionTag

Type: **Class** FunctionTag *Package:* ntcip2306v109

The abstract parent class for all NTCIP2306 function tags.

Connections

Connector	Source	Target	Notes
Generalization	Public	Public	
Source -> Destination	NTCIP2306FunctionTag	FunctionTag	
Association	Public	Protected sessionTag	
Source -> Destination	NTCIP2306FunctionTag	NTCIP2306SessionTag	
Generalization	Public	Public	
Source -> Destination	HTTPGetOCTag	NTCIP2306FunctionTag	
Generalization	Public	Public	

Connector	Source	Target	Notes
Source -> Destination	HTTPPostOCRequestTag	NTCIP2306FunctionTag	
Generalization	Public	Public	
Source -> Destination	SOAPRROCRequestTag	NTCIP2306FunctionTag	
Generalization	Public	Public	
Source -> Destination	SOAPRRECTag	NTCIP2306FunctionTag	
Generalization	Public	Public	
Source -> Destination	FTPGetOCTag	NTCIP2306FunctionTag	
Generalization	Public	Public	
Source -> Destination	SOAPSUBECTag	NTCIP2306FunctionTag	
Generalization	Public	Public	
Source -> Destination	HTTPPostOCReplyTag	NTCIP2306FunctionTag	
Generalization	Public	Public	
Source -> Destination	HTTPGetECTag	NTCIP2306FunctionTag	
Realization SRS39 to NTCIP2306FunctionTag	Public	Public	
Source -> Destination	SRS39	NTCIP2306FunctionTag	
Generalization	Public	Public	
Source -> Destination	SOAPRROCReplyTag	NTCIP2306FunctionTag	
Generalization	Public	Public	
Source -> Destination	HTTPPostECTag	NTCIP2306FunctionTag	
Generalization	Public	Public	
Source -> Destination	FTPGetECTag	NTCIP2306FunctionTag	
Generalization	Public	Public	
Source -> Destination	SOAPPUBECTag	NTCIP2306FunctionTag	
Generalization	Public	Public	
Source -> Destination	RIValidateTag	NTCIP2306FunctionTag	
Realization SRS38 to NTCIP2306FunctionTag	Public	Public	
Source -> Destination	SRS38	NTCIP2306FunctionTag	
Generalization	Public	Public	
Source -> Destination	SOAPSUBOCTag	NTCIP2306FunctionTag	
Generalization	Public	Public	
Source -> Destination	SOAPPUBOCTag	NTCIP2306FunctionTag	

Attributes

Attribute	Notes	Constraints and tags
sessionTag NTCIP2306SessionTag Protected	the session tag reference object	*Default:*
STORE_SOURCE_CONFIG_NAME String Public Static Const	constant	*Default:* "RI.storeSourceOnStateChange"
storeSourceOnStateChange boolean Protected	flag for whether to store the source on state change	*Default:*

Operations

Method	Notes	Parameters
initializeReturnParameters() void Public	All NTCIP2306 function tags are capable of storing context variables. Initialize those variables here.	
setupEnvironment() void Public	Gets the references of the session and the test results from the parent TestCase This method gets called once all attributes are set from the macro language. Any required setup should go here.	

5.3.1.2.8 NTCIP2306SessionTag

Type: **Class** SessionTag *Package:* ntcip2306v109

A Session tag for the C2CRI plug-in. An example of its use might:

<pre><source> <testcase xmlns="jelly:jameleon"> <C2CRI-session infoStd="tmdd-v3.0" beginSession="true"> <C2CRI-assert-element functionId="Check that the data element is the input value." element="Center Name" value = "${TMC_Name}"/> </C2CRI-session> </testcase>

</source></pre>

@jameleon.function name="NTCIP2306-session"

Connections

Connector	Source	Target	Notes
Generalization	Public	Public	
Source -> Destination	NTCIP2306SessionTag	SessionTag	

Connector	Source	Target	Notes
Association	Public	Private wsdlInstance	
Source -> Destination	NTCIP2306SessionTag	RIWSDL	
Association	Public	Private theService	
Source -> Destination	NTCIP2306SessionTag	NTCIP2306Controller	
Realization SRS39 to NTCIP2306SessionTag	Public	Public	
Source -> Destination	SRS39	NTCIP2306SessionTag	
Realization SRS38 to NTCIP2306SessionTag	Public	Public	
Source -> Destination	SRS38	NTCIP2306SessionTag	
Association	Public	Protected sessionTag	
Source -> Destination	NTCIP2306FunctionTag	NTCIP2306SessionTag	

Attributes

Attribute	Notes	Constraints and tags
appStd String Protected	@jameleon.attribute contextName="appStd"	*Default:* "NTCIP2306v01_69"
riMode String Protected	@jameleon.attribute contextName="RIMode"	*Default:*
subPubMapping String Protected	A unique identifier for this test case. @jameleon.attribute	*Default:*
testCaseIdentifier String Protected	A unique identifier for this test case. @jameleon.attribute required="true" contextName="testCaseIdentifier"	*Default:*
theService NTCIP2306Controller Private Static	the Service object managed by this Session	*Default:*
wsdlInstance RIWSDL Private		*Default:*
wsdlURL String Protected	@jameleon.attribute contextName="RI_WEBSERVICEURL"	*Default:*

Attribute	Notes	Constraints and tags

Operations

Method	Notes	Parameters
getAppStd() String Public	Gets the application Layer Standard @return the application layer standard selected.	
getRequestAppStd() String Protected	Gets the application layer standard to request @return The application layer standard to use in startApplication.	
getTheCenterService() NTCIP2306Controller Public	getter for the service @return the NTCIP2306 controller	
getWsdlInstance() RIWSDL Public	getter for the WSDL @return the WSDL	
setUpSession() void Public	Sets the up session. Pre-Conditions: N/A Post-Conditions: N/A	
startApplication() void Public	Ensures that things are started off correctly.	
subPubStringMapConverter() HashMap<OperationIdentifier, OperationIdentifier> Private	converts a string which represents a subscription publication mapping to a map of operation identifiers @return a map representing the subscription publication relationships	String [in] mappingString â€“ a string representing the map between the subscription and publication operation. ; Separates a set of subscription publication operation pairs. : Separates the subscription from a publication operation. , divides the service name port and operation names that represent a particular subscription or publicationString [in] centerMode â€“ the mode of the RI
tearDownSession() void Public	Tear down session. Pre-Conditions: N/A Post-Conditions: N/A	

Method	Notes	Parameters

5.3.1.2.9 FTPGetECTag

Type: **Class** NTCIP2306FunctionTag *Package:* tags

Perform an FTP Get Operation as an EC.

@jameleon.function name="FTP-GET-EC" type="action"

@jameleon.step Send the message get request

Connections

Connector	Source	Target	Notes
Generalization	Public	Public	
Source -> Destination	FTPGetECTag	NTCIP2306FunctionTag	
Realization SRS39.5 to FTPGetECTag	Public	Public	
Source -> Destination	SRS39.5	FTPGetECTag	
Realization	Public	Public	
Source -> Destination	SRS39.10	FTPGetECTag	

Attributes

Attribute	Notes	Constraints and tags
OPERATIONNAME String Protected	The name of the dialog associated with this request @jameleon.attribute required="true"	*Default:*
PORTNAME String Protected	The name of the dialog associated with this request @jameleon.attribute required="true"	*Default:*
SERVICENAME String Protected	The name of the dialog associated with this request @jameleon.attribute required="true"	*Default:*
WSDLFILE String Protected	The name of the dialog associated with this request @jameleon.attribute	*Default:*

Operations

Method	Notes	Parameters
testBlock() void Public	Test block. Pre-Conditions: N/A Post-Conditions: N/A	

5.3.1.2.10 FTPGetOCTag

Type: **Class** NTCIP2306FunctionTag *Package:* tags

Perform an FTP GET operation as an OC.

@jameleon.function name="FTP-GET-OC" type="action"

@jameleon.step Respond to the message get request

Connections

Connector	Source	Target	Notes
Generalization Source -> Destination	Public FTPGetOCTag	Public NTCIP2306FunctionTag	
Realization SRS38.4 to FTPGetOCTag Source -> Destination	Public SRS38.4	Public FTPGetOCTag	
Realization SRS38.9 to FTPGetOCTag Source -> Destination	Public SRS38.9	Public FTPGetOCTag	

Attributes

Attribute	Notes	Constraints and tags
OPERATIONNAME String Protected	The name of the dialog associated with this request @jameleon.attribute required="true"	*Default:*
PORTNAME String Protected	The name of the dialog associated with this get request @jameleon.attribute required="true"	*Default:*
RESPONSEMESSAGE String Protected	The name of the dialog associated with this request @jameleon.attribute required="true"	*Default:*
SERVICENAME String	The name of the dialog associated with this get request	*Default:*

Attribute	Notes	Constraints and tags
Protected	@jameleon.attribute required="true"	
SKIPENCODING String Protected	A flag indicating whether the system should not apply encoding to the response message provided. @jameleon.attribute	*Default:* "FALSE"

Operations

Method	Notes	Parameters
testBlock() void Public	Test block. Pre-Conditions: N/A Post-Conditions: N/A	

5.3.1.2.11 HTTPGetECTag

Type: **Class** NTCIP2306FunctionTag *Package:* tags

Perform an HTTP Get Operation as an EC.

@jameleon.function name="HTTP-GET-EC" type="action"

@jameleon.step Send the message get request

Connections

Connector	Source	Target	Notes
Generalization	Public	Public	
Source -> Destination	HTTPGetECTag	NTCIP2306FunctionTag	
Realization	Public	Public	
Source -> Destination	SRS39.10	HTTPGetECTag	
Realization SRS39.5 to HTTPGetECTag	Public	Public	
Source -> Destination	SRS39.5	HTTPGetECTag	

Attributes

Attribute	Notes	Constraints and tags
OPERATIONNAME String Protected	The name of the dialog associated with this request @jameleon.attribute required="true"	*Default:*
PORTNAME String Protected	The name of the dialog associated with this request	*Default:*

Attribute	Notes	Constraints and tags
	@jameleon.attribute required="true"	
SERVICENAME String Protected	The name of the dialog associated with this request @jameleon.attribute required="true"	*Default:*

Operations

Method	Notes	Parameters
testBlock() void Public	Test block. Pre-Conditions: N/A Post-Conditions: N/A	

5.3.1.2.12 HTTPGetOCTag

Type: **Class** NTCIP2306FunctionTag *Package:* tags

Perform an HTTP Get operation as an OC.

@jameleon.function name="HTTP-GET-OC" type="action"

@jameleon.step Send the message get request

Connections

Connector	Source	Target	Notes
Generalization Source -> Destination	Public HTTPGetOCTag	Public NTCIP2306FunctionTag	
Realization SRS38.4 to HTTPGetOCTag Source -> Destination	Public SRS38.4	Public HTTPGetOCTag	
Realization SRS38.9 to HTTPGetOCTag Source -> Destination	Public SRS38.9	Public HTTPGetOCTag	

Attributes

Attribute	Notes	Constraints and tags
OPERATIONNAME String Protected	The name of the dialog associated with this request @jameleon.attribute required="true"	*Default:*
PORTNAME String Protected	The name of the dialog associated with this get request	*Default:*

Attribute	Notes	Constraints and tags
	@jameleon.attribute required="true"	
RESPONSEMESSAGE String Protected	The name of the dialog associated with this request @jameleon.attribute required="true"	*Default:*
SERVICENAME String Protected	The name of the dialog associated with this get request @jameleon.attribute required="true"	*Default:*
SKIPENCODING String Protected	A flag indicating whether the system should not apply encoding to the response message provided. @jameleon.attribute	*Default:* "FALSE"

Operations

Method	Notes	Parameters
testBlock() void Public	Test block. Pre-Conditions: N/A Post-Conditions: N/A	

5.3.1.2.13 HTTPPostECTag

Type: **Class** NTCIP2306FunctionTag *Package:* tags

Perform a HTTP Post operation as an EC.

@jameleon.function name="HTTP-POST-EC" type="action"

@jameleon.step Send the message POST request

Connections

Connector	Source	Target	Notes
Generalization	Public	Public	
Source -> Destination	HTTPPostECTag	NTCIP2306FunctionTag	
Realization	Public	Public	
Source -> Destination	SRS39.10	HTTPPostECTag	
Realization SRS39.5 to HTTPPostECTag	Public	Public	
Source -> Destination	SRS39.5	HTTPPostECTag	

Attributes

Attribute	Notes	Constraints and tags
OPERATIONNAME String Protected	The name of the dialog associated with this request @jameleon.attribute required="true"	*Default:*
PORTNAME String Protected	The name of the dialog associated with this request @jameleon.attribute required="true"	*Default:*
REQUESTMESSAGE String Protected	The message being sent as a request. @jameleon.attribute required="true"	*Default:*
SERVICENAME String Protected	The name of the dialog associated with this request @jameleon.attribute required="true"	*Default:*
SKIPENCODING String Protected	A flag indicating whether the system should not apply encoding to the response message provided. @jameleon.attribute	*Default:* "FALSE"

Operations

Method	Notes	Parameters
testBlock() void Public	Test block. Pre-Conditions: N/A Post-Conditions: N/A	

5.3.1.2.14 HTTPPostOCReplyTag

Type: **Class** NTCIP2306FunctionTag *Package:* tags

Send the Post Operation Response

@jameleon.function name="HTTP-POST-OC-REPLY" type="action"

@jameleon.step Respond to the message Post request

Connections

Connector	Source	Target	Notes
Generalization	Public	Public	
Source -> Destination	HTTPPostOCReplyTag	NTCIP2306FunctionTag	
Realization SRS38.4 to	Public	Public	

Connector	Source	Target	Notes
HTTPPostOCReplyTag Source -> Destination	SRS38.4	HTTPPostOCReplyTag	

Attributes

Attribute	Notes	Constraints and tags
OPERATIONNAME String Protected	The name of the dialog associated with this request @jameleon.attribute required="true"	*Default:*
PORTNAME String Protected	The name of the dialog associated with this get request @jameleon.attribute required="true"	*Default:*
RESPONSEMESSAGE String Protected	The name of the dialog associated with this request @jameleon.attribute required="true"	*Default:*
SERVICENAME String Protected	The name of the dialog associated with this get request @jameleon.attribute required="true"	*Default:*
SKIPENCODING String Protected	A flag indicating whether the system should not apply encoding to the response message provided. @jameleon.attribute	*Default:* "FALSE"

Operations

Method	Notes	Parameters
testBlock() void Public	Test block. Pre-Conditions: N/A Post-Conditions: N/A	

5.3.1.2.15 HTTPPostOCRequestTag

Type: **Class** NTCIP2306FunctionTag *Package:* tags

Get a Post Operation Request.

@jameleon.function name="HTTP-POST-OC-REQUEST" type="action"

@jameleon.step Respond to the message Post request

Connections

Connector	Source	Target	Notes
Generalization Source -> Destination	Public HTTPPostOCRequestTag	Public NTCIP2306FunctionTag	
Realization SRS38.9 to HTTPPostOCRequestTag Source -> Destination	Public SRS38.9	Public HTTPPostOCRequestTag	

Attributes

Attribute	Notes	Constraints and tags
OPERATIONNAME String Protected	The name of the dialog associated with this request @jameleon.attribute required="true"	*Default:*
PORTNAME String Protected	The name of the dialog associated with this get request @jameleon.attribute required="true"	*Default:*
SERVICENAME String Protected	The name of the dialog associated with this get request @jameleon.attribute required="true"	*Default:*

Operations

Method	Notes	Parameters
testBlock() void Public	Test block. Pre-Conditions: N/A Post-Conditions: N/A	

5.3.1.2.16 RIValidateTag

Type: **Class** NTCIP2306FunctionTag *Package:* tags

Validates the requested message

@jameleon.function name="ri-validate" type="action"

@jameleon.step Validate the requested message

Connections

Connector	Source	Target	Notes
Generalization Source -> Destination	Public RIValidateTag	Public NTCIP2306FunctionTag	

Connector	Source	Target	Notes
Realization SRS38.6 to RIValidateTag Source -> Destination	Public SRS38.6	Public RIValidateTag	
Realization SRS39.7 to RIValidateTag Source -> Destination	Public SRS39.7	Public RIValidateTag	

Attributes

Attribute	Notes	Constraints and tags
messageName String Protected	The name of the message to validate @jameleon.attribute contextName="RIMessageName" required="true"	*Default:*
useTheProvidedWSDL boolean Protected	Future: Flag indicating whether the message should be validated against the testConfig WSDL or the Test Suite WSDL. @jameleon.attribute	*Default:*

Operations

Method	Notes	Parameters
testBlock() void Public	Test block. Pre-Conditions: N/A Post-Conditions: N/A	

5.3.1.2.17 SOAPPUBECTag

Type: **Class** NTCIP2306FunctionTag *Package:* tags

SOAP Message Publication for an EC.

@jameleon.function name="SOAP-PUB-EC" type="action"

@jameleon.step Receive the SOAP Subscription request

Connections

Connector	Source	Target	Notes
Generalization Source -> Destination	Public SOAPPUBECTag	Public NTCIP2306FunctionTag	
Realization SRS39.4 to SOAPPUBECTag Source -> Destination	Public SRS39.4	Public SOAPPUBECTag	

Connector	Source	Target	Notes
Realization	Public	Public	
Source -> Destination	SRS39.10	SOAPPUBECTag	

Attributes

Attribute	Notes	Constraints and tags
PUBLICATIONOPERATIONNAME String Protected	The name of the dialog associated with this request @jameleon.attribute required="true"	*Default:*
PUBLICATIONPORTNAME String Protected	The name of the port associated with this request @jameleon.attribute required="true"	*Default:*
PUBLICATIONSERVICENAME String Protected	The name of the service associated with this request @jameleon.attribute required="true"	*Default:*
SKIPENCODING String Protected	A flag indicating whether the system should not apply encoding to the response message provided. @jameleon.attribute	*Default:* "FALSE"

Operations

Method	Notes	Parameters
testBlock() void Public	Test block. Pre-Conditions: N/A Post-Conditions: N/A	

5.3.1.2.18 SOAPPUBOCTag

Type: **Class** NTCIP2306FunctionTag *Package:* tags

SOAP Message Publication for an OC

@jameleon.function name="SOAP-PUB-OC" type="action"

@jameleon.step Send the message SOAP Publication

Connections

Connector	Source	Target	Notes
Generalization	Public	Public	

Connector	Source	Target	Notes
Source -> Destination	SOAPPUBOCTag	NTCIP2306FunctionTag	
Realization SRS38.5 to SOAPPUBOCTag Source -> Destination	Public SRS38.5	Public SOAPPUBOCTag	
Realization SRS38.9 to SOAPPUBOCTag Source -> Destination	Public SRS38.9	Public SOAPPUBOCTag	

Attributes

Attribute	Notes	Constraints and tags
PUBLICATIONMESSAGE String Protected	The message being sent as a Subscription request. @jameleon.attribute required="true"	*Default:*
PUBLICATIONOPERATIONNAME String Protected	The name of the dialog associated with this publication @jameleon.attribute required="true"	*Default:*
PUBLICATIONPORTNAME String Protected	The name of the port associated with this request @jameleon.attribute required="true"	*Default:*
PUBLICATIONSERVICENAME String Protected	The name of the service associated with this request @jameleon.attribute required="true"	*Default:*
SKIPENCODING String Protected	A flag indicating whether the system should not apply encoding to the response message provided. @jameleon.attribute	*Default:* "FALSE"

Operations

Method	Notes	Parameters
testBlock() void Public	Test block. Pre-Conditions: N/A Post-Conditions: N/A	

5.3.1.2.19 SOAPRRECTag

Type: **Class** NTCIP2306FunctionTag *Package:* tags

SOAP message Request Response for an EC

@jameleon.function name="SOAP-RR-EC" type="action"

@jameleon.step Send the message POST request

Connections

Connector	Source	Target	Notes
Generalization	Public	Public	
Source -> Destination	SOAPRRECTag	NTCIP2306FunctionTag	
Realization SRS39.5 to SOAPRRECTag Source -> Destination	Public SRS39.5	Public SOAPRRECTag	
Realization	Public	Public	
Source -> Destination	SRS39.10	SOAPRRECTag	

Attributes

Attribute	Notes	Constraints and tags
OPERATIONNAME String Protected	The name of the dialog associated with this request @jameleon.attribute required="true"	*Default:*
PORTNAME String Protected	The name of the dialog associated with this request @jameleon.attribute required="true"	*Default:*
REQUESTMESSAGE String Protected	The message being sent as a request. @jameleon.attribute required="true"	*Default:*
SERVICENAME String Protected	The name of the dialog associated with this request @jameleon.attribute required="true"	*Default:*
SKIPENCODING String Protected	A flag indicating whether the system should not apply encoding to the response message provided. @jameleon.attribute	*Default:* "FALSE"

Operations

Method	Notes	Parameters

Method	Notes	Parameters
testBlock() void Public		

5.3.1.2.20 SOAPRROCReplyTag

Type: **Class** NTCIP2306FunctionTag *Package:* tags

Send the message get request

@jameleon.function name="SOAP-RR-OC-REPLY" type="action"

@jameleon.step Respond to the message Post request

Connections

Connector	Source	Target	Notes
Generalization Source -> Destination	Public SOAPRROCReplyTag	Public NTCIP2306FunctionTag	
Realization SRS38.4 to SOAPRROCReplyTag Source -> Destination	Public SRS38.4	Public SOAPRROCReplyTag	

Attributes

Attribute	Notes	Constraints and tags
OPERATIONNAME String Protected	The name of the dialog associated with this request @jameleon.attribute required="true"	*Default:*
PORTNAME String Protected	The name of the dialog associated with this get request @jameleon.attribute required="true"	*Default:*
RESPONSEMESSAGE String Protected	The name of the dialog associated with this request @jameleon.attribute required="true"	*Default:*
SERVICENAME String Protected	The name of the dialog associated with this get request @jameleon.attribute required="true"	*Default:*
SKIPENCODING String Protected	A flag indicating whether the system should not apply encoding to the response message provided.	*Default:* "FALSE"

Attribute	Notes	Constraints and tags
	@jameleon.attribute	

Operations

Method	Notes	Parameters
testBlock() void Public	Test block. Pre-Conditions: N/A Post-Conditions: N/A	

5.3.1.2.21 SOAPRROCRequestTag

Type: **Class** NTCIP2306FunctionTag *Package:* tags

Send the message get request

@jameleon.function name="SOAP-RR-OC-REQUEST" type="action"

@jameleon.step Receive the SOAP request

Connections

Connector	Source	Target	Notes
Generalization Source -> Destination	Public SOAPRROCRequestTag	Public NTCIP2306FunctionTag	
Realization SRS38.9 to SOAPRROCRequestTag Source -> Destination	Public SRS38.9	Public SOAPRROCRequestTag	

Attributes

Attribute	Notes	Constraints and tags
OPERATIONNAME String Protected	The name of the dialog associated with this request @jameleon.attribute required="true"	*Default:*
PORTNAME String Protected	The name of the dialog associated with this get request @jameleon.attribute required="true"	*Default:*
SERVICENAME String Protected	The name of the dialog associated with this get request @jameleon.attribute required="true"	*Default:*

Operations

Method	Notes	Parameters
testBlock() void Public	Test block. Pre-Conditions: N/A Post-Conditions: N/A	

5.3.1.2.22 SOAPSUBECTag

Type: **Class** NTCIP2306FunctionTag *Package:* tags

Send the message get request

@jameleon.function name="SOAP-SUB-EC" type="action"

@jameleon.step Send the message SOAP Subscription

Connections

Connector	Source	Target	Notes
Generalization Source -> Destination	Public SOAPSUBECTag	Public NTCIP2306FunctionTag	
Realization Source -> Destination	Public SRS39.10	Public SOAPSUBECTag	
Realization SRS39.6 to SOAPSUBECTag Source -> Destination	Public SRS39.6	Public SOAPSUBECTag	
Realization SRS39.5 to SOAPSUBECTag Source -> Destination	Public SRS39.5	Public SOAPSUBECTag	

Attributes

Attribute	Notes	Constraints and tags
SKIPENCODING String Protected	A flag indicating whether the system should not apply encoding to the response message provided. @jameleon.attribute	*Default:* "FALSE"
SUBSCRIPTIONMESSAGE String Protected	The message being sent as a Subscription request. @jameleon.attribute required="true"	*Default:*
SUBSCRIPTIONOPERATIONNAME String Protected	The name of the dialog associated with this request @jameleon.attribute required="true"	*Default:*

Attribute	Notes	Constraints and tags
SUBSCRIPTIONPORTNAME String Protected	The name of the dialog associated with this request @jameleon.attribute required="true"	*Default:*
SUBSCRIPTIONSERVICENAME String Protected	The name of the dialog associated with this request @jameleon.attribute required="true"	*Default:*

Operations

Method	Notes	Parameters
testBlock() void Public	Test block. Pre-Conditions: N/A Post-Conditions: N/A	

5.3.1.2.23 SOAPSUBOCTag

Type: **Class** NTCIP2306FunctionTag *Package:* tags

Send the message get request.

@jameleon.function name="SOAP-SUB-OC" type="action"

@jameleon.step Receive the SOAP Subscription request

Connections

Connector	Source	Target	Notes
Generalization Source -> Destination	Public SOAPSUBOCTag	Public NTCIP2306FunctionTag	
Realization SRS38.9 to SOAPSUBOCTag Source -> Destination	Public SRS38.9	Public SOAPSUBOCTag	

Attributes

Attribute	Notes	Constraints and tags
SKIPENCODING String Protected	A flag indicating whether the system should not apply encoding to the response message provided. @jameleon.attribute	*Default:* "FALSE"
SUBSCRIPTIONOPERATIONNAME String	The name of the dialog associated with this request	*Default:*

Attribute	Notes	Constraints and tags
Protected	@jameleon.attribute required="true"	
SUBSCRIPTIONPORTNAME String	The name of the dialog associated with this get request	*Default:*
Protected	@jameleon.attribute required="true"	
SUBSCRIPTIONSERVICENAME String	The name of the dialog associated with this get request	*Default:*
Protected	@jameleon.attribute required="true"	

Operations

Method	Notes	Parameters
testBlock() void Public	Test block. Pre-Conditions: N/A Post-Conditions: N/A	

5.3.1.2.24 RIExecuteScriptTag

Type: **Class** JUnitFunctionTag *Package:* tags

Used to acceptance test Jameleon tags. Executes a Jameleon script and checks various outputs.

@jameleon.function name="ri-execute-script" type="action"

@jameleon.step Execute the given script

@jameleon.step Validate that the script ran correctly

Connections

Connector	Source	Target	Notes
Association	Public	Protected listener	
Source -> Destination	RIExecuteScriptTag	TestCaseListener	
Association	Public	Protected result	
Source -> Destination	RIExecuteScriptTag	TestCaseResult	
Association	Public	Protected jmln	
Source -> Destination	RIExecuteScriptTag	ExecuteTestCase	
Association	Public	Protected originalCounter	
Source -> Destination	RIExecuteScriptTag	TestCaseCounter	
Generalization	Public	Public	
Source -> Destination	RIExecuteScriptTag	JUnitFunctionTag	

Connector	Source	Target	Notes
Realization SRS44 to RIExecuteScriptTag Source -> Destination	Public SRS44	Public RIExecuteScriptTag	

Attributes

Attribute	Notes	Constraints and tags
checkOutcomeFailed boolean Protected	Pass only if the script failed @jameleon.attribute	*Default:*
checkOutcomePassed boolean Protected	Pass only if the script passed @jameleon.attribute	*Default:*
DEBUG boolean Protected Static Const		*Default:* false
denyAll DenyAllFilter Protected		*Default:*
errorMsgContains String Protected	A snippet of the error message or reason for that is expected in stdout. @jameleon.attribute	*Default:*
executionTimeGreaterThan int Protected	The execution time should be greater than this number if set @jameleon.attribute default="-1"	*Default:*
executionTimeLessThan int Protected	The execution time should be greater than this number if set @jameleon.attribute default="-1"	*Default:*
failOnFunctionId String Protected	A functionId where no failure should occur @jameleon.attribute	*Default:*
htmlResults StringWriter Protected	/** A snippet of the error message or reason for that is expected in stdout. @jameleon.attribute / protected String testCaseName;	*Default:*

Attribute	Notes	Constraints and tags
jmln ExecuteTestCase Protected		*Default:*

listener TestCaseListener
Protected
		Default: new TestCaseListener(){
		public void beginTestCase(TestCaseEvent event) {
		result = null;
		}
		public void endTestCase(TestCaseEvent event) {
		TestCaseTag tct = (TestCaseTag)event.getSource();
		result = tct.getResults();
		}
		}

originalCounter TestCaseCounter Protected		*Default:*
parser SAXReader Protected Static		*Default:* new SAXReader()
result TestCaseResult Protected		*Default:*
script String Protected	The script to execute @jameleon.attribute required="true"	*Default:*
stdOutWriter Writer Protected		*Default:*
timestampedResults Writer Protected		*Default:*

Operations

Method	Notes	Parameters
assertExecutionTimeGreaterThan() void Protected		long [in] greaterThanTime
assertExecutionTimeLessThan() void Protected		long [in] lessThanTime
assertFunctionIdFailed() void Protected		String [in] failOnFunctionId
assertFunctionIdNotFailed() void Protected		String [in] noFailOnFunctionId
assertHtmlTestCaseName() void Protected		String [in] expected
assertNumOfFailures() void Protected		int [in] expected
assertNumOfFunctionsRun() void Protected		int [in] expected
assertNumOfTestCasesFailed() void Protected		int [in] expected
assertNumOfTestCasesRun() void Protected		int [in] expected
assertOutcomeEquals() void Protected		String [in] expected
assertOutcomeFailed() void Protected		
assertOutcomePassed() void Protected		
assertTestCaseDocFileEndsWith() void Protected		String [in] expected
executeScript() String Protected		
getDocument() Document		StringWriter [in] results

Method	Notes	Parameters
Protected		
getExecutionTime() long		
Protected		
getFunctionsRun() List		
Protected		
getHtmlDisplayedOutcome() String		
Protected		
getHtmlDisplayedTestCaseName() String		
Protected		
getHtmlDocument() Document		
Protected		
getOutcome() String		
Protected		
getRowFromHtml() String		
Protected		
getTestCaseDocFile() String		
Protected		
getTestCasesFailed() int		
Protected		
getTestCasesRun() int		
Protected		
getValueFromElement() String		String [in] elementName String [in] message
Protected		
reverseTestCaseCount() void	Rolls back the test case execution count. We do this so that the test case inside the script being run does not increment in the total # runs in the case it is data-driven. Basically, one script equals one test case run.	
Private		
setup() void		
Public		
setupEnvironment() void		
Protected		
setUpFunctionResults() void		

Method	Notes	Parameters
Protected		
stripHtmlRow() String		
Protected		
tearDown() void		
Public		
testBlock() void		
Public		

5.3.1.2.25 RIMessageVerifyTag

Type: **Class** RIFunctionTag *Package:* tags

Verifies the designated portion of the message content.

@deprecated

@jameleon.function name="OLDri-MessageVerify" type="action"

@jameleon.step Verify the message content as specified

Connections

Connector	Source	Target	Notes
Generalization	Public	Public	
Source -> Destination	RIMessageVerifyTag	RIFunctionTag	

Attributes

Attribute	Notes	Constraints and tags
alternateElementName String Protected	The alternate name of the element to verify @jameleon.attribute	*Default:*
alternateInstanceValue String Protected	The alternative value to which the element should be set to. @jameleon.attribute	*Default:* null
elementName String Protected	The name of the element to verify @jameleon.attribute required="true"	*Default:*
instanceName String Protected	The name of the instance to verify against @jameleon.attribute required="true"	*Default:*
instanceValue String	The value to which the element should be set	*Default:* null

Attribute	Notes	Constraints and tags
Protected	to. @jameleon.attribute	
msgType String Protected	The type of the message to verify against with (REQUEST, RESPONSE, RECEIVED) @jameleon.attribute required="true"	*Default:*

Operations

Method	Notes	Parameters
testBlock() void Public		

5.3.1.2.26 RIPauseTag

Type: **Class** FunctionTag *Package:* tags

Pause the RI Script.

@jameleon.function name="ri-pause" type="navigation"

@jameleon.step Pause the test script until the user confirms

Connections

Connector	Source	Target	Notes
Generalization Source -> Destination	Public RIPauseTag	Public FunctionTag	
Realization SRS38.7 to RIPauseTag Source -> Destination	Public SRS38.7	Public RIPauseTag	
Realization SRS39.8 to RIPauseTag Source -> Destination	Public SRS39.8	Public RIPauseTag	

Operations

Method	Notes	Parameters
testBlock() void Public	Test block. Pre-Conditions: N/A Post-Conditions: N/A	

5.3.1.2.27 RISubscriptTag

Type: **Class** TestCaseTag *Package:* tags

Extends the jameleon TestCase Tag for RI usage. Every test case script must have at least one testcase tag containing all other Jameleon tags.

Some of this tags attribute may affect every nested Jameleon tag. Many of the attributes in this tag can be set globally via a jameleon.conf file.

The order of setting variables in the context follows:

- Load the CSV file variables and put them in the context.

- Load the $testEnvironment-Applications.properties and then Applications.properties and only set the variables that aren't set in the previous files. In other words, if there are variables that are going to be the same (like the page title) across multiple test cases, then first variable set wins.

- Execute the function tag and set the attributes in the context. If you want key/values in the CSV and properties files to override the script attribute, then the function point author uses the setDefaultVariableValue() method in the set method for that attribute.

- If the function point is using a map-variable, then override all settings to set the variable to the mapFrom variable name.

@jameleon.function name="subscript"

@jameleon.function name="sub-script"

@jameleon.function name="testprocedure"

@jameleon.function name="test-procedure"

Connections

Connector	Source	Target	Notes
Generalization Source -> Destination	Public RISubscriptTag	Public TestCaseTag	
Realization SRS88.2 to RISubscriptTag Source -> Destination	Public SRS88.2	Public RISubscriptTag	
Realization SRS44 to RISubscriptTag Source -> Destination	Public SRS44	Public RISubscriptTag	
Realization SRS45 to RISubscriptTag Source -> Destination	Public SRS45	Public RISubscriptTag	
Realization SRS88.1 to RISubscriptTag	Public SRS88.1	Public RISubscriptTag	

Connector	Source	Target	Notes
Source -> Destination			
Realization SRS88 to RISubscriptTag Source -> Destination	Public SRS88	Public RISubscriptTag	

Operations

Method	Notes	Parameters
doTag() void Public	This method executes the tags inside the test-case tag. If <code>useCSV=true</code>, then the data for the functional points will be grabbed from a CSV file. Otherwise the attributes are all expected to be set the contained functional points. If the functional points' attributes are set and useCSV is set to true, then the data in the CSV file will override the data set in the function points' attributes.	XMLOutput [in] out

5.3.1.2.28 RITerminateTag

Type: **Class** FunctionTag *Package:* tags

Terminate the RI Script.

@jameleon.function name="ri-terminate" type="action"

@jameleon.step Pause the test script until the user confirms

Connections

Connector	Source	Target	Notes
Generalization Source -> Destination	Public RITerminateTag	Public FunctionTag	
Realization SRS39.9 to RITerminateTag Source -> Destination	Public SRS39.9	Public RITerminateTag	
Realization SRS38.8 to RITerminateTag Source -> Destination	Public SRS38.8	Public RITerminateTag	

Attributes

Attribute	Notes	Constraints and tags
Message String Protected	The message associated with this script termination. @jameleon.attribute required="false"	*Default:* ""

Attribute	Notes	Constraints and tags

Operations

Method	Notes	Parameters
testBlock() void Public	Test block. Pre-Conditions: N/A Post-Conditions: N/A	

5.3.1.2.29 C2CRIFunctionTag

Type: **Class** FunctionTag *Package:* tags

The Class C2CRIFunctionTag.

Connections

Connector	Source	Target	Notes
Generalization	Public	Public	
Source -> Destination	C2CRIFunctionTag	FunctionTag	
Association	Public	Protected sessionTag	
Source -> Destination	C2CRIFunctionTag	C2CRISessionTag	
Generalization	Public	Public	
Source -> Destination	PublicationECReplyTag	C2CRIFunctionTag	
Generalization	Public	Public	
Source -> Destination	AutoVerifyMessageTag	C2CRIFunctionTag	
Generalization	Public	Public	
Source -> Destination	RequestResponseECTag	C2CRIFunctionTag	
Realization SRS8.1.1 to C2CRIFunctionTag Source -> Destination	Public SRS8.1.1	Public C2CRIFunctionTag	
Generalization	Public	Public	
Source -> Destination	SubscriptionOCReplyTag	C2CRIFunctionTag	
Generalization	Public	Public	
Source -> Destination	PublicationECReceiveTag	C2CRIFunctionTag	
Generalization	Public	Public	
Source -> Destination	RequestResponseOCReplyTag	C2CRIFunctionTag	
Generalization	Public	Public	

Connector	Source	Target	Notes
Source -> Destination	SubscriptionECTag	C2CRIFunctionTag	
Generalization	Public	Public	
Source -> Destination	SubscriptionOCReceiveTag	C2CRIFunctionTag	
Generalization	Public	Public	
Source -> Destination	UserVerifyMessageTag	C2CRIFunctionTag	
Generalization	Public	Public	
Source -> Destination	RequestResponseOCReceiveTag	C2CRIFunctionTag	
Generalization	Public	Public	
Source -> Destination	RIMessageVerifyTag	C2CRIFunctionTag	
Generalization	Public	Public	
Source -> Destination	PublicationOCTag	C2CRIFunctionTag	

Attributes

Attribute	Notes	Constraints and tags
sessionTag C2CRISessionTag Protected	The session tag.	*Default:*
STORE_SOURCE_CONFIG_NAME String Public Static Const	The Constant STORE_SOURCE_CONFIG_NAME.	*Default:* "RI.storeSourceOnStateChange"
storeSourceOnStateChange boolean Protected	The store source on state change.	*Default:*

Operations

Method	Notes	Parameters
setupEnvironment() void Public	Gets the references of the session and the test results from the parent TestCase This method gets called once all attributes are set from the macro language. Any required setup should go here.	

5.3.1.2.30 C2CRISessionTag

Type: **Class** SessionTag *Package:* tags

A Session tag for the C2CRI plug-in. An example of its use might:

<pre><source> <testcase xmlns="jelly:jameleon"> <C2CRI-session infoStd="tmdd-v3.0" beginSession="true"> <C2CRI-assert-element functionId="Check that the data element is the input value." element="Center Name" value = "${TMC_Name}"/> </C2CRI-session> </testcase>

</source></pre>

@jameleon.function name="C2CRI-session"

Connections

Connector	Source	Target	Notes
Sequence getInstance()	Public	Public	
Source -> Destination	C2CRISessionTag	InformationLayerStandardFactory	
Association	Public	Private theInformationLayerStandard	
Source -> Destination	C2CRISessionTag	InformationLayerStandard	
Sequence	Public	Public	
Source -> Destination	C2CRISessionTag	Jelly	
Sequence getApplicationStandard()	Public	Public	
Source -> Destination	C2CRISessionTag	ApplicationLayerStandardFactory	
Sequence	Public	Public	
Source -> Destination	C2CRISessionTag	Jelly	
Sequence stopServices()	Public	Public	
Source -> Destination	C2CRISessionTag	InformationLayerStandard	
Sequence getInformationStandard()	Public	Public	
Source -> Destination	C2CRISessionTag	InformationLayerStandardFactory	
Generalization	Public	Public	
Source -> Destination	C2CRISessionTag	SessionTag	
Sequence initializeStandard(testCaseName, centerMode, appStandard)	Public	Public	
Source -> Destination	C2CRISessionTag	InformationLayerStandard	
Association	Public	Private theApplicationLayerStandard	
Source -> Destination	C2CRISessionTag	ApplicationLayerStandard	
Sequence	Public	Public	

Connector	Source	Target	Notes
Source -> Destination	C2CRISessionTag	RequestResponseOCReceiveTag	
Sequence	Public	Public	
Source -> Destination	C2CRISessionTag	RequestResponseOCReplyTag	
Sequence getInstance()	Public	Public	
Source -> Destination	C2CRISessionTag	ApplicationLayerStandardFactory	
Sequence getTheInformationLayerStandard()	Public	Public	
Source -> Destination	RequestResponseOCReplyTag	C2CRISessionTag	
Sequence getTheInformationLayerStandard()	Public	Public	
Source -> Destination	RequestResponseOCReceiveTag	C2CRISessionTag	
Sequence	Public	Public	
Source -> Destination	ApplicationLayerStandardFactory	C2CRISessionTag	
Sequence	Public	Public	
Source -> Destination	InformationLayerStandard	C2CRISessionTag	
Sequence tearDownSession()	Public	Public	
Source -> Destination	Jelly	C2CRISessionTag	
Sequence SessionTag()	Public	Public	
Source -> Destination	Jelly	C2CRISessionTag	
Sequence	Public	Public	
Source -> Destination	InformationLayerStandardFactory	C2CRISessionTag	
Realization SRS8.1.1 to C2CRISessionTag	Public	Public	
Source -> Destination	SRS8.1.1	C2CRISessionTag	
Sequence	Public	Public	
Source -> Destination	InformationLayerStandardFactory	C2CRISessionTag	
Association	Public	Protected sessionTag	
Source -> Destination	C2CRIFunctionTag	C2CRISessionTag	
Sequence	Public	Public	
Source -> Destination	InformationLayerStanda	C2CRISessionTag	

Connector	Source	Target	Notes
	rd		
Sequence setUpSession()	Public	Public	
Source -> Destination	Jelly	C2CRISessionTag	

Attributes

Attribute	Notes	Constraints and tags
appStd String Protected	@jameleon.attribute contextName="appStd"	*Default:*
appStdSuiteURL String Protected	@jameleon.attribute contextName="APPSTD_SUITEURL"	*Default:*
encodingType String Protected	The encoding type that will be used for this service. Only applied when WSDL not used. @jameleon.attribute	*Default:*
infoStd String Protected	@jameleon.attribute contextName="infoStd"	*Default:*
infoStdSuiteURL String Protected	@jameleon.attribute contextName="INFOSTD_SUITEURL"	*Default:*
isPublicationOperation boolean Protected	Flag indicating whether the operation specified is a publication. @jameleon.attribute	*Default:* false
lclAddress String Protected	@jameleon.attribute contextName="lclAddress"	*Default:*
lclPort String Protected	@jameleon.attribute contextName="lclPort"	*Default:*
messageList ArrayList Protected	A handle for the C2C messages	*Default:*
operationName String Protected	The operation name of the portType name of the service to be tested. Must match the	*Default:*

Attribute	Notes	Constraints and tags
	WSDL definition @jameleon.attribute	
password String Protected	A password to be applied. @jameleon.attribute	*Default:* ""
portName String Protected	The port name of the service to be tested. Must match the WSDL definition @jameleon.attribute	*Default:*
publicationDialog String Protected	If provided, the name of the publication dialog that will implemented by this session. @jameleon.attribute contextName="publicationDialog"	*Default:*
requestDialog String Protected	If provided, the name of the request dialog that will implemented by this session. @jameleon.attribute contextName="requestDialog"	*Default:*
riMode String Protected	@jameleon.attribute contextName="RIMode"	*Default:*
serviceName String Protected	The service name to be tested. Must match the WSDL definition @jameleon.attribute	*Default:*
subscriptionDialog String Protected	If provided, the name of the subscription dialog that will implemented by this session. @jameleon.attribute contextName="subscriptionDialog"	*Default:*
sutAddress String Protected	@jameleon.attribute contextName="SUTAddress"	*Default:*
sutPort String Protected	@jameleon.attribute contextName="SUTPort"	*Default:*

Attribute	Notes	Constraints and tags
testCaseIdentifier String Protected	A unique identifier for this test case. @jameleon.attribute required="true"	*Default:*
theApplicationLayerStandard ApplicationLayerStandard Private	the Service object managed by this Session	*Default:*
theInformationLayerStandard InformationLayerStandard Private		*Default:*
transportType String Protected	The transport type that will be used for this service. Only applied when WSDL not used. @jameleon.attribute	*Default:*
userName String Protected	A user name to be applied. @jameleon.attribute	*Default:* ""
useWSDL boolean Protected	Flag indicating whether the service will be defined from a WSDL file. @jameleon.attribute	*Default:* true
wsdlURL String Protected	@jameleon.attribute contextName="RI_WEBSERVICEURL"	*Default:*

Operations

Method	Notes	Parameters
getAppStd() String Public	Gets the application Layer Standard @return the application layer standard selected.	
getInfoStd() String Public	Gets the information Layer Standard. @return the information layer standard selected.	

Method	Notes	Parameters
getlclAddress() String Public	Gets the address to be used for the RI @return the address to be used for the RI.	
getlclPort() int Public	Gets the port to be used for the RI @return the port to be used for the RI.	
getOperationName() String Public		
getRequestAppStd() String Protected	Gets the application layer standard to request @return The application layer standard to use in startApplication.	
getRequestInfoStd() String Protected	Gets the information layer standard to request @return The info standard to use in startApplication.	
getSUTAddress() String Public	Gets the address to be used for the SUT @return the address to be used for the SUT.	
getSUTPort() int Public	Gets the port to be used for the SUT @return the port to be used for the SUT.	
getTheApplicationLayerStandard() ApplicationLayerStandard Public		
getTheInformationLayerStandard() InformationLayerStandard Public		
setUpSession() void	Sets the up session. Pre-Conditions: N/A Post-Conditions: N/A	

Method	Notes	Parameters
Public		
startApplication() void	Ensures that things are started off correctly.	
Public		
tearDownSession() void		
Public		

5.3.1.2.31 PublicationECReceiveTag

Type: **Class** C2CRIFunctionTag *Package:* tags

Send the message request

@jameleon.function name="PUBLICATION-EC-RECEIVE" type="action"

@jameleon.step Receive the publication message

Connections

Connector	Source	Target	Notes
Association Source -> Destination	Public PublicationECReceiveTag	Private theInformationLayerController InformationLayerController	
Generalization Source -> Destination	Public PublicationECReceiveTag	Public C2CRIFunctionTag	
Realization SRS92 to PublicationECReceiveTag Source -> Destination	Public SRS92	Public PublicationECReceiveTag	
Realization SRS8.1.1 to PublicationECReceiveTag Source -> Destination	Public SRS8.1.1	Public PublicationECReceiveTag	

Attributes

Attribute	Notes	Constraints and tags
ALLOWACCESS String Protected	The flag for whether to allow the requesting center access @jameleon.attribute required="false" default = "True"	*Default:*
AUTHENTICATIONCHECK String Protected	The flag for whether to perform authentication on the received message @jameleon.attribute required="false" default	*Default:*

Attribute	Notes	Constraints and tags
	= "False"	
DATAVALID String Protected	The flag for whether there is valid data available @jameleon.attribute required="false" default="True"	*Default:*
Dialog String Protected	The name of the dialog associated with this request @jameleon.attribute required="true"	*Default:*
PUBLICATIONERRORRESPONSEEXPECTED String Protected	A flag which indicates whether a error response is expected @jameleon.attribute required="false"	*Default:* "False"
PUBLICATIONERRORTYPEEXPECTED String Protected	The type error response that is expected @jameleon.attribute required="false"	*Default:*
PUBLICATIONSTORECEIVE String Protected	The number of publications expected @jameleon.attribute required="false"	*Default:*
RESPONSETIMEREQUIRED String Protected	The time in milliseconds within which an owner center has to send a response @jameleon.attribute required="true"	*Default:*
theInformationLayerController InformationLayerController Private	The information layer controller.	*Default:*

Operations

Method	Notes	Parameters
testBlock() void Public	Test block. Pre-Conditions: N/A Post-Conditions: N/A	

5.3.1.2.32 PublicationECReplyTag

Type: **Class** C2CRIFunctionTag *Package:* tags

Send the reply message

@jameleon.function name="PUBLICATION-EC-REPLY" type="action"

@jameleon.step Send the message response

Connections

Connector	Source	Target	Notes
Generalization Source -> Destination	Public PublicationECReplyTag	Public C2CRIFunctionTag	
Association Source -> Destination	Public PublicationECReplyTag	Private theInformationLayerController InformationLayerController	
Realization SRS8.1.1 to PublicationECReplyTag Source -> Destination	Public SRS8.1.1	Public PublicationECReplyTag	
Realization SRS92 to PublicationECReplyTag Source -> Destination	Public SRS92	Public PublicationECReplyTag	

Attributes

Attribute	Notes	Constraints and tags
CONTENTVERIFIED String Protected	A flag which indicates whether the request optional content was successfully verified @jameleon.attribute required="true"	*Default:* "NotSet"
Dialog String Protected	The name of the dialog associated with this subscription @jameleon.attribute required="true"	*Default:*
PUBLICATIONRESPONSEMESSAGE Object Protected	The message response to send @jameleon.attribute required="true"	*Default:*
RESPONSETIMEREQUIRED String Protected	The time in milliseconds within which an owner center has to send a response @jameleon.attribute required="true"	*Default:*
theInformationLayerController InformationLayerController	The information layer controller.	*Default:*

Attribute	Notes	Constraints and tags
Private		

Operations

Method	Notes	Parameters
testBlock() void Public	Test block. Pre-Conditions: N/A Post-Conditions: N/A	

5.3.1.2.33 PublicationOCTag

Type: **Class** C2CRIFunctionTag *Package:* tags

Send the message request

@jameleon.function name="PUBLICATION-OC" type="action"

@jameleon.step Send the message request

Connections

Connector	Source	Target	Notes
Association Source -> Destination	Public PublicationOCTag	Private theInformationLayerController InformationLayerController	
Generalization Source -> Destination	Public PublicationOCTag	Public C2CRIFunctionTag	
Realization Source -> Destination	Public SRS107	Public PublicationOCTag	
Realization SRS92 to PublicationOCTag Source -> Destination	Public SRS92	Public PublicationOCTag	
Realization Source -> Destination	Public SRS133	Public PublicationOCTag	
Realization Source -> Destination	Public SRS130	Public PublicationOCTag	
Realization Source -> Destination	Public SRS118	Public PublicationOCTag	
Realization SRS8.1.1 to PublicationOCTag Source -> Destination	Public SRS8.1.1	Public PublicationOCTag	

Connector	Source	Target	Notes
Realization	Public	Public	
Source -> Destination	SRS142	PublicationOCTag	
Realization	Public	Public	
Source -> Destination	SRS127	PublicationOCTag	
Realization	Public	Public	
Source -> Destination	SRS136	PublicationOCTag	
Realization	Public	Public	
Source -> Destination	SRS139	PublicationOCTag	
Realization	Public	Public	
Source -> Destination	SRS121	PublicationOCTag	
Realization	Public	Public	
Source -> Destination	SRS114	PublicationOCTag	
Realization	Public	Public	
Source -> Destination	SRS124	PublicationOCTag	

Attributes

Attribute	Notes	Constraints and tags
Dialog String Protected	The name of the dialog associated with this request @jameleon.attribute required="true"	*Default:*
PUBLICATIONCOUNT Integer Protected	The number of publications that have been sent including this one @jameleon.attribute required="true"	*Default:*
PUBLICATIONERRORRESPONSEEXPECTED String Protected	A flag which indicates whether a error response is expected @jameleon.attribute required="true"	*Default:*
PUBLICATIONERRORTYPEEXPECTED String Protected	The type error response that is expected @jameleon.attribute required="true"	*Default:*
PUBLICATIONMESSAGE Object Protected	The message request to send @jameleon.attribute required="true"	*Default:*

Attribute	Notes	Constraints and tags
RESPONSETIMEREQUIRED String Protected	The time in milliseconds within which an owner center has to send a response @jameleon.attribute required="true"	*Default:*
theInformationLayerController InformationLayerController Private	The information layer controller.	*Default:*

Operations

Method	Notes	Parameters
testBlock() void Public	Test block. Pre-Conditions: N/A Post-Conditions: N/A	

5.3.1.2.34 RIAddEntityElementTag

Type: **Class** *Package:* tags

This script tag adds a new element to the given entity. It uses the instance of the RIEmulation class to access the entities.

@jameleon.function name="ri-AddEntityElement" type="action"

@jameleon.step Add the entity data element as specified

Connections

Connector	Source	Target	Notes
Realization Source -> Destination	Public SRS41.6	Public RIAddEntityElementTag	

Attributes

Attribute	Notes	Constraints and tags
elementName String Protected	The name of the element to add @jameleon.attribute required="true"	*Default:*
elementValue String Protected	The value to which the element should be set to. @jameleon.attribute	*Default:*
entityName String	The name of the entity to add the element to	*Default:*

Attribute	Notes	Constraints and tags
Protected	@jameleon.attribute required="true"	

Operations

Method	Notes	Parameters
testBlock() void Public	Test block. Pre-Conditions: N/A Post-Conditions: N/A	

5.3.1.2.35 RIAddEntityTag

Type: **Class** *Package:* tags

This script tag adds a new instance of an entity type to the current entity data. It uses the instance of the RIEmulation class to access the entities.

@jameleon.function name="ri-AddEntity" type="action"

@jameleon.step Add the entity as specified

Connections

Connector	Source	Target	Notes
Realization	Public	Public	
Source -> Destination	SRS41.6	RIAddEntityTag	

Attributes

Attribute	Notes	Constraints and tags
entityId String Protected	The identifier to use for the added entity @jameleon.attribute required="true"	*Default:*
entityType String Protected	The name of the type of entity to add @jameleon.attribute	*Default:*

Operations

Method	Notes	Parameters
testBlock() void Public	Test block. Pre-Conditions: N/A Post-Conditions: N/A	

5.3.1.2.36 RIDeleteEntityElementTag

Type: **Class** *Package:* tags

This script tag deletes the specified element from the entity data. It uses the instance of the RIEmulation class to access the entities.

@jameleon.function name="ri-DeleteEntityElement" type="action"

@jameleon.step remove the specified entity element as specified

Connections

Connector	Source	Target	Notes
Realization	Public	Public	
Source -> Destination	SRS41.7	RIDeleteEntityElementTag	

Attributes

Attribute	Notes	Constraints and tags
entityelement String Private	The identifier of the entity from which the element will be deleted.	*Default:*
entityId String Private	The identifier for the entity from which the element will be deleted.	*Default:*
entityType String Protected	The type of entity to delete @jameleon.attribute	*Default:*
entityValue String Protected	The identifier of the entity to be deleted. @jameleon.attribute	*Default:*

Operations

Method	Notes	Parameters
testBlock() void Public	Test block. Pre-Conditions: N/A Post-Conditions: N/A	

5.3.1.2.37 RIDeleteEntityTag

Type: **Class** *Package:* tags

This script tag deletes the specified entity. It uses the instance of the RIEmulation class to access the entities.

@jameleon.function name="ri-DeleteEntity" type="action"

@jameleon.step Delete the entity data as specified

Connections

Connector	Source	Target	Notes
Realization	Public	Public	
Source -> Destination	SRS41.7	RIDeleteEntityTag	

Attributes

Attribute	Notes	Constraints and tags
entityId String Protected	The identifier of the entity that will be deleted. @jameleon.attribute	*Default:*
entityType String Protected	The type of entity that will be deleted @jameleon.attribute	*Default:*

Operations

Method	Notes	Parameters
testBlock() void Public	Test block. Pre-Conditions: N/A Post-Conditions: N/A	

5.3.1.2.38 RIGetEntityElementValueTag

Type: **Class** *Package:* tags

This script tag returns the value associated with the entity element specified. It uses the instance of the RIEmulation class to access the entities.

@jameleon.function name="ri-GetEntityElementValue" type="action"

@jameleon.step Initialize the entity data as specified

Connections

Connector	Source	Target	Notes
Realization	Public	Public	
Source -> Destination	SRS41.5	RIGetEntityElementValueTag	
Realization	Public	Public	
Source -> Destination	SRS41.4	RIGetEntityElementValueTag	

Attributes

Attribute	Notes	Constraints and tags
elementName String	The name of the element to retrieve	*Default:*

Attribute	Notes	Constraints and tags
Protected	@jameleon.attribute required="true"	
elementValue string Private	The returned value associated with the entity specified.	*Default:*
entityId String Protected	The identifier of the entity. @jameleon.attribute	*Default:*
entityType String Protected	The type of entity to retrieve @jameleon.attribute	*Default:*

Operations

Method	Notes	Parameters
testBlock() void Public	Test block. Pre-Conditions: N/A Post-Conditions: N/A	

5.3.1.2.39 RIInitializeEntityTag

Type: **Class** *Package:* tags

This script tag Initializes the entity information. It uses the instance of the RIEmulation class to perform this function.

@jameleon.function name="ri-InitializeEntity" type="action"

@jameleon.step Initialize the entity data as specified

Connections

Connector	Source	Target	Notes
Realization	Public	Public	
Source -> Destination	SRS41.1	RIInitializeEntityTag	

Attributes

Attribute	Notes	Constraints and tags
entityType String Protected	The type of entity that should be initialized to it's default data. @jameleon.attribute	*Default:*

Operations

Method	Notes	Parameters
testBlock() void Public	Test block. Pre-Conditions: N/A Post-Conditions: N/A	

5.3.1.2.40 RIMessageVerifyTag

Type: **Class** C2CRIFunctionTag *Package:* tags

Creates a new message of the specified type (Request, Response, Received)

@jameleon.function name="ri-MessageVerify" type="action"

@jameleon.step Verify the message content as specified

Connections

Connector	Source	Target	Notes
Generalization Source -> Destination	Public RIMessageVerifyTag	Public C2CRIFunctionTag	
Realization SRS8.1.1 to RIMessageVerifyTag Source -> Destination	Public SRS8.1.1	Public RIMessageVerifyTag	
Realization SRS92 to RIMessageVerifyTag Source -> Destination	Public SRS92	Public RIMessageVerifyTag	

Attributes

Attribute	Notes	Constraints and tags
alternateElementName String Protected	The alternate name of the element to verify @jameleon.attribute	*Default:*
alternateInstanceValue String Protected	The alternative value to which the element should be set to. @jameleon.attribute	*Default:* null
elementName String Protected	The name of the element to verify @jameleon.attribute required="true"	*Default:*
elementType String Protected	The value which indicates whether the element being verified is a message, data-frame or data-element.	*Default:* null

Attribute	Notes	Constraints and tags
	@jameleon.attribute	
instanceName String Protected	The name of the instance to verify against @jameleon.attribute required="true"	*Default:*
instanceValue String Protected	The value to which the element should be set to. @jameleon.attribute	*Default:*
msgType String Protected	The type of the message to verify against with (REQUEST, RESPONSE, RECEIVED) @jameleon.attribute required="true"	*Default:*

Operations

Method	Notes	Parameters
testBlock() void Public	Test block. Pre-Conditions: N/A Post-Conditions: N/A	

5.3.1.2.41 RIUpdateEntityElementTag

Type: **Class** *Package:* tags

This script tag sets the value associated with the entity element specified. It uses the instance of the RIEmulation class to access the entity.

@jameleon.function name="ri-UpdateEntityElement" type="action"

@jameleon.step Update the entity data value as specified

Connections

Connector	Source	Target	Notes
Realization	Public	Public	
Source -> Destination	SRS41.2	RIUpdateEntityElement Tag	
Realization	Public	Public	
Source -> Destination	SRS41.3	RIUpdateEntityElement Tag	

Attributes

Attribute	Notes	Constraints and tags
elementName String Protected	The name of the element to update @jameleon.attribute required="true"	*Default:*
elementValue String Protected	The value to which the element should be set to. @jameleon.attribute	*Default:*
entityid String Protected	The identifier for the entity to be updated. @jameleon.attribute	*Default:*
entityType String Protected	The type of the entity information to update. @jameleon.attribute	*Default:*

Operations

Method	Notes	Parameters
testBlock() void Public	Test block. Pre-Conditions: N/A Post-Conditions: N/A	

5.3.1.2.42 RequestResponseECTag

Type: **Class** C2CRIFunctionTag *Package:* tags

Send the message request

@jameleon.function name="REQUEST-RESPONSE-EC" type="action"

@jameleon.step Send the message request

Connections

Connector	Source	Target	Notes
Generalization Source -> Destination	Public RequestResponseECTag	Public C2CRIFunctionTag	
Association Source -> Destination	Public RequestResponseECTag	Private theInformationLayerController InformationLayerController	
Realization SRS8.1.1 to RequestResponseECTag Source -> Destination	Public SRS8.1.1	Public RequestResponseECTag	

Connector	Source	Target	Notes
Realization Source -> Destination	Public SRS134	Public RequestResponseECTag	
Realization SRS92 to RequestResponseECTag Source -> Destination	Public SRS92	Public RequestResponseECTag	

Attributes

Attribute	Notes	Constraints and tags
Dialog String Protected	The name of the dialog associated with this request @jameleon.attribute required="true"	*Default:*
ERRORRESPONSEEXPECTED String Protected	A flag which indicates whether a error response is expected @jameleon.attribute required="true"	*Default:*
ERRORTYPEEXPECTED String Protected	The type error response that is expected @jameleon.attribute required="true"	*Default:*
REQUESTMESSAGE Object Protected	The message request to send @jameleon.attribute required="true"	*Default:*
RESPONSETIMEREQUIRED String Protected	The time in milliseconds within which an owner center has to send a response @jameleon.attribute required="true"	*Default:*
theInformationLayerController InformationLayerController Private	The information layer controller.	*Default:*

Operations

Method	Notes	Parameters
testBlock() void Public	Test block. Pre-Conditions: N/A Post-Conditions: N/A	

5.3.1.2.43 RequestResponseOCReceiveTag

Type: **Class** C2CRIFunctionTag *Package:* tags

Send the message request

@jameleon.function name="REQUEST-RESPONSE-OC-RECEIVE" type="action"

@jameleon.step Receive the message request

Connections

Connector	Source	Target	Notes
Sequence Source -> Destination	Public RequestResponseOCReceiveTag	Public Jelly	
Association Source -> Destination	Public RequestResponseOCReceiveTag	Private theInformationLayerController InformationLayerController	
Sequence getTheInformationLayerStandard() Source -> Destination	Public RequestResponseOCReceiveTag	Public C2CRISessionTag	
Sequence getInformationLayerController() Source -> Destination	Public RequestResponseOCReceiveTag	Public InformationLayerStandard	
Sequence performRequestResponseOCReceive(dialog) Source -> Destination	Public RequestResponseOCReceiveTag	Public InformationLayerController	
Generalization Source -> Destination	Public RequestResponseOCReceiveTag	Public C2CRIFunctionTag	
Sequence performRequestResponseOCReceive(dialog) Source -> Destination	Public RequestResponseOCReceiveTag	Public InformationLayerController	
Realization SRS92 to RequestResponseOCReceiveTag Source -> Destination	Public SRS92	Public RequestResponseOCReceiveTag	
Realization Source -> Destination	Public SRS131	Public RequestResponseOCReceiveTag	

Connector	Source	Target	Notes
Sequence doTag(out)	Public	Public	
Source -> Destination	Jelly	RequestResponseOCReceiveTag	
Realization SRS8.1.1 to RequestResponseOCReceiveTag	Public	Public	
Source -> Destination	SRS8.1.1	RequestResponseOCReceiveTag	
Realization	Public	Public	
Source -> Destination	SRS109	RequestResponseOCReceiveTag	
Realization	Public	Public	
Source -> Destination	SRS115	RequestResponseOCReceiveTag	
Realization	Public	Public	
Source -> Destination	SRS122	RequestResponseOCReceiveTag	
Realization	Public	Public	
Source -> Destination	SRS99	RequestResponseOCReceiveTag	
Realization	Public	Public	
Source -> Destination	SRS128	RequestResponseOCReceiveTag	
Realization	Public	Public	
Source -> Destination	SRS134	RequestResponseOCReceiveTag	
Realization	Public	Public	
Source -> Destination	SRS108	RequestResponseOCReceiveTag	
Sequence	Public	Public	
Source -> Destination	InformationLayerController	RequestResponseOCReceiveTag	
Realization	Public	Public	
Source -> Destination	SRS137	RequestResponseOCReceiveTag	
Realization	Public	Public	
Source -> Destination	SRS119	RequestResponseOCReceiveTag	
Realization	Public	Public	
Source -> Destination	SRS125	RequestResponseOCReceiveTag	

Connector	Source	Target	Notes
Realization	Public	Public	
Source -> Destination	SRS117	RequestResponseOCReceiveTag	
Sequence	Public	Public	
Source -> Destination	C2CRISessionTag	RequestResponseOCReceiveTag	
Sequence	Public	Public	
Source -> Destination	InformationLayerStandard	RequestResponseOCReceiveTag	
Sequence	Public	Public	
Source -> Destination	InformationLayerController	RequestResponseOCReceiveTag	
Realization	Public	Public	
Source -> Destination	SRS140	RequestResponseOCReceiveTag	

Attributes

Attribute	Notes	Constraints and tags
ALLOWACCESS String Protected	A flag indicating whether the external center should be allowed access to the requested information. @jameleon.attribute required="true"	*Default:*
AUTHENTICATIONCHECK String Protected	A flag which indicates whether authentication checking is expected/enabled @jameleon.attribute required="true"	*Default:*
DATAVALID String Protected	The flag for whether there is valid data available @jameleon.attribute required="false"	*Default:* "True"
Dialog String Protected	The name of the dialog associated with this request @jameleon.attribute required="true"	*Default:*
ERRORRESPONSEEXPECTED String Protected	A flag which indicates whether a error response is expected @jameleon.attribute required="true"	*Default:*

Attribute	Notes	Constraints and tags
ERRORTYPEEXPECTED String Protected	The type error response that is expected @jameleon.attribute required="true"	*Default:*
OPERATORID String Protected	The type organization ID that was provided for the test case @jameleon.attribute required="true"	*Default:*
PASSWORD String Protected	The password that was provided for the test case @jameleon.attribute required="true"	*Default:*
RESPONSETIMEREQUIRED String Protected	The time in milliseconds within which an owner center has to send a response @jameleon.attribute required="true"	*Default:*
theInformationLayerController InformationLayerController Private	The information layer controller.	*Default:*
USERNAME String Protected	The username that was provided for the test case @jameleon.attribute required="true"	*Default:*

Operations

Method	Notes	Parameters
testBlock() void Public	Test block. Pre-Conditions: N/A Post-Conditions: N/A	

5.3.1.2.44 RequestResponseOCReplyTag

Type: **Class** C2CRIFunctionTag *Package:* tags

Send the reply message

@jameleon.function name="REQUEST-RESPONSE-OC-REPLY" type="action"

@jameleon.step Send the message response

Connections

Connector	Source	Target	Notes

Connector	Source	Target	Notes
Sequence getTheInformationLayerStandard() Source -> Destination	Public RequestResponseOCReplyTag	Public C2CRISessionTag	
Sequence getRRResponse(dialog, requestMessage) Source -> Destination	Public RequestResponseOCReplyTag	Public RIEmulation	
Association Source -> Destination	Public RequestResponseOCReplyTag	Private theApplicationLayerStandard InformationLayerController	
Sequence getInformationLayerController() Source -> Destination	Public RequestResponseOCReplyTag	Public InformationLayerStandard	
Generalization Source -> Destination	Public RequestResponseOCReplyTag	Public C2CRIFunctionTag	
Sequence performRequestResponseOCResponse(dialog, responseMessage, isErrorResponse) Source -> Destination	Public RequestResponseOCReplyTag	Public InformationLayerController	
Sequence performRequestResponseOCResponse(dialog, responseMessage, isErrorResponse) Source -> Destination	Public RequestResponseOCReplyTag	Public InformationLayerController	
Realization Source -> Destination	Public SRS125	Public RequestResponseOCReplyTag	
Sequence Source -> Destination	Public InformationLayerController	Public RequestResponseOCReplyTag	
Realization Source -> Destination	Public SRS108	Public RequestResponseOCReplyTag	
Realization SRS92 to RequestResponseOCReply	Public SRS92	Public RequestResponseOCRe	

Connector	Source	Target	Notes
Tag		plyTag	
Source -> Destination			
Realization SRS8.1.1 to RequestResponseOCReply Tag	Public SRS8.1.1	Public RequestResponseOCReplyTag	
Source -> Destination			
Realization	Public	Public	
Source -> Destination	SRS119	RequestResponseOCReplyTag	
Sequence	Public	Public	
Source -> Destination	RIEmulation	RequestResponseOCReplyTag	
Realization	Public	Public	
Source -> Destination	SRS100	RequestResponseOCReplyTag	
Realization	Public	Public	
Source -> Destination	SRS131	RequestResponseOCReplyTag	
Sequence	Public	Public	
Source -> Destination	InformationLayerStandard	RequestResponseOCReplyTag	
Realization	Public	Public	
Source -> Destination	SRS99	RequestResponseOCReplyTag	
Realization	Public	Public	
Source -> Destination	SRS117	RequestResponseOCReplyTag	
Realization	Public	Public	
Source -> Destination	SRS122	RequestResponseOCReplyTag	
Realization	Public	Public	
Source -> Destination	SRS109	RequestResponseOCReplyTag	
Realization	Public	Public	
Source -> Destination	SRS140	RequestResponseOCReplyTag	
Realization	Public	Public	
Source -> Destination	SRS137	RequestResponseOCReplyTag	
Realization	Public	Public	

Connector	Source	Target	Notes
Source -> Destination	SRS115	RequestResponseOCReplyTag	
Sequence	Public	Public	
Source -> Destination	InformationLayerController	RequestResponseOCReplyTag	
Sequence	Public	Public	
Source -> Destination	C2CRISessionTag	RequestResponseOCReplyTag	
Realization	Public	Public	
Source -> Destination	SRS128	RequestResponseOCReplyTag	

Attributes

Attribute	Notes	Constraints and tags
CONTENTVERIFIED String Protected	A flag which indicates whether the request optional content was successfully verified @jameleon.attribute required="true"	*Default:* "NotSet"
Dialog String Protected	The name of the dialog associated with this request @jameleon.attribute required="true"	*Default:*
RESPONSEMESSAGE Object Protected	The message response to send @jameleon.attribute required="true"	*Default:*
RESPONSETIMEREQUIRED String Protected	The time in milliseconds within which an owner center has to send a response @jameleon.attribute required="true"	*Default:*
theApplicationLayerStandard InformationLayerController Private	The application layer standard.	*Default:*

Operations

Method	Notes	Parameters
testBlock() void Public	Test block. Pre-Conditions: N/A Post-Conditions: N/A	

5.3.1.2.45 SubscriptionECTag

Type: **Class** C2CRIFunctionTag *Package:* tags

Send the message request

@jameleon.function name="SUBSCRIPTION-EC" type="action"

@jameleon.step Send the message request

Connections

Connector	Source	Target	Notes
Generalization Source -> Destination	Public SubscriptionECTag	Public C2CRIFunctionTag	
Association Source -> Destination	Public SubscriptionECTag	Private informationLayerController InformationLayerController	
Realization SRS92 to SubscriptionECTag Source -> Destination	Public SRS92	Public SubscriptionECTag	
Realization SRS8.1.1 to SubscriptionECTag Source -> Destination	Public SRS8.1.1	Public SubscriptionECTag	

Attributes

Attribute	Notes	Constraints and tags
Dialog String Protected	The name of the dialog associated with this request @jameleon.attribute required="true"	*Default:*
informationLayerController InformationLayerController Private	The information layer controller.	*Default:*
RESPONSETIMEREQUIRED String Protected	The time in milliseconds within which an owner center has to send a response @jameleon.attribute required="false" default="60000"	*Default:*
SUBSCRIPTIONERRORRESPONSEEXPECTED String Protected	A flag which indicates whether a error response is expected @jameleon.attribute required="true"	*Default:*

Attribute	Notes	Constraints and tags
	name="SUBSCRIPTIONERRORRESPONSEEXPECTED"	
SUBSCRIPTIONERRORTYPEEXPECTED String Protected	The type error response that is expected @jameleon.attribute required="true" name="SUBSCRIPTIONERRORTYPEEXPECTED"	*Default:*
SUBSCRIPTIONMESSAGE Object Protected	The message request to send @jameleon.attribute required="true" name = "SUBSCRIPTIONMESSAGE"	*Default:*

Operations

Method	Notes	Parameters
testBlock() void Public	Test block. Pre-Conditions: N/A Post-Conditions: N/A	

5.3.1.2.46 SubscriptionOCReceiveTag

Type: **Class** C2CRIFunctionTag *Package:* tags

Send the message request

@jameleon.function name="SUBSCRIPTION-OC-RECEIVE" type="action"

@jameleon.step Receive the message request

Connections

Connector	Source	Target	Notes
Generalization Source -> Destination	Public SubscriptionOCReceiveTag	Public C2CRIFunctionTag	
Association Source -> Destination	Public SubscriptionOCReceiveTag	Private theInformationLayerController InformationLayerController	
Realization SRS92 to SubscriptionOCReceiveTag Source -> Destination	Public SRS92	Public SubscriptionOCReceiveTag	
Realization	Public	Public	

Connector	Source	Target	Notes
Source -> Destination	SRS113	SubscriptionOCReceiveTag	
Realization	Public	Public	
Source -> Destination	SRS116	SubscriptionOCReceiveTag	
Realization	Public	Public	
Source -> Destination	SRS123	SubscriptionOCReceiveTag	
Realization	Public	Public	
Source -> Destination	SRS138	SubscriptionOCReceiveTag	
Realization	Public	Public	
Source -> Destination	SRS110	SubscriptionOCReceiveTag	
Realization	Public	Public	
Source -> Destination	SRS141	SubscriptionOCReceiveTag	
Realization SRS8.1.1 to SubscriptionOCReceiveTag	Public	Public	
Source -> Destination	SRS8.1.1	SubscriptionOCReceiveTag	
Realization	Public	Public	
Source -> Destination	SRS120	SubscriptionOCReceiveTag	
Realization	Public	Public	
Source -> Destination	SRS132	SubscriptionOCReceiveTag	
Realization	Public	Public	
Source -> Destination	SRS117.5	SubscriptionOCReceiveTag	
Realization	Public	Public	
Source -> Destination	SRS126	SubscriptionOCReceiveTag	
Realization	Public	Public	
Source -> Destination	SRS129	SubscriptionOCReceiveTag	
Realization	Public	Public	
Source -> Destination	SRS135	SubscriptionOCReceiveTag	

Attributes

Attribute	Notes	Constraints and tags
ALLOWACCESS String Protected	The type error response that is expected @jameleon.attribute required="true"	*Default:*
AUTHENTICATIONCHECK String Protected	A flag which indicates whether a error response is expected @jameleon.attribute required="true"	*Default:*
DATAVALID String Protected	The flag for whether there is valid data available @jameleon.attribute required="false" default="True"	*Default:*
Dialog String Protected	The name of the dialog associated with this request @jameleon.attribute required="true"	*Default:*
OPERATORID String Protected	The type error response that is expected @jameleon.attribute required="true"	*Default:*
PASSWORD String Protected	The type error response that is expected @jameleon.attribute required="true"	*Default:*
PERIODICUPDATERANGE String Protected	The minimum time and maximum time (separated by a ":") in seconds for which an owner center supports periodic publications. @jameleon.attribute required="false" default="1:31536000";	*Default:*
RESPONSETIMEREQUIRED String Protected	The time in milliseconds within which an owner center has to send a response @jameleon.attribute required="false" default="60000";	*Default:*
SUBSCRIPTIONERRORRESPONSEEXPECTED String Protected	A flag which indicates whether a error response is expected @jameleon.attribute required="true"	*Default:*
SUBSCRIPTIONERRORTYPEEXPECTED String	The type error response that is expected	*Default:*

Attribute	Notes	Constraints and tags
Protected	@jameleon.attribute required="true"	
theInformationLayerController InformationLayerController Private		*Default:*
USERNAME String Protected	The type error response that is expected @jameleon.attribute required="true"	*Default:*

Operations

Method	Notes	Parameters
testBlock() void Public		

5.3.1.2.47 SubscriptionOCReplyTag

Type: **Class** C2CRIFunctionTag *Package:* tags

Send the reply message

@jameleon.function name="SUBSCRIPTION-OC-REPLY" type="action"

@jameleon.step Send the message response

Connections

Connector	Source	Target	Notes
Generalization Source -> Destination	Public SubscriptionOCReplyTag	Public C2CRIFunctionTag	
Association Source -> Destination	Public SubscriptionOCReplyTag	Private theInformationLayerController InformationLayerController	
Realization Source -> Destination	Public SRS120	Public SubscriptionOCReplyTag	
Realization SRS92 to SubscriptionOCReplyTag Source -> Destination	Public SRS92	Public SubscriptionOCReplyTag	
Realization Source -> Destination	Public SRS129	Public SubscriptionOCReplyTa	

Connector	Source	Target	Notes
		g	
Realization	Public	Public	
Source -> Destination	SRS117.5	SubscriptionOCReplyTa g	
Realization	Public	Public	
Source -> Destination	SRS135	SubscriptionOCReplyTa g	
Realization	Public	Public	
Source -> Destination	SRS126	SubscriptionOCReplyTa g	
Realization	Public	Public	
Source -> Destination	SRS110	SubscriptionOCReplyTa g	
Realization	Public	Public	
Source -> Destination	SRS132	SubscriptionOCReplyTa g	
Realization	Public	Public	
Source -> Destination	SRS138	SubscriptionOCReplyTa g	
Realization	Public	Public	
Source -> Destination	SRS116	SubscriptionOCReplyTa g	
Realization	Public	Public	
Source -> Destination	SRS141	SubscriptionOCReplyTa g	
Realization	Public	Public	
Source -> Destination	SRS113	SubscriptionOCReplyTa g	
Realization SRS8.1.1 to SubscriptionOCReplyTag	Public	Public	
Source -> Destination	SRS8.1.1	SubscriptionOCReplyTa g	
Realization	Public	Public	
Source -> Destination	SRS123	SubscriptionOCReplyTa g	

Attributes

Attribute	Notes	Constraints and tags
CONTENTVERIFIED String Protected	A flag which indicates whether the request optional content was successfully verified @jameleon.attribute required="true"	*Default:*

Attribute	Notes	Constraints and tags
	default="NotSet"	
Dialog String Protected	The name of the dialog associated with this subscription @jameleon.attribute required="true"	*Default:*
RESPONSETIMEREQUIRED String Protected	The time in milliseconds within which an owner center has to send a response @jameleon.attribute required="false" default="60000"	*Default:*
SUBSCRIPTIONRESPONSEMESSAGE Object Protected	The message response to send @jameleon.attribute required="true"	*Default:*
theInformationLayerController InformationLayerController Private	The information layer controller.	*Default:*

Operations

Method	Notes	Parameters
testBlock() void Public	Test block. Pre-Conditions: N/A Post-Conditions: N/A	

5.3.1.2.48 UserVerifyMessageTag

Type: **Class** C2CRIFunctionTag *Package:* tags

Presents a dialog for the user to verify a message.

@jameleon.function name="USER-VERIFY-MESSAGE" type="action"

@jameleon.step Allow the user to verify a message

Connections

Connector	Source	Target	Notes
Generalization Source -> Destination	Public UserVerifyMessageTag	Public C2CRIFunctionTag	
Realization SRS8.1.1 to UserVerifyMessageTag Source -> Destination	Public SRS8.1.1	Public UserVerifyMessageTag	

Connector	Source	Target	Notes
Realization SRS92 to UserVerifyMessageTag Source -> Destination	Public SRS92	Public UserVerifyMessageTag	

Attributes

Attribute	Notes	Constraints and tags
msgName String Protected	The name of the message to create. @jameleon.attribute required="true"	*Default:*
optionList String Protected	The set of optional NRTM selections, data elements, instances. @jameleon.attribute required="false"	*Default:* ""
userDataFile String Protected	The name/location of the test case data file. @jameleon.attribute required="false"	*Default:* ""

Operations

Method	Notes	Parameters
testBlock() void Public	Process the input variables, present the user verification dialog and process results.	

5.3.1.3 Context Diagram

N/A

5.3.1.4 Variability Guide

The design supports the creation of additional script tags as needed. The currently defined set of script tags are included as part of the Jameleon library as well as NTCIP 2306 and TMDD test suites.

5.3.1.5 Rationale

The C2C RI needs to be able to support repeatable tests. Test scripts provide this capability as they implement the test procedures that were specified for each of the standards.

5.3.2 Standards

5.3.2.1 Primary Presentation

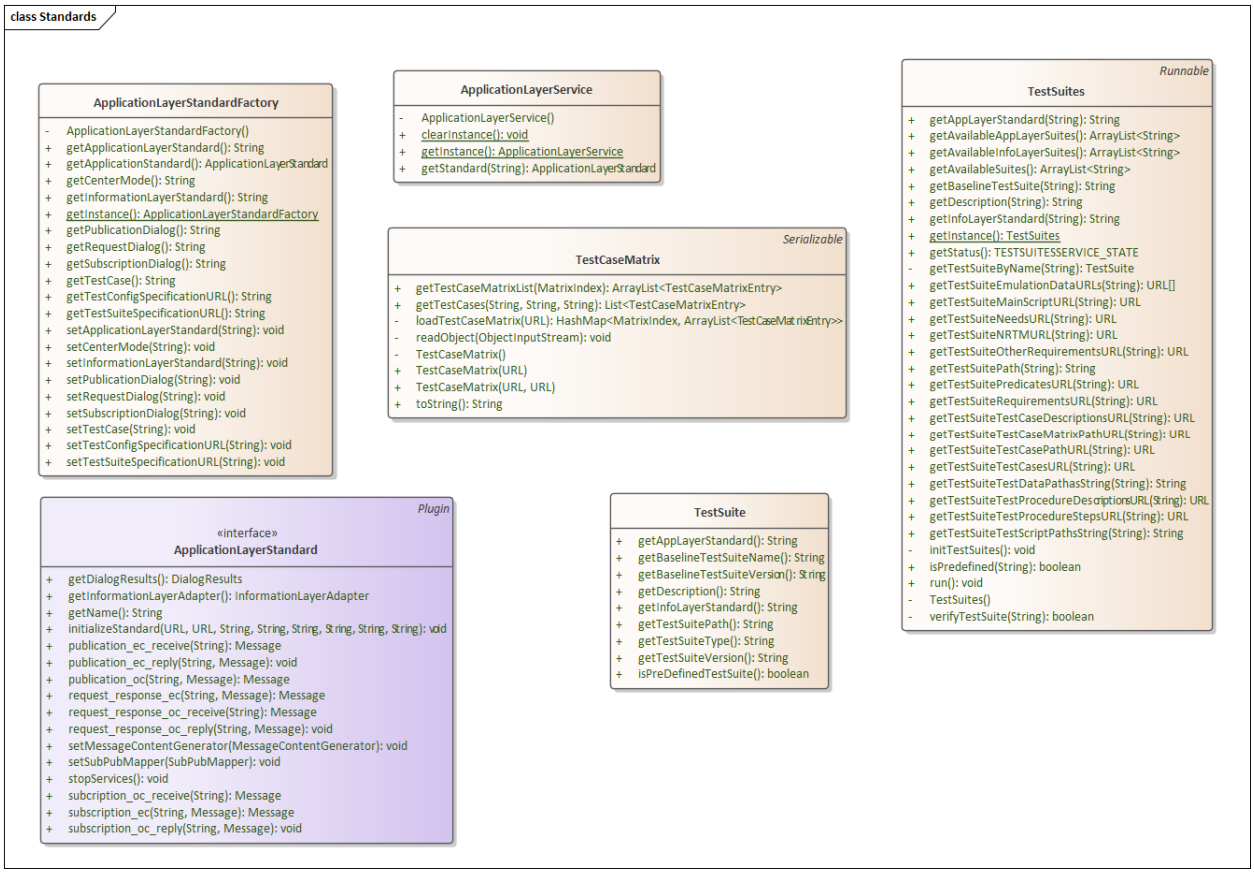

Figure 5-17 Standards Modules

The C2C RI supports Information Layer and Application Layer standards. The detailed test cases, scripts and core standard information are maintained within Test Suites.

5.3.2.2 Element Catalog

5.3.2.2.1 TestCaseMatrix

Type: **Class** Serializable *Package:* testmodel

The Class TestCaseMatrix represents the associations defined between the needs, requirements and test cases.

Connections

Connector	Source	Target	Notes
Association	Public	Private testCaseMatrix	
Source -> Destination	DefaultLayerParameters	TestCaseMatrix	

Connector	Source	Target	Notes
Realization SRS30 to TestCaseMatrix Source -> Destination	Public SRS30	Public TestCaseMatrix	

Attributes

Attribute	Notes	Constraints and tags
headerList ArrayList<String> Public	The header list.	*Default:*
itemType_Index int Private	The item type index.	*Default:*
needID_Index int Private	The need id index.	*Default:*
precondition_Index int Private	The precondition index.	*Default:*
requirementID_Index int Private	The requirement id index.	*Default:*
tcID_Index int Private	The test case id index.	*Default:*
testCaseMatrix HashMap<MatrixIndex, ArrayList<TestCaseMatrixEntry>> Private	The test case matrix.	*Default:*
testCaseMatrixItems ArrayList<TestCaseMatrixEntry> Private	The test case matrix items.	*Default:* new ArrayList<TestCaseMatrixEntry>()

Operations

Method	Notes	Parameters

Method	Notes	Parameters
getTestCaseMatrixList() ArrayList<TestCaseMatrixEntry> Public	Gets the test case matrix list. @return the test case matrix list	MatrixIndex [in] matrixID the matrix id
getTestCases() List<TestCaseMatrixEntry> Public	Gets the test cases. @return the test cases	String [in] needID the need idString [in] requirementID the requirement idString [in] itemType the item type
loadTestCaseMatrix() HashMap<MatrixIndex, ArrayList<TestCaseMatrixEntry>> Private	Load test case matrix. Pre-Conditions: N/A Post-Conditions: N/A @return the hash map	URL [in] testSuiteTCMPath the test suite tcm path
TestCaseMatrix() Public	Instantiates a new test case matrix. Pre-Conditions: N/A Post-Conditions: N/A	URL [in] predefinedTestSuiteTCMPath the predefined test suite tcm pathURL [in] customTestSuiteTCMPath the custom test suite tcm path
TestCaseMatrix() Public	Instantiates a new test case matrix. Pre-Conditions: N/A Post-Conditions: N/A	URL [in] testSuiteNRTMPath the test suite nrtm path
TestCaseMatrix() Private	Instantiates a new test case matrix. Pre-Conditions: N/A Post-Conditions: N/A	
toString() String Public	(non-Javadoc) @see java.lang.Object#toString()	

5.3.2.2.2 ApplicationLayerService

Type: **Class** *Package:* applayer

ApplicationLayerService looks for the specified application layer standard among the set of found plugins.

Connections

Connector	Source	Target	Notes

Connector	Source	Target	Notes
Realization SRS64 to ApplicationLayerService Source -> Destination	Public SRS64	Public ApplicationLayerService	

Attributes

Attribute	Notes	Constraints and tags
loader ServiceLoader<ApplicationLayerStandard> Private	The loader.	*Default:*
service ApplicationLayerService Private Static	The service.	*Default:*

Operations

Method	Notes	Parameters
ApplicationLayerService() Private	Creates a new instance of DictionaryService.	
Static **clearInstance()** void Public	Clear the singleton static instance of ApplicationLayerService.	
Static **getInstance()** ApplicationLayerService Public	Retrieve the singleton static instance of ApplicationLayerService. @return single instance of ApplicationLayerService	
getStandard() ApplicationLayerStandard Public	Retrieve the application layer standard from the first provider that matches the provided standard name. @return the standard	String [in] standard the standard

5.3.2.2.3 ApplicationLayerStandardFactory

Type: **Class** *Package:* applayer

A factory for creating ApplicationLayerStandard objects.

Connections

Connector	Source	Target	Notes
Sequence Source -> Destination	Public ApplicationLayerStandardFactory	Public C2CRISessionTag	
Sequence getApplicationStandard() Source -> Destination	Public C2CRISessionTag	Public ApplicationLayerStandardFactory	
Realization SRS64 to ApplicationLayerStandard Factory Source -> Destination	Public SRS64	Public ApplicationLayerStandardFactory	
Sequence getInstance() Source -> Destination	Public C2CRISessionTag	Public ApplicationLayerStandardFactory	

Attributes

Attribute	Notes	Constraints and tags
applicationLayerStandard String Private	The application layer standard.	*Default:*
centerMode String Private	The center mode.	*Default:*
informationLayerStandard String Private	The information layer standard.	*Default:*
log Logger Protected Static	The log.	*Default:* Logger.getLogger(ApplicationLayerStandardFactory.class.getName())
publicationDialog String Private	The publication dialog.	*Default:*
requestDialog String Private	The request dialog.	*Default:*

Attribute	Notes	Constraints and tags
subscriptionDialog String Private	The subscription dialog.	*Default:*
testCase String Private	The test case.	*Default:*
testConfigSpecificationURL String Private	The test config specification url.	*Default:*
testSuiteSpecificationURL String Private	The test suite specification url.	*Default:*
thisFactory ApplicationLayerStandardFactory Private Static	this factory.	*Default:*

Operations

Method	Notes	Parameters
ApplicationLayerStandardFactory() Private	Instantiates a new application layer standard factory. Pre-Conditions: N/A Post-Conditions: N/A	
getApplicationLayerStandard() String Public	Gets the application layer standard. @return the application layer standard	
getApplicationStandard() ApplicationLayerStandard Public	Gets the application standard. @return the application standard	
getCenterMode() String Public	Gets the center mode. @return the center mode	
getInformationLayerStandar	Gets the information layer standard.	

Method	Notes	Parameters
d() String Public	@return the information layer standard	
Static **getInstance()** ApplicationLayerStandardFactory Public	Gets the single instance of ApplicationLayerStandardFactory. Pre-Conditions: N/A Post-Conditions: N/A @return single instance of ApplicationLayerStandardFactory	
getPublicationDialog() String Public	Gets the publication dialog. @return the publication dialog	
getRequestDialog() String Public	Gets the request dialog. @return the request dialog	
getSubscriptionDialog() String Public	Gets the subscription dialog. @return the subscription dialog	
getTestCase() String Public	Gets the test case. @return the test case	
getTestConfigSpecificationURL() String Public	Gets the test config specification url. @return the test config specification url	
getTestSuiteSpecificationURL() String Public	Gets the test suite specification url. @return the test suite specification url	
setApplicationLayerStandard() void Public	Sets the application layer standard.	String [in] applicationLayerStandard the new application layer standard

Method	Notes	Parameters
setCenterMode() void	Sets the center mode.	String [in] centerMode
Public		the new center mode
setInformationLayerStandard() void	Sets the information layer standard.	String [in] informationLayerStandard
Public		the new information layer standard
setPublicationDialog() void	Sets the publication dialog.	String [in] publicationDialog
Public		the new publication dialog
setRequestDialog() void	Sets the request dialog.	String [in] requestDialog
Public		the new request dialog
setSubscriptionDialog() void	Sets the subscription dialog.	String [in] subscriptionDialog
Public		the new subscription dialog
setTestCase() void	Sets the test case.	String [in] testCase
Public		the new test case
setTestConfigSpecificationURL() void	Sets the test config specification url.	String [in] testConfigSpecificationURL
Public		the new test config specification url
setTestSuiteSpecificationURL() void	Sets the test suite specification url.	String [in] testSuiteSpecificationURL
Public		the new test suite specification url

5.3.2.2.4 ApplicationLayerStandard

Type: **Interface** Plugin *Package:* applayer

The Interface ApplicationLayerStandard.

Connections

Connector	Source	Target	Notes
Sequence	Public	Public	
Source -> Destination	ApplicationLayerStandard	InformationLayerController	
Sequence	Public	Public	
Source -> Destination	ApplicationLayerStandard	InformationLayerStandard	
Sequence	Public	Public	
Source -> Destination	ApplicationLayerStandard	InformationLayerController	
Sequence	Public	Public	
Source -> Destination	ApplicationLayerStanda	InformationLayerStanda	

Connector	Source	Target	Notes
	rd	rd	
Sequence	Public	Public	
Source -> Destination	ApplicationLayerStandard	InformationLayerController	
Sequence	Public	Public	
Source -> Destination	ApplicationLayerStandard	InformationLayerController	
Sequence Response from the OC	Public	Public	
Source -> Destination	ApplicationLayerStandard	External System Under Test	
Association	Public	Private	
Source -> Destination	InformationLayerStandardFactory	applicationLayerStandard	
		ApplicationLayerStandard	
Realization	Public	Public	
Source -> Destination	SRS137	ApplicationLayerStandard	
Realization	Public	Public	
Source -> Destination	SRS132	ApplicationLayerStandard	
Realization	Public	Public	
Source -> Destination	SRS117.5	ApplicationLayerStandard	
Realization	Public	Public	
Source -> Destination	SRS119	ApplicationLayerStandard	
Realization	Public	Public	
Source -> Destination	NTCIP2306ApplicationLayerStandard	ApplicationLayerStandard	
Realization	Public	Public	
Source -> Destination	SRS110	ApplicationLayerStandard	
Realization	Public	Public	
Source -> Destination	SRS138	ApplicationLayerStandard	
Realization	Public	Public	
Source -> Destination	SRS130	ApplicationLayerStandard	
Realization	Public	Public	
Source -> Destination	SRS111	ApplicationLayerStanda	

Connector	Source	Target	Notes
		rd	
Realization	Public	Public	
Source -> Destination	SRS117	ApplicationLayerStandard	
Realization	Public	Public	
Source -> Destination	SRS123	ApplicationLayerStandard	
Realization	Public	Public	
Source -> Destination	SRS142	ApplicationLayerStandard	
Association	Public	Private	
Source -> Destination	RISessionTag	theApplicationLayerStandard	
		ApplicationLayerStandard	
Association	Public	Private	
Source -> Destination	TestRequestResponseOC	theApplicationLayerStandard	
		ApplicationLayerStandard	
Realization	Public	Public	
Source -> Destination	SRS112	ApplicationLayerStandard	
Association	Public	Private	
Source -> Destination	TestRequestResponseEC	theApplicationLayerStandard	
		ApplicationLayerStandard	
Realization	Public	Public	
Source -> Destination	SRS113	ApplicationLayerStandard	
Sequence	Public	Public	
request_response_oc_reply (dialog, responseMessage)	InformationLayerController	ApplicationLayerStandard	
Source -> Destination			
Realization	Public	Public	
Source -> Destination	SRS134	ApplicationLayerStandard	
Realization	Public	Public	
Source -> Destination	SRS127	ApplicationLayerStandard	
Realization	Public	Public	

Connector	Source	Target	Notes
Source -> Destination	SRS116	ApplicationLayerStandard	
Realization	Public	Public	
Source -> Destination	SRS131	ApplicationLayerStandard	
Realization	Public	Public	
Source -> Destination	SRS115	ApplicationLayerStandard	
Sequence	Public	Public	
initializeStandard(serviceSpecificationURL, baseStandardSpecificationURL, centerMode, informationLayerStandard, testCaseName, requestDialog, subscriptionDialog, publicationDialog)	InformationLayerStandard	ApplicationLayerStandard	
Source -> Destination			
Sequence	Public	Public	
request_response_oc_receive(dialog)	InformationLayerController	ApplicationLayerStandard	
Source -> Destination			
Realization	Public	Public	
Source -> Destination	SRS135	ApplicationLayerStandard	
Realization	Public	Public	
Source -> Destination	SRS108	ApplicationLayerStandard	
Realization	Public	Public	
Source -> Destination	SRS140	ApplicationLayerStandard	
Realization	Public	Public	
Source -> Destination	SRS122	ApplicationLayerStandard	
Realization	Public	Public	
Source -> Destination	SRS126	ApplicationLayerStandard	
Association	Public	Private	
Source -> Destination	TestSubscriptionOC	theApplicationLayerStandard	
		ApplicationLayerStandard	

Connector	Source	Target	Notes
Sequence request_response_oc_reply (dialog, responseMessage) Source -> Destination	Public InformationLayerController	Public ApplicationLayerStandard	
Realization Source -> Destination	Public SRS114	Public ApplicationLayerStandard	
Sequence stopServices() Source -> Destination	Public InformationLayerStandard	Public ApplicationLayerStandard	
Realization Source -> Destination	Public SRS129	Public ApplicationLayerStandard	
Association Source -> Destination	Public C2CRISessionTag	Private theApplicationLayerStandard ApplicationLayerStandard	
Realization Source -> Destination	Public SRS121	Public ApplicationLayerStandard	
Realization Source -> Destination	Public SRS136	Public ApplicationLayerStandard	
Realization Source -> Destination	Public SRS116.3	Public ApplicationLayerStandard	
Realization Source -> Destination	Public SRS128	Public ApplicationLayerStandard	
Realization Source -> Destination	Public SRS133	Public ApplicationLayerStandard	
Realization Source -> Destination	Public SRS120	Public ApplicationLayerStandard	
Association Source -> Destination	Public TestPublicationOC	Private theApplicationLayerStandard ApplicationLayerStandard	

Connector	Source	Target	Notes
Sequence request_response_oc_receive(dialog) Source -> Destination	Public InformationLayerController	Public ApplicationLayerStandard	
Association Source -> Destination	Public TestPublicationEC	Private theApplicationLayerStandard ApplicationLayerStandard	
Association Source -> Destination	Public TMDDv303InformationLayerStandard	Private appStandard ApplicationLayerStandard	
Realization Source -> Destination	Public SRS109	Public ApplicationLayerStandard	
Realization Source -> Destination	Public SRS118	Public ApplicationLayerStandard	
Realization Source -> Destination	Public SRS107	Public ApplicationLayerStandard	
Sequence Request from the EC Source -> Destination	Public External System Under Test	Public ApplicationLayerStandard	
Association Source -> Destination	Public TestSubscriptionEC	Private theApplicationLayerStandard ApplicationLayerStandard	
Realization Source -> Destination	Public SRS139	Public ApplicationLayerStandard	
Realization Source -> Destination	Public SRS124	Public ApplicationLayerStandard	
Association Source -> Destination	Public CenterMonitor	Private appStandard ApplicationLayerStandard	
Realization Source -> Destination	Public SRS141	Public ApplicationLayerStandard	

Connector	Source	Target	Notes
Realization	Public	Public	
Source -> Destination	SRS125	ApplicationLayerStandard	

Operations

Method	Notes	Parameters
getDialogResults() DialogResults Public	Gets the dialog results. @return the dialog results	
getInformationLayerAdapter() InformationLayerAdapter Public	Gets the information layer adapter. @return The InformationLayerAdapter Interface provided by this standard.	
getName() String Public	Gets the name. @return The name of the standard.	
initializeStandard() void Public	Initialize standard. Pre-Conditions: N/A Post-Conditions: N/A	URL [in] serviceSpecificationURL the service specification urlURL [in] baseStandardSpecificationURL the base standard specification urlString [in] centerMode the center modeString [in] informationLayerStandard the information layer standardString [in] testCaseName the test case nameString [in] requestDialog the request dialogString [in] subscriptionDialog the subscription dialogString [in] publicationDialog the publication dialog
publication_ec_receive() Message Public	Publication_ec_receive. Pre-Conditions: N/A Post-Conditions: N/A @return the message	String [in] dialog the dialog

Method	Notes	Parameters
publication_ec_reply() void Public	Publication_ec_reply. Pre-Conditions: N/A Post-Conditions: N/A	String [in] dialog the dialogMessage [in] responseMessage the response message
publication_oc() Message Public	Publication_oc. Pre-Conditions: N/A Post-Conditions: N/A @return the message	String [in] dialog the dialogMessage [in] publicationMessage the publication message
request_response_ec() Message Public	Request_response_ec. Pre-Conditions: N/A Post-Conditions: N/A @return the message	String [in] dialog the dialogMessage [in] requestMessage the request message
request_response_oc_receive() Message Public	Request_response_oc_receive. Pre-Conditions: N/A Post-Conditions: N/A @return the message	String [in] dialog the dialog
request_response_oc_reply() void Public	Request_response_oc_reply. Pre-Conditions: N/A Post-Conditions: N/A	String [in] dialog the dialogMessage [in] responseMessage the response message
setMessageContentGenerator() void Public	Sets the message content generator.	MessageContentGenerator [in] msgGenerator the new message content generator
setSubPubMapper() void Public	Sets the sub pub mapper.	SubPubMapper [in] subPubMapper the new sub pub mapper
stopServices() void Public	Stop services. Pre-Conditions: N/A Post-Conditions: N/A	
subcription_oc_receive() Message Public	Subcription_oc_receive. Pre-Conditions: N/A Post-Conditions: N/A @return the message	String [in] dialog the dialog
subscription_ec() Message Public	Subscription_ec. Pre-Conditions: N/A Post-Conditions: N/A @return the message	String [in] dialog the dialogMessage [in] subscriptionMessage

Method	Notes	Parameters
		the subscription message
subscription_oc_reply() void Public	Subscription_oc_reply. Pre-Conditions: N/A Post-Conditions: N/A	String [in] dialog the dialogMessage [in] responseMessage the response message

5.3.2.2.5 TestSuites

Type: **Class** Runnable *Package:* testmodel

This class establishes and maintains the Test Suites that are available to the RI. It also implements JasperReports API interface to enable the reporting of Test Suite information (scripts).

Connections

Connector	Source	Target	Notes
Association Source -> Destination	Public TestSuites	Private testSuitesServiceState TESTSUITESSERVICE_STATE	
Association Source -> Destination	Public TestConfiguration	Private availableSuites TestSuites	
Realization SRS64 to TestSuites Source -> Destination	Public SRS64	Public TestSuites	
Realization SRS30.4 to TestSuites Source -> Destination	Public SRS30.4	Public TestSuites	
Realization SRS85 to TestSuites Source -> Destination	Public SRS85	Public TestSuites	
Realization SRS30 to TestSuites Source -> Destination	Public SRS30	Public TestSuites	
Realization SRS68 to TestSuites Source -> Destination	Public SRS68	Public TestSuites	
Realization SRS30.2 to TestSuites Source -> Destination	Public SRS30.2	Public TestSuites	

Connector	Source	Target	Notes
Realization SRS30.3 to TestSuites Source -> Destination	Public SRS30.3	Public TestSuites	

Attributes

Attribute	Notes	Constraints and tags
availableAppLayerSuites ArrayList<String> Private	The available app layer suites.	*Default:*
availableInfoLayerSuites ArrayList<String> Private	The available info layer suites.	*Default:*
availableSuites ArrayList<String> Private	The list of available Test Suites.	*Default:*
testSuites TestSuites Private Static	The test suites.	*Default:*
testSuitesList ArrayList<TestSuite> Private	The test suites list.	*Default:*
testSuitesServiceState TESTSUITESSERVICE_STATE Private Static	The test suites service state.	*Default:* TESTSUITESSERVICE_STATE.INITIAL

Operations

Method	Notes	Parameters
getAppLayerStandard() String Public	returns the application layer standard associated with the test suite. @return the app layer standard	String [in] theSuite the the suite
getAvailableAppLayerSuites() ArrayList<String>	returns the list of Application Layer test suites that are available to the RI application.	

Method	Notes	Parameters
Public	@return the available app layer suites	
getAvailableInfoLayerSuites() ArrayList<String> Public	returns the list of Information Layer test suites that are available to the RI application. @return the available info layer suites	
getAvailableSuites() ArrayList<String> Public	returns the list of test suites that are available to the RI application. @return the available suites	
getBaselineTestSuite() String Public	returns the information layer standard associated with the test suite. @return the baseline test suite	String [in] theSuite the the suite
getDescription() String Public	returns the Test Suite Description associated with the test suite. @return the description	String [in] theSuite the the suite
getInfoLayerStandard() String Public	returns the information layer standard associated with the test suite. @return the info layer standard	String [in] theSuite the the suite
Static **getInstance()** TestSuites Public	Gets the single instance of TestSuites. Pre-Conditions: N/A Post-Conditions: N/A @return single instance of TestSuites	
getStatus() TESTSUITESSERVICE_STATE Public	Gets the status. @return the status	
getTestSuiteMainScriptURL() URL	Gets the test suite main script url.	String [in] theSuite

Method	Notes	Parameters
Public	@return the test suite main script url	the the suite
getTestSuiteNeedsURL() URL Public	returns the URL associated with the test suites Needs CSV File. @return the test suite needs url	String [in] theSuite the the suite
getTestSuiteNRTMURL() URL Public	returns the URL associated with the test suites NRTM CSV File. @return the test suite nrtmurl	String [in] theSuite the the suite
getTestSuiteOtherRequirementsURL() URL Public	returns the URL associated with the test suites OtherRequirements CSV File. @return the test suite other requirements url	String [in] theSuite the the suite
getTestSuitePath() String Public	returns the directory path associated with the test suite. @return the test suite path	String [in] theSuite the the suite
getTestSuitePredicatesURL() URL Public	returns the URL associated with the test suites Predicates CSV File. @return the test suite predicates url	String [in] theSuite the the suite
getTestSuiteRequirementsURL() URL Public	returns the URL associated with the test suites Requirements CSV File. @return the test suite requirements url	String [in] theSuite the the suite
getTestSuiteTestCaseMatrixPathURL() URL Public	returns the URL associated with the test suites TestCaseMatrix CSV File. @return the test suite test case matrix path url	String [in] theSuite the the suite

Method	Notes	Parameters
getTestSuiteTestCasePathURL() URL Public	returns the URL associated with the test suites TestCases CSV File. @return the test suite test case path url	String [in] theSuite the the suite
getTestSuiteTestCasesURL() URL Public	returns the URL associated with the test suites TestCases CSV File. @return the test suite test cases url	String [in] theSuite the the suite
getTestSuiteTestDataPathasString() String Public	returns the String representation of the URL associated with the test suites Data Path. @return the test suite test data pathas string	String [in] theSuite the the suite
getTestSuiteTestScriptPathsString() String Public	returns the String representation of the URL associated with the test suites Scripts Path. @return the test suite test script paths string	String [in] theSuite the the suite
initTestSuites() void Private	Inits the test suites. Pre-Conditions: N/A Post-Conditions: N/A	
isPredefined() boolean Public	returns a flag indicating whether the test suite is predefined. @return true, if is predefined	String [in] theSuite the the suite
run() void Public	(non-Javadoc) @see java.lang.Runnable#run()	
TestSuites() Private	Constructor operation for Test Suites. It first searches for the Test Suites distributed with the RI. It performs a validity check on the Test Suites and adds them to it's list of Test Suites if valid. It then searches for user provided Test Suites (using an RI Parameter for the location). Any valid user provided Test Suites found are stored in the list with a special designation for user provided.	

Method	Notes	Parameters
verifyTestSuite() boolean Private	Performs a validity check on a Test Suite. For Test Suites at the distribution location an additional check will be performed to ensure the configuration is properly signed/encoded. @return true, if successful	String [in] testSuite the test suite

5.3.2.2.5.1 TestSuite

Type: **Class** *Package:* testmodel

The TestSuite Class represents each Test Suite recognized by the RI as being loaded on the application platform.

Connections

Connector	Source	Target	Notes
Realization SRS85.2 to TestSuite Source -> Destination	Public SRS85.2	Public TestSuite	
Realization SRS85 to TestSuite Source -> Destination	Public SRS85	Public TestSuite	
Realization SRS30 to TestSuite Source -> Destination	Public SRS30	Public TestSuite	
Realization SRS85.1 to TestSuite Source -> Destination	Public SRS85.1	Public TestSuite	
Realization SRS64 to TestSuite Source -> Destination	Public SRS64	Public TestSuite	
Realization SRS91 to TestSuite Source -> Destination	Public SRS91	Public TestSuite	
Realization SRS68 to TestSuite Source -> Destination	Public SRS68	Public TestSuite	

Attributes

Attribute	Notes	Constraints and tags
appLayerStandard String Protected	The app layer standard.	*Default:* ""
baselineTestSuiteName String Protected	The baseline test suite name.	*Default:* ""
baselineTestSuiteVersion String Protected	The baseline test suite version.	*Default:* ""
description String Protected	The description.	*Default:* ""
infoLayerStandard String Protected	The info layer standard.	*Default:* ""
preDefinedTestSuite boolean Protected	The pre defined test suite.	*Default:* false
testSuiteName String Protected	The test suite name.	*Default:* ""
testSuitePath String Protected	The test suite path.	*Default:* ""
testSuiteType String Protected	The test suite type.	*Default:* ""
testSuiteVersion String Protected	The test suite version.	*Default:* ""

Operations

Method	Notes	Parameters
getAppLayerStandard() String	Gets the app layer standard. @return the app layer standard	

Method	Notes	Parameters
Public		
getBaselineTestSuiteName() String Public	Gets the baseline test suite name. @return the baseline test suite name	
getBaselineTestSuiteVersion() String Public	Gets the baseline test suite version. @return the baseline test suite version	
getDescription() String Public	Gets the description. @return the description	
getInfoLayerStandard() String Public	Gets the info layer standard. @return the info layer standard	
getTestSuitePath() String Public	Gets the test suite path. @return the test suite path	
getTestSuiteType() String Public	Gets the test suite type. @return the test suite type	
getTestSuiteVersion() String Public	Gets the test suite version. @return the test suite version	
isPreDefinedTestSuite() boolean Public	Checks if is pre defined test suite. Pre-Conditions: N/A Post-Conditions: N/A @return true, if is pre defined test suite	

5.3.2.3 Context Diagram

N/A

5.3.2.4 Variability Guide

N/A

5.3.2.5 Rationale

N/A

5.3.3 Test User Profile

The test user profile functionality has not been implemented due to lack of requirements for it.

5.3.4 Interface Spec

5.3.4.1 Primary Presentation

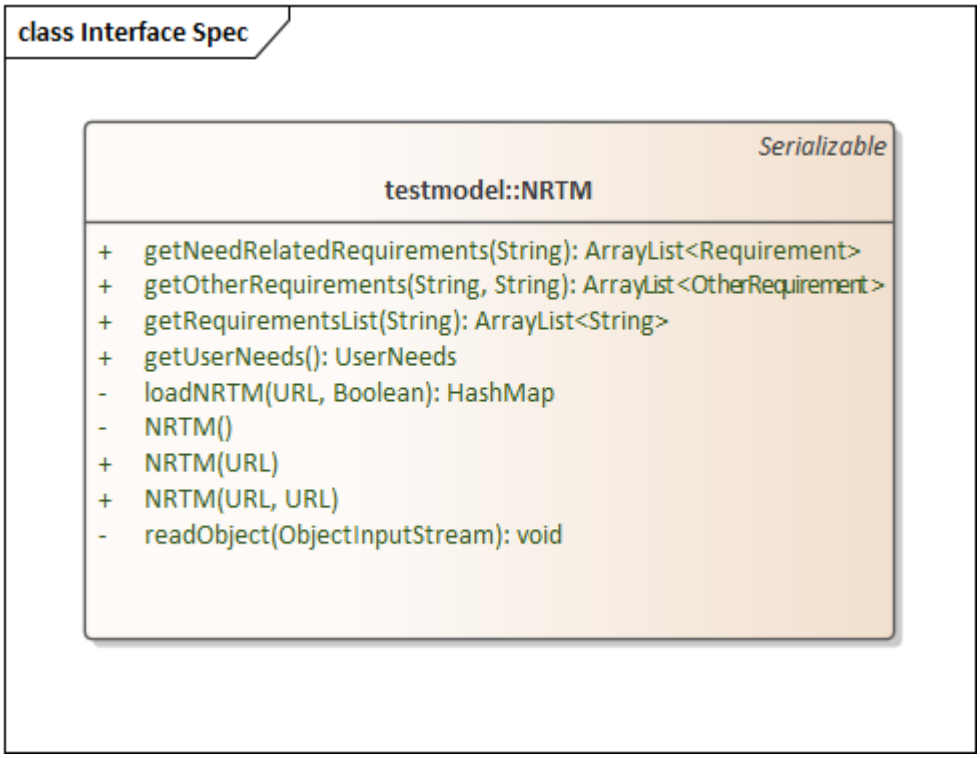

Figure 5-18 Interface Spec Modules

The NRTM module captures and provides the users NRTM selections.

5.3.4.2 Element Catalog

5.3.4.2.1 NRTM

Type: **Class** Serializable *Package:* testmodel

The Class NRTM represents a single entry in the NRTM.

Connections

Connector	Source	Target	Notes
Association	Public	Private userNeeds	
Source -> Destination	NRTM	UserNeeds	
Association	Public	Private nrtm	
Source -> Destination	NeedsTableModel	NRTM	
Association	Public	Private standardNRTM	
Source -> Destination	TestDefinitionCoordinator	NRTM	
Association	Public	Private standardAppNRTM	
Source -> Destination	TestDefinitionCoordinator	NRTM	
Realization SRS30 to NRTM	Public	Public	
Source -> Destination	SRS30	NRTM	
Association	Public	Private nrtm	
Source -> Destination	OtherRequirementsTableModel	NRTM	
Association	Public	Private nrtm	
Source -> Destination	RequirementsTableModel	NRTM	
Association	Public	Private nrtm	
Source -> Destination	DefaultLayerParameters	NRTM	

Attributes

Attribute	Notes	Constraints and tags
conformance_Index int Private	The conformance_ index.	*Default:*
headerList ArrayList<String> Public	The header list.	*Default:*

Attribute	Notes	Constraints and tags
id_Index int Private	The id_ index.	*Default:*
needConformance_Index int Private	The need conformance_ index.	*Default:*
needFlag_Index int Private	The need flag_ index.	*Default:*
needFlagValue_Index int Private	The need flag value_ index.	*Default:*
needSupport_Index int Private	The need support_ index.	*Default:*
nrtm HashMap Private	The nrtm.	*Default:*
otherReqParameter_Index int Private	The other req parameter_ index.	*Default:*
otherRequirements_Index int Private	The other requirements_ index.	*Default:*
otherRequirementValues_Index int Private	The other requirement values_ index.	*Default:*
reqFlag_Index int Private	The req flag_ index.	*Default:*
reqFlagValue_Index int Private	The req flag value_ index.	*Default:*
requirement_Index int	The requirement_ index.	*Default:*

Attribute	Notes	Constraints and tags
Private		
requirementID_Index int Private	The requirement i d_ index.	*Default:*
support_Index int Private	The support_ index.	*Default:*
uNID_Index int Private	The u ni d_ index.	*Default:*
userNeed_Index int Private	The user need_ index.	*Default:*
userNeeds UserNeeds Private	Temporary Constructors to transition from segmented NRTM to united NRTM.	*Default:* new UserNeeds()

Operations

Method	Notes	Parameters
getNeedRelatedRequirements () ArrayList<Requirement> Public	returns the current information layer Project Requirements. @return the need related requirements	String [in] needID the need id
getOtherRequirements() ArrayList<OtherRequirement> Public	returns the current information layer Project Requirements. @return the other requirements	String [in] needID the need idString [in] requirementID the requirement id
getRequirementsList() ArrayList<String> Public	Gets the requirements list. @return the requirements list	String [in] needID the need id
getUserNeeds() UserNeeds Public	returns the current information layer User Needs. @return the user needs	

Method	Notes	Parameters
loadNRTM() HashMap Private	Load nrtm. Pre-Conditions: N/A Post-Conditions: N/A @return the hash map	URL [in] testSuiteNRTMPath the test suite nrtm pathBoolean [in] extension the extension
NRTM() Private	Instantiates a new nrtm. Pre-Conditions: N/A Post-Conditions: N/A	
NRTM() Public	Instantiates a new nrtm. Pre-Conditions: N/A Post-Conditions: N/A	URL [in] testSuiteNRTMPath the test suite nrtm path
NRTM() Public	Instantiates a new nrtm. Pre-Conditions: N/A Post-Conditions: N/A	URL [in] predefinedTestSuiteNRTMPath the predefined test suite nrtm pathURL [in] customTestSuiteNRTMPath the custom test suite nrtm path

5.3.4.3 Context Diagram

N/A

5.3.4.4 Variability Guide

The NRTM module is constant. However, the data utilized by the module is provided as part of the Test Configuration File which is derived from the Test Suites.

5.3.4.5 Rationale

The C2C RI needs to provide the user with access to each standards NRTM information in order to make project related selections.

5.3.5 SUT

The SUT functionality is incorporated within Test Configuration files.

5.3.6 Configuration Manager

5.3.6.1 Primary Presentation

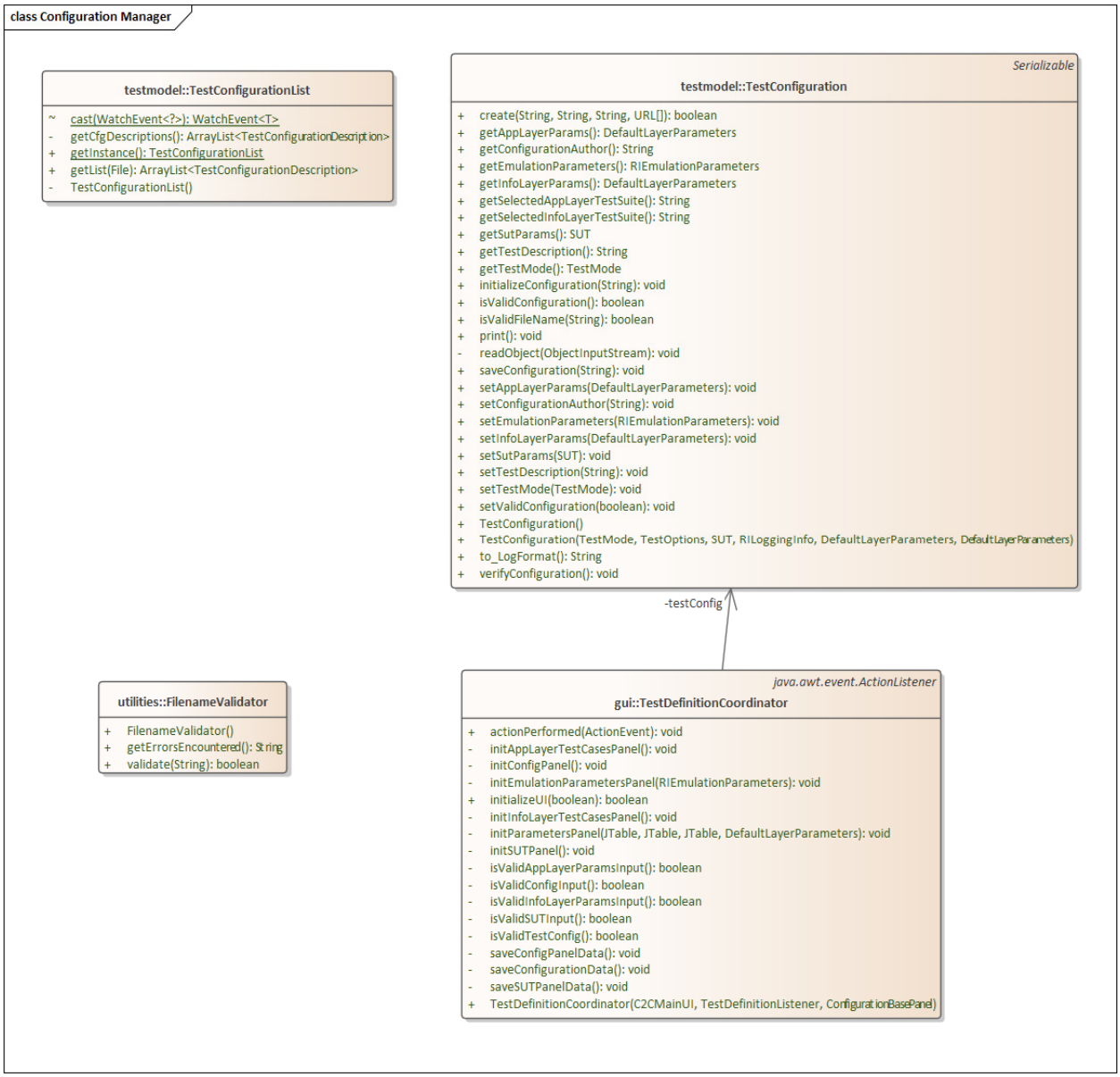

Figure 5-19 Configuration Manager Modules

The Configuration Manager Modules provide the systems access and control of Test Configuration files.

5.3.6.2 Element Catalog

5.3.6.2.1 TestConfiguration

Type: **Class** Serializable *Package:* testmodel

This class represents all information and operations associated with a Test Configuration.

Tagged Values

Ÿ annotations = @XmlRootElement(name = "testConfiguration")@XmlAccessorType(XmlAccessType.FIELD).

Connections

Connector	Source	Target	Notes
Association	Public	Private availableSuites	
Source -> Destination	TestConfiguration	TestSuites	
Association	Public	Private appLayerParams	
Source -> Destination	TestConfiguration	DefaultLayerParameters	
Association	Public	Private testingOptions	
Source -> Destination	TestConfiguration	TestOptions	
Association	Public	Private infoLayerParams	
Source -> Destination	TestConfiguration	DefaultLayerParameters	
Association	Public	Private sutParams	
Source -> Destination	TestConfiguration	SUT	
Association	Public	Private logInfo	
Source -> Destination	TestConfiguration	RILoggingInfo	
Association	Public	Private testMode	
Source -> Destination	TestConfiguration	TestMode	
Association	Public	Private testConfiguration	
Source -> Destination	RIExecuteTestCase	TestConfiguration	
Association	Public	Private testConfig	
Source -> Destination	RITestEngine	TestConfiguration	
Association	Public	Private testConfiguration	
Source -> Destination	TestConfigCreateAction	TestConfiguration	
Association	Public	Private testConfiguration	
Source -> Destination	TestConfigCreateUI	TestConfiguration	
Association	Public	Package selectedTestConfigObj	
Source -> Destination	RIReports	TestConfiguration	

Connector	Source	Target	Notes
Association Source -> Destination	Public ReportCreateAction	Package config TestConfiguration	
Association Source -> Destination	Public TestDefinitionCoordinator	Private testConfig TestConfiguration	
Realization SRS22 to TestConfiguration Source -> Destination	Public SRS22	Public TestConfiguration	

Attributes

Attribute	Notes	Constraints and tags
appLayerParams DefaultLayerParameters Private	reference to the application layer parameter information contained within this test configuration.	*Default:*
availableSuites TestSuites Private	a reference to the currently available test suites recognized by the RI.	*Default:*
configurationAuthor String Private	The RI user that saved/created this test configuration.	*Default:*
configVersion long Private	the format version of this Test Configuration.	*Default:* 1
infoLayerParams DefaultLayerParameters Private	reference to the information layer parameter data contained within this test configuration.	*Default:*
logInfo org.fhwa.c2cri.logger.RILoggingInfo Private	a reference to the currently available test logging data.	*Default:*
riEmulationParams RIEmulationParameters Private	The user specified parameters that would be used during entity emulation.	*Default:*
selectedAppLayerTestSuite	The Application Layer test suite selected for	*Default:*

Attribute	Notes	Constraints and tags
String Private	this test configuration.	
selectedInfoLayerTestSuite String Private	The information Layer test suite selected for this test configuration.	*Default:*
sutParams SUT Private	reference to the System Under Test information contained within this test configuration.	*Default:*
testDescription String Private	description information for this test configuration.	*Default:*
testingOptions TestOptions Private	reference to the TestOptions information contained within this test configuration.	*Default:*
testMode TestMode Private	reference to the Test Mode selection information contained within this test configuration.	*Default:*
validConfiguration boolean Private	flag indicating whether this test configuration has been validated.	*Default:* false

Operations

Method	Notes	Parameters
create() boolean Public	Creates the. Pre-Conditions: N/A Post-Conditions: N/A @return true, if successful	String [in] configDescription the config descriptionString [in] infoLayerTestSuite the info layer test suiteString [in] appLayerTestSuite the app layer test suite
getAppLayerParams() DefaultLayerParameters Public	Gets the app layer params. @return the appLayerParams	
getConfigurationAuthor() String Public	Gets the configuration author. @return the configuration author	

Method	Notes	Parameters
getInfoLayerParams() DefaultLayerParameters Public	Gets the info layer params. @return the infoLayerParams	
getLogInfo() RILoggingInfo Public	Gets the log info. @return the log info	
getSelectedAppLayerTestSuite() String Public	Gets the selected app layer test suite. @return the selected app layer test suite	
getSelectedInfoLayerTestSuite() String Public	Gets the selected info layer test suite. @return the selected info layer test suite	
getSutParams() SUT Public	Gets the sut params. @return the sutParams	
getTestDescription() String Public	Gets the test description. @return the test description	
getTestingOptions() TestOptions Public	Gets the testing options. @return the testingOptions	
getTestMode() TestMode Public	Gets the test mode. @return the testMode	
initializeConfiguration() void Public	Load the specified test configuration file. If no file name is provided then initialize the test configuration using all default	String [in] configurationFile the configuration file

Method	Notes	Parameters
	parameters. If there is a problem encountered loading the test configuration throw an exception and indicate what problems were encountered.	
isValidConfiguration() boolean Public	Checks if is valid configuration. Pre-Conditions: N/A Post-Conditions: N/A @return true, if is valid configuration	
isValidFileName() boolean Public	Checks whether the confiig file provided is valid. @return true, if is valid file name	String [in] fileName the file name
print() void Public	Prints the. Pre-Conditions: N/A Post-Conditions: N/A	
saveConfiguration() void Public	Save the test configuration to the specified test configuration file. If no file name is provided then throw an exception. If the file exists, warn the user about an overwrite condition. If there is a problem encountered while storing the test configuration throw an exception and indicate what problems were encountered. Set the configurationAuthor field to the current RI users user name. The TestConfiguration is saved as an object, so all local and associated information will be stored in the file.	String [in] configurationFile the configuration file
setAppLayerParams() void Public	Sets the app layer params.	DefaultLayerParameters [in] appLayerParams the appLayerParams to set
setConfigurationAuthor() void Public	Sets the configuration author.	String [in] configurationAuthor the new configuration author
setInfoLayerParams() void Public	Sets the info layer params.	DefaultLayerParameters [in] infoLayerParams the infoLayerParams to set
setLogInfo() void Public	Sets the log info.	RILoggingInfo [in] logInfo the new log info
setSutParams() void	Sets the sut params.	SUT [in] sutParams

Method	Notes	Parameters
Public		the sutParams to set
setTestDescription() void	Sets the test description.	String [in] testDescription
Public		the new test description
setTestingOptions() void	Sets the testing options.	TestOptions [in] testingOptions
Public		the testingOptions to set
setTestMode() void	Sets the test mode.	TestMode [in] testMode
Public		the testMode to set
setValidConfiguration() void	Sets the valid configuration.	boolean [in] validConfiguration
Public		the new valid configuration
TestConfiguration()	TestConfiguration basic Constructor.	
Public		
TestConfiguration()	TestConfiguration basic Constructor.	TestMode [in] modes
Public		the modesTestOptions [in] options
		the optionsSUT [in] sut
		the sutRILoggingInfo [in] log
		the logDefaultLayerParameters [in] apps
		the appsDefaultLayerParameters [in] info
		the info
to_LogFormat() String Public	Converts the Test Configuration into a form that can be written to and read from the Test Log. @return the string	
verifyConfiguration() void Public	Check the elements of the configuration file to confirm that all elements are valid.	

5.3.6.2.2 TestConfigurationList

Type: **Class** *Package:* testmodel

The Class TestConfigurationList maintains a running list of Configuration Files.

Connections

Connector	Source	Target	Notes

Connector	Source	Target	Notes
Association	Public	Private configUpdater	
Source -> Destination	TestConfigurationList	ConfigFileUpdater	
Realization SRS48.1 to TestConfigurationList	Public	Public	
Source -> Destination	SRS48.1	TestConfigurationList	

Attributes

Attribute	Notes	Constraints and tags
cfgDescriptions ArrayList<TestConfigurationDescription> Private Static	The cfg descriptions.	*Default:* new ArrayList<>()

configFilter FilenameFilter
Package
	The config filter.	*Default:* new FilenameFilter() {
		@Override
		public boolean accept(File dir, String name) {
		if ((name.length()>5) && (name.toUpperCase().endsWith(".RI CFG"))) {
		return true;
		}
		return false;
		}
		}

configUpdater ConfigFileUpdater Private Static	The config updater.	*Default:*
directory File Private Static	The directory.	*Default:*
executor ExecutorService Package Const	The executor.	*Default:* Executors.newCachedThreadPool()
NTHREDS int Private	The Constant NTHREDS.	*Default:* 4

Attribute	Notes	Constraints and tags
Static Const		
thisInstance TestConfigurationList Private Static	The this instance.	*Default:* new TestConfigurationList()

Operations

Method	Notes	Parameters
Static **cast()** WatchEvent<T> Package	Cast. Pre-Conditions: N/A Post-Conditions: N/A @param <T> the generic type @return the watch event	WatchEvent<?> [in] event the event
getCfgDescriptions() ArrayList<TestConfigurationDescription> Private	Gets the cfg descriptions. @return the cfg descriptions	
Static **getInstance()** TestConfigurationList Public	Gets the single instance of TestConfigurationList. Pre-Conditions: N/A Post-Conditions: N/A @return single instance of TestConfigurationList	
getList() ArrayList<TestConfigurationDescription> Public	Gets the list. @return the list	File [in] dir the dir
TestConfigurationList() Private	Instantiates a new test configuration list. Pre-Conditions: N/A Post-Conditions: N/A	

5.3.6.2.3 TestDefinitionCoordinator

Type: **Class** java.awt.event.ActionListener *Package:* gui

The Class TestDefinitionCoordinator.

Connections

Connector	Source	Target	Notes
Association Source -> Destination	Public TestDefinitionCoordinator	Private standardNRTM NRTM	
Association Source -> Destination	Public TestDefinitionCoordinator	Private parentObject TestDefinitionListener	
Association Source -> Destination	Public TestDefinitionCoordinator	Private standardAppNRTM NRTM	
Association Source -> Destination	Public TestDefinitionCoordinator	Private corePanel ConfigurationBasePanel	
Association Source -> Destination	Public TestDefinitionCoordinator	Private mainUI C2CMainUI	
Association Source -> Destination	Public TestDefinitionCoordinator	Private testConfig TestConfiguration	
Realization SRS20.3 to TestDefinitionCoordinator Source -> Destination	Public SRS20.3	Public TestDefinitionCoordinator	
Realization SRS13.1 to TestDefinitionCoordinator Source -> Destination	Public SRS13.1	Public TestDefinitionCoordinator	
Realization SRS21.3 to TestDefinitionCoordinator Source -> Destination	Public SRS21.3	Public TestDefinitionCoordinator	
Realization SRS17 to TestDefinitionCoordinator Source -> Destination	Public SRS17	Public TestDefinitionCoordinator	
Association Source -> Destination	Public RIUICoordinator	Private definitionUICoordinator TestDefinitionCoordinator	
Realization SRS84 to TestDefinitionCoordinator Source -> Destination	Public SRS84	Public TestDefinitionCoordinator	
Realization SRS20.5 to TestDefinitionCoordinator Source -> Destination	Public SRS20.5	Public TestDefinitionCoordinator	

Connector	Source	Target	Notes
Realization SRS20.2 to TestDefinitionCoordinator Source -> Destination	Public SRS20.2	Public TestDefinitionCoordinator	
Realization SRS12.1 to TestDefinitionCoordinator Source -> Destination	Public SRS12.1	Public TestDefinitionCoordinator	
Realization SRS63 to TestDefinitionCoordinator Source -> Destination	Public SRS63	Public TestDefinitionCoordinator	
Realization SRS20 to TestDefinitionCoordinator Source -> Destination	Public SRS20	Public TestDefinitionCoordinator	
Realization SRS12 to TestDefinitionCoordinator Source -> Destination	Public SRS12	Public TestDefinitionCoordinator	
Realization SRS20.4 to TestDefinitionCoordinator Source -> Destination	Public SRS20.4	Public TestDefinitionCoordinator	
Realization SRS21.2 to TestDefinitionCoordinator Source -> Destination	Public SRS21.2	Public TestDefinitionCoordinator	
Realization SRS20.1 to TestDefinitionCoordinator Source -> Destination	Public SRS20.1	Public TestDefinitionCoordinator	
Realization Source -> Destination	Public SRS84.2	Public TestDefinitionCoordinator	
Realization SRS14 to TestDefinitionCoordinator Source -> Destination	Public SRS14	Public TestDefinitionCoordinator	
Realization Source -> Destination	Public SRS84.3	Public TestDefinitionCoordinator	
Realization SRS84.1 to TestDefinitionCoordinator Source -> Destination	Public SRS84.1	Public TestDefinitionCoordinator	
Realization SRS62 to TestDefinitionCoordinator Source -> Destination	Public SRS62	Public TestDefinitionCoordinator	
Realization SRS21 to	Public	Public	

Connector	Source	Target	Notes
TestDefinitionCoordinator Source -> Destination	SRS21	TestDefinitionCoordinator	
Realization SRS19 to TestDefinitionCoordinator Source -> Destination	Public SRS19	Public TestDefinitionCoordinator	
Realization SRS12.2 to TestDefinitionCoordinator Source -> Destination	Public SRS12.2	Public TestDefinitionCoordinator	
Realization SRS13.2 to TestDefinitionCoordinator Source -> Destination	Public SRS13.2	Public TestDefinitionCoordinator	
Realization SRS21.4 to TestDefinitionCoordinator Source -> Destination	Public SRS21.4	Public TestDefinitionCoordinator	
Realization SRS13 to TestDefinitionCoordinator Source -> Destination	Public SRS13	Public TestDefinitionCoordinator	
Realization SRS21.1 to TestDefinitionCoordinator Source -> Destination	Public SRS21.1	Public TestDefinitionCoordinator	

Attributes

Attribute	Notes	Constraints and tags
configFileOpen boolean Private	The config file open.	*Default:* false
CONFIGURATION_FILE_EXTENSION String Protected Static	The configuration file extension.	*Default:* ".ricfg"
corePanel ConfigurationBasePanel Private Static	The core panel.	*Default:*
mainUI C2CMainUI Private	The main ui.	*Default:*
parentObject TestDefinitionListener	The parent object.	*Default:*

Attribute	Notes	Constraints and tags
Private		
RI_DEFINE_STATE_NAME String Protected Static	The ri define state name.	*Default:* "Running"
RI_EXECUTE_STATE_NAME String Protected Static	The ri execute state name.	*Default:* "Paused"
RI_INIT_STATE_NAME String Protected Static	The ri init state name.	*Default:* "Initial"
RI_OPTIONS_STATE_NAME String Protected Static	The ri options state name.	*Default:* "Terminating"
RI_REPORT_STATE_NAME String Protected Static	The ri report state name.	*Default:* "Resuming"
sorter TableRowSorter<NeedsTableModel> Private	The sorter.	*Default:*
sorter2 TableRowSorter<RequirementsTableModel> Private	The sorter2.	*Default:*
sorter3 TableRowSorter<OtherRequirementsTableModel> Private	The sorter3.	*Default:*
standardAppNRTM NRTM Private	The standard app nrtm.	*Default:*

Attribute	Notes	Constraints and tags
standardNRTM NRTM Private	The standard nrtm.	*Default:*
testConfig TestConfiguration Private	The test config.	*Default:*
testDefinitionUIState String Private	The test definition ui state.	*Default:*
validationErrorsFound boolean Private	The validation errors found.	*Default:* false

Operations

Method	Notes	Parameters
actionPerformed() void Public	Handles actions from the TestExecutionPanel.	ActionEvent [in] e the action event received
initAppLayerTestCasesPanel() void Private	Sets the Test Cases Parameters on the Application Layer Standard Parameters Config Panel.	
initConfigPanel() void Private	Sets the Test Config Parameters on the Config Panel.	
initializeUI() boolean Public	Initialize ui. Pre-Conditions: N/A Post-Conditions: N/A @return true, if successful	boolean [in] createNewConfig the create new config
initInfoLayerTestCasesPanel() void Private	Sets the Test Cases Parameters on the Information Layer Standard Parameters Config Panel.	
initParametersPanel() void Private	Inits the parameters panel. Pre-Conditions: N/A Post-Conditions: N/A	JTable [in] standardNeedsTable the standard needs tableJTable [in] standardRequirementsTable the standard requirements

Method	Notes	Parameters
		tableJTable [in] parametersTable the parameters tableDefaultLayerParameters [in] layerParams the layer params
initSUTPanel() void Private	Inits the sut panel. Pre-Conditions: N/A Post-Conditions: N/A	
isValidAppLayerParamsInput() boolean Private	Tests user input on the appLayerParams Panel. @return true, if is valid app layer params input	
isValidConfigInput() boolean Private	Tests user input on the Configuration Panel. @return true, if is valid config input	
isValidInfoLayerParamsInput() boolean Private	Tests user input on the infoLayerParams Panel. @return true, if is valid info layer params input	
isValidSUTInput() boolean Private	Tests user input on the SUT Panel. @return true, if is valid sut input	
isValidTestConfig() boolean Private	Tests all areas of Test Configuration user input. @return true, if is valid test config	
saveConfigPanelData() void Private	Stores the Test Config Parameters from the Config Panel.	
saveConfigurationData() void Private	Stores the Test Config Parameters from the Config and SUT Panels.	
saveSUTPanelData() void	Stores the Test Config Parameters from the	

Method	Notes	Parameters
Private	SUT Panel.	
TestDefinitionCoordinator() Public	Instantiates a new test definition coordinator. Pre-Conditions: N/A Post-Conditions: N/A	C2CMainUI [in] parentFrame the parent frameTestDefinitionListener [in] parentObject the parent objectConfigurationBasePanel [in] definitionPanel the definition panel

5.3.6.2.4 FilenameValidator

Type: **Class** *Package:* utilities

Whole combination is means, 1 to 35 characters with any lower or upper case character, digit or special symbol " ", "_" and "-" only.

Connections

Connector	Source	Target	Notes
Realization SRS13.1 to FilenameValidator Source -> Destination	Public SRS13.1	Public FilenameValidator	

Attributes

Attribute	Notes	Constraints and tags
errorList ArrayList<String> Private	The error list.	*Default:* new ArrayList<String>()
FILENAME_PATTERN String Private Static Const	The Constant FILENAME_PATTERN.	*Default:* "^[a-zA-Z0-9 _-]{1,35}$"
matcher Matcher Private	The matcher.	*Default:*
MAX_FILE_LENGTH Integer Private Static	The max file length.	*Default:* 35

Attribute	Notes	Constraints and tags
pattern Pattern Private	The pattern.	*Default:*

Operations

Method	Notes	Parameters
FilenameValidator() Public	Instantiates a new filename validator. Pre-Conditions: N/A Post-Conditions: N/A	
getErrorsEncountered() String Public	Gets the errors encountered. @return the errors encountered	
validate() boolean Public	Validate filename with regular expression. @return true valid filename, false invalid filename	String [in] filename filename for validation

5.3.6.3 Context Diagram

N/A

5.3.6.4 Variability Guide

N/A

5.3.6.5 Rationale

N/A

5.4 Center Model

5.4.1 Entity (devices)

5.4.1.1 Entity Manager

5.4.1.1.1 Primary Presentation

5.4.1.1.2 Element Catalog

5.4.1.1.2.1 Center Active Verification Emulation

5.4.1.1.2.1.1 Primary Presentation

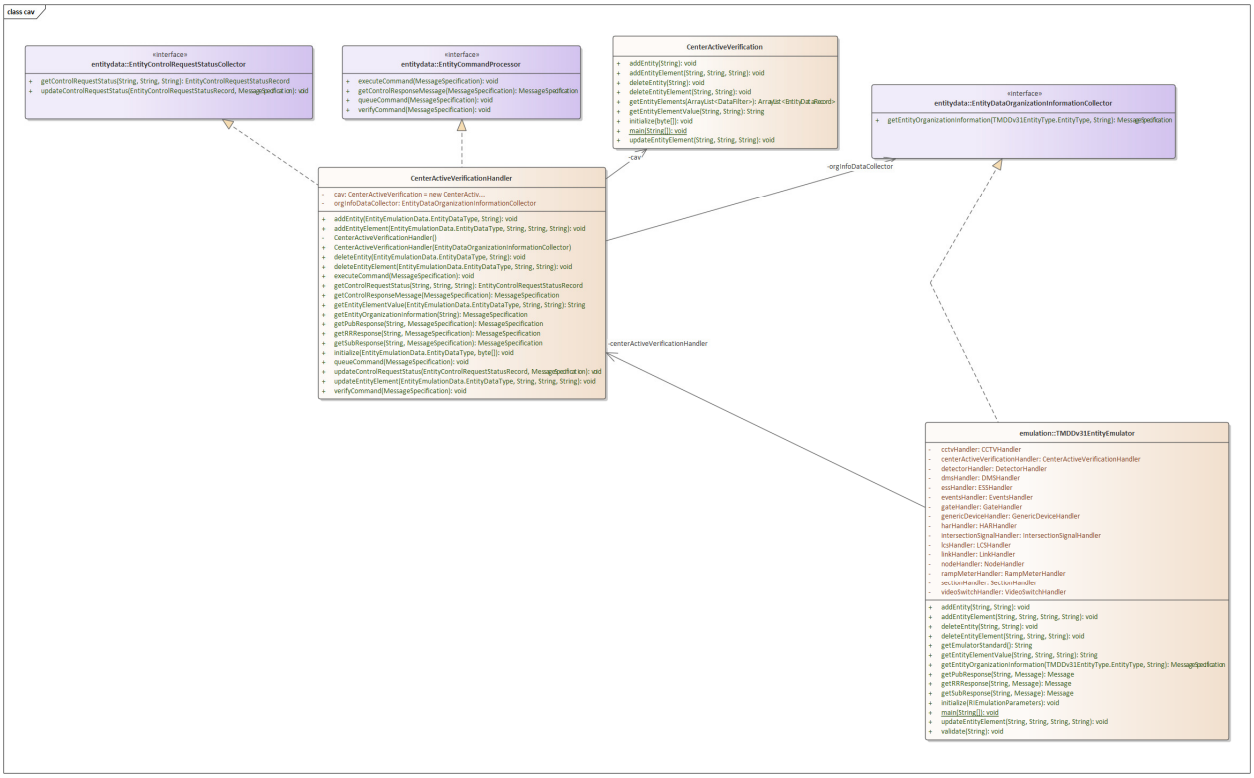

Figure 5-20 Center Active Verification Emulation Modules

Mandatory Need Emulation: Add additional emulation beyond TMDD device entities to include center activation verification, links, and routes.

5.4.1.1.2.1.2 Element Catalog

5.4.1.1.2.1.2.1 CenterActiveVerificationHandler

An implementation of the AbstractEntityHandler that processes requests and provides responses for all CenterActiveVerification entities.

5.4.1.1.2.1.2.2 CenterActiveVerification

This class represents the set of Center Active Verification information for the center. The information that is maintained is described in TMDD Volume II - Section 3.2.3.2. It includes operations to allow the addition, deletion and update of elements within the Center Active Verification Information.

5.4.1.1.2.1.2.3 TMDDv31EntityEmulator

This module implements the RIEmulator interface which is required for all C2C RI Emulation plugins. All C2C RI Test Suites implement this common interface for the retrievel and editing of emulation data as well as the processing of control, response and publication messages.

5.4.1.1.2.1.2.4 EntityControlRequestStatusCollector

This interface defines operations for collecting control request status information. As part of the emulation design it is referenced, however none of the operations are required since there are no applicable commands associated with this entity data.

5.4.1.1.2.1.2.5 EntityCommandProcessor

This interface defines operations for collecting control request status information. As part of the emulation design it is referenced, however none of the operations are required since there are no applicable commands associated with this entity data.

5.4.1.1.2.1.2.6 EntityDataOrganizationCollector

This interface defines operations for collecting organization information header information.

5.4.1.1.2.1.3 Context Diagram

N/A

5.4.1.1.2.1.4 Variability Guide

This module has a unique implementation for each TMDD test suite version. This document only reference the v3.1 instance.

5.4.1.1.2.1.5 Rationale

The design of this module is consistent with the design pattern used for other entities within the C2C RI. Additional Entity Emulation detail processing information is provided within Section 6.2.2 of this document.

5.4.1.1.2.2 Node Emulation

5.4.1.1.2.2.1 Primary Presentation

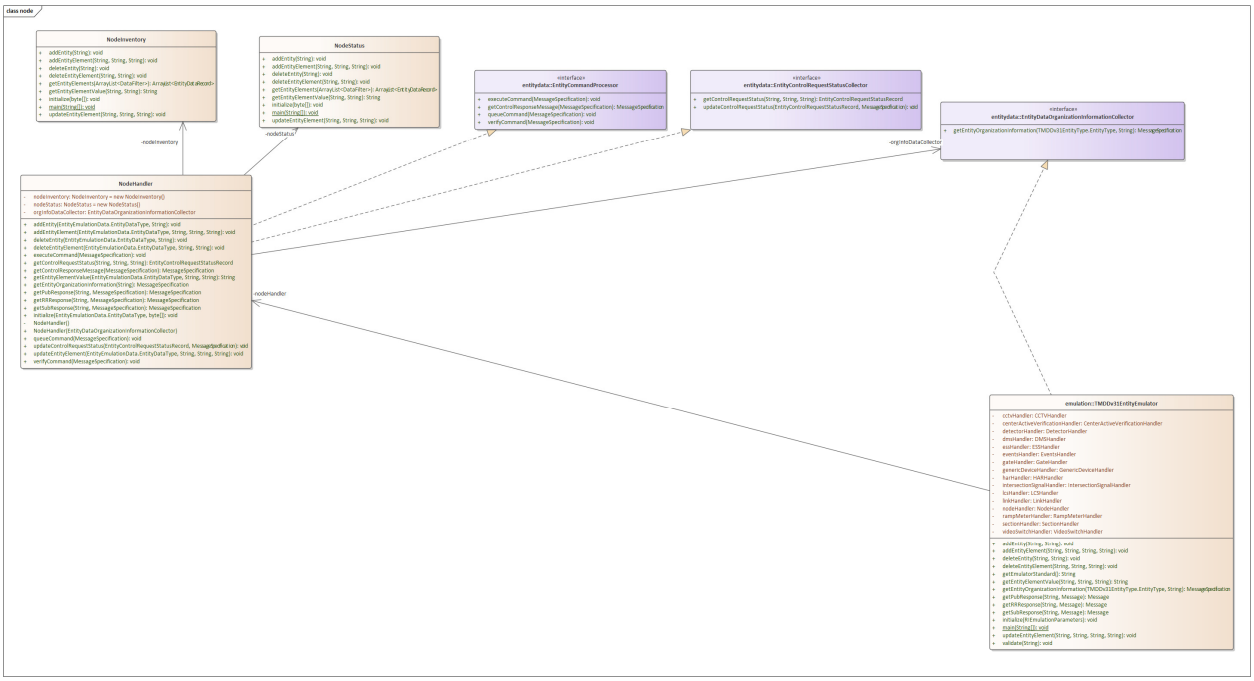

Figure 5-21 Node Emulation Modules

5.4.1.1.2.2.2 Element Catalog

5.4.1.1.2.2.2.1 NodeHandler

An implementation of the AbstractEntityHandler that processes requests and provides responses for all Node entities.

5.4.1.1.2.2.2.2 NodeInventory

This class represents the set of Node Inventory information for the center. The information that is maintained is described in TMDD Volume II - Section 3.2.14.1. It includes operations to allow the addition, deletion and update of elements within the Node Inventory.

5.4.1.1.2.2.2.3 NodeStatus

This class represents the set of Node Status information for the center. The information that is maintained is described in TMDD Volume II - Section 3.2.14.2. It includes operations to allow the addition, deletion and update of elements within the Node Status.

5.4.1.1.2.2.2.4 TMDDv31EntityEmulator

This module implements the RIEmulator interface which is required for all C2C RI Emulation plugins. All C2C RI Test Suites implement this common interface for the retrievel and editing of emulation data as well as the processing of control, response and publication messages.

5.4.1.1.2.2.2.5 EntityControlRequestStatusCollector

This interface defines operations for collecting control request status information. As part of the emulation design it is referenced, however none of the operations are required since there are no applicable commands associated with this entity data.

5.4.1.1.2.2.2.6 EntityCommandProcessor

This interface defines operations for collecting control request status information. As part of the emulation design it is referenced, however none of the operations are required since there are no applicable commands associated with this entity data.

5.4.1.1.2.2.2.7 EntityDataOrganizationCollector

This interface defines operations for collecting organization information header information.

5.4.1.1.2.2.3 Context Diagram

N/A

5.4.1.1.2.2.4 Variability Guide

This module has a unique implementation for each TMDD test suite version. This document only reference the v3.1 instance.

5.4.1.1.2.2.5 Rationale

The design of this module is consistent with the design pattern used for other entities within the C2C RI. Additional Entity Emulation detail processing information is provided within Section 6.2.2 of this document.

5.4.1.1.2.3 Link Emulation

5.4.1.1.2.3.1 Primary Presentation

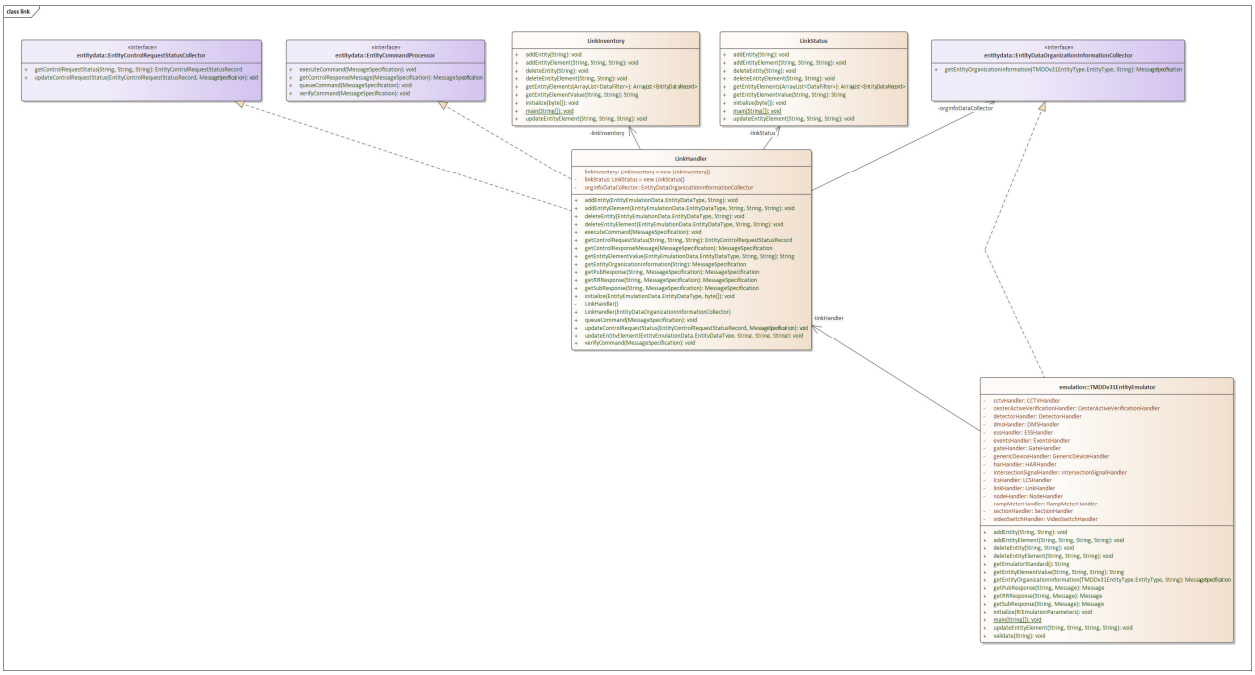

Figure 5-22 Link Emulation Modules

5.4.1.1.2.3.2 Element Catalog

5.4.1.1.2.3.2.1 LinkHandler

An implementation of the AbstractEntityHandler that processes requests and provides responses for all Link entities.

5.4.1.1.2.3.2.2 LinkInventory

This class represents the set of Link Inventory information for the center. The information that is maintained is described in TMDD Volume II - Section 3.2.13.1. It includes operations to allow the addition, deletion and update of elements within the Link Inventory.

5.4.1.1.2.3.2.3 LinkStatus

This class represents the set of Link status information for the center. The information that is maintained is described in TMDD Volume II - Section 3.2.13.2. It includes operations to allow the addition, deletion and update of elements within the Link Status.

5.4.1.1.2.3.2.4 TMDDv31EntityEmulator

This module implements the RIEmulator interface which is required for all C2C RI Emulation plugins. All C2C RI Test Suites implement this common interface for the retrievel and editing of emulation data as well as the processing of control, response and publication messages.

5.4.1.1.2.3.2.5 EntityControlRequestStatusCollector

This interface defines operations for collecting control request status information. As part of the emulation design it is referenced, however none of the operations are required since there are no applicable commands associated with this entity data.

5.4.1.1.2.3.2.6 EntityCommandProcessor

This interface defines operations for collecting control request status information. As part of the emulation design it is referenced, however none of the operations are required since there are no applicable commands associated with this entity data.

5.4.1.1.2.3.2.7 EntityDataOrganizationCollector

This interface defines operations for collecting organization information header information.

5.4.1.1.2.3.3 Context Diagram

N/A

5.4.1.1.2.3.4 Variability Guide

This module has a unique implementation for each TMDD test suite version. This document only reference the v3.1 instance.

5.4.1.1.2.3.5 Rationale

The design of this module is consistent with the design pattern used for other entities within the C2C RI. Additional Entity Emulation detail processing information is provided within Section 6.2.2 of this document.

5.4.1.1.2.4 CommandQueue

Type: **Class** *Package:* centermodel

This class represents the command queue for entities that receive control command requests (HAR, DMS, Section, etc).

Attributes

Attribute	Notes	Constraints and tags
queueLength int Private	The configured length of the command queue. Once the elements of the queue for a type of entity exceeds this number, the command is processed. The maximum queue length will be set at 100.	*Default:* 0

Operations

Method	Notes	Parameters
addCommand() void Public	This method is used by handlers to add commands to the command queue.	Sting [in] entityType
clearCommandRequest() void Public	Indicate to the command queue that the given command should be cleared. Valid reasons are cancelled and completed.	String [in] entityType Command [in] commandRequest String [in] reason
getEntityCommands() CommandList Public	This method returns a list of the active commands for the given entity type.	String [in] entityType
getEntityCommandStatus() CommandStatus Public	This method returns the status of a command request matching the given entity type and request id.	String [in] entityType String [in] requestId
getNextCommandRequest() Command Public	Returns the first unprocessed command from the command queue for the given entity type.	String [in] entityType
initialize() Public	Initializes (clears) the command queues and sets the queue length to the value specified.	int [in] commandQueueLength

Method	Notes	Parameters
	The method will not allow the queue length to be set above it's maximum value.	

5.4.1.1.2.5 RIEmulation

Type: **Class** *Package:* centermodel

When the RI is running a Test, and Owner Center is the Center Mode specified in the Test Configuration, this class will oversee the performance and setup of RI Entity Emulation actions specified in the test scripts for the entities specified in the currently adopted version of the TMDD.

This class acts as the main interface for a service which provides updated (and sometimes simulated) entity information.

This class provides methods which allow for external actions on entity information. All entities support all of the actions.

Connections

Connector	Source	Target	Notes
Association	Public	Private updateService	
Source -> Destination	RIEmulation	EntityDataUpdateServic e	
Sequence	Public	Public	
Source -> Destination	RIEmulation	RequestResponseOCRe plyTag	
Realization	Public	Public	
Source -> Destination	SRS127.2	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS114.2.1	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS140.5	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS120.1	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS140.4	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS137.6	RIEmulation	
Realization	Public	Public	

Connector	Source	Target	Notes
Source -> Destination	SRS109.5	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS130	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS114.3	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS146.3.1	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS142.2	RIEmulation	
Realization	Public	Public	
SRS40.1 to RIEmulation Source -> Destination	SRS40.1	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS115.5	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS112.5	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS124.3	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS147	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS125.1	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS115.3.1	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS109.4	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS148	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS131.8	RIEmulation	
Realization	Public	Public	
SRS41.3 to RIEmulation Source -> Destination	SRS41.3	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS112.3	RIEmulation	
Realization	Public	Public	

Connector	Source	Target	Notes
Source -> Destination	SRS109	RIEmulation	
Realization SRS40.4 to RIEmulation	Public	Public	
Source -> Destination	SRS40.4	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS119.1	RIEmulation	
Realization SRS40.2 to RIEmulation	Public	Public	
Source -> Destination	SRS40.2	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS114.3.1	RIEmulation	
Realization SRS40.7 to RIEmulation	Public	Public	
Source -> Destination	SRS40.7	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS128.5	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS122.4	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS136.1.1	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS130.3	RIEmulation	
Sequence getRRResponse(dialog, requestMessage)	Public	Public	
Source -> Destination	RequestResponseOCReplyTag	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS126.1	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS98	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS137	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS130.2.1	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS117.5	RIEmulation	
Realization	Public	Public	

Connector	Source	Target	Notes
Source -> Destination	SRS128.6	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS139.4	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS142	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS121.1.1	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS132.1	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS129.1	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS131.7	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS128.2	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS113.1	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS131.5	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS137.3	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS111.1	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS120.2	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS122.1	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS119.2	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS137.8	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS40.11	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS133.4.1	RIEmulation	
Realization	Public	Public	

Connector	Source	Target	Notes
Source -> Destination	SRS129	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS136	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS136.1	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS122.3	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS137.3.1	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS137.7	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS115.7	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS117.1	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS125.5	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS113	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS114.1.1	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS125.4	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS127.2.1	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS97	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS137.4	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS125.6	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS146.3	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS131	RIEmulation	
Realization	Public	Public	

Connector	Source	Target	Notes
Source -> Destination	SRS127.4.1	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS141.1	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS124.2	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS114	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS111	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS119	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS116	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS112.4	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS124.3.1	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS128	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS118.1	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS139.2	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS115.9	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS140	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS136.3.1	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS118.1.1	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS131.4	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS133.3.1	RIEmulation	
Realization	Public	Public	

Connector	Source	Target	Notes
Source -> Destination	SRS121.1	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS115.3	RIEmulation	
Association subscription requests Unspecified	Public SubPubController	Public RIEmulation	The SubPubController references RI Emulation in order to confirm subscription requests.
Realization	Public	Public	
Source -> Destination	SRS133.3	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS122	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS134.2	RIEmulation	
Realization SRS40.9 to RIEmulation Source -> Destination	Public SRS40.9	Public RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS125	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS118.3.1	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS115	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS113.2	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS118.3	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS116.6	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS125.3	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS131.3.1	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS142.1.1	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS114.2	RIEmulation	
Realization	Public	Public	

Connector	Source	Target	Notes
Source -> Destination	SRS133.1	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS124.1.1	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS136.2	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS133.2	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS128.1	RIEmulation	
Realization SRS41.1 to RIEmulation	Public	Public	
Source -> Destination	SRS41.1	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS122.3.1	RIEmulation	
NoteLink	Public	Public	
Source -> Destination	<anonymous>	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS139.1	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS146.1.1	RIEmulation	
Realization SRS41.2 to RIEmulation	Public	Public	
Source -> Destination	SRS41.2	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS134.4	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS133	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS140.1	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS146.2	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS135	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS40.10	RIEmulation	
Realization	Public	Public	

Connector	Source	Target	Notes
Source -> Destination	SRS136.3	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS125.8	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS124.1	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS125.3.1	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS127.1	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS121.2	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS139.4.1	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS147.1	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS117.3	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS109.2	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS146.1	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS140.2	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS116.1	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS133.1.1	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS127.3	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS116.2	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS137.1	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS115.4	RIEmulation	
Realization	Public	Public	

Connector	Source	Target	Notes
Source -> Destination	SRS134.7	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS111.1.1	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS134.3	RIEmulation	
Realization	Public	Public	
SRS41.4 to RIEmulation	SRS41.4	RIEmulation	
Source -> Destination			
Realization	Public	Public	
Source -> Destination	SRS115.2	RIEmulation	
Realization	Public	Public	
SRS41.6 to RIEmulation	SRS41.6	RIEmulation	
Source -> Destination			
Realization	Public	Public	
Source -> Destination	SRS115.8	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS112.1	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS127.3.1	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS111.2.1	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS134	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS136.2.1	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS133.4	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS109.3	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS130.2	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS130.1	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS112.2	RIEmulation	
Realization	Public	Public	

Connector	Source	Target	Notes
Source -> Destination	SRS112	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS134.2.1	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS118.2	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS128.4	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS135.1	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS126	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS123.1	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS109.1	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS131.6	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS140.3.1	RIEmulation	
Realization SRS41.7 to RIEmulation	Public	Public	
Source -> Destination	SRS41.7	RIEmulation	
Realization SRS40.6 to RIEmulation	Public	Public	
Source -> Destination	SRS40.6	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS127	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS131.1	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS145	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS144	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS116.5	RIEmulation	
Realization SRS41.5 to RIEmulation	Public	Public	

Connector	Source	Target	Notes
Source -> Destination	SRS41.5	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS118.2.1	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS141	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS115.6	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS138	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS131.3	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS130.1.1	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS124.2.1	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS122.2	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS115.1	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS130.3.1	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS117.6	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS117	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS142.1	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS121.3.1	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS134.6	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS116.4	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS128.3	RIEmulation	
Realization	Public	Public	

Connector	Source	Target	Notes
Source -> Destination	SRS139.3.1	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS121.2.1	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS128.3.1	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS146.2.1	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS122.5	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS121	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS117.4	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS139.1.1	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS116.3	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS114.1	RIEmulation	
Realization SRS40.8 to RIEmulation	Public	Public	
Source -> Destination	SRS40.8	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS121.3	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS124	RIEmulation	
Realization SRS40.3 to RIEmulation	Public	Public	
Source -> Destination	SRS40.3	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS125.7	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS93	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS122.6	RIEmulation	
Realization	Public	Public	

Connector	Source	Target	Notes
Source -> Destination	SRS137.5	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS139	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS146	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS123	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS139.3	RIEmulation	
Realization SRS40.5 to RIEmulation	Public	Public	
Source -> Destination	SRS40.5	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS138.2	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS120.3	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS109.1.1	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS139.2.1	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS134.1	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS132	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS140.3	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS131.2	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS127.1.1	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS120	RIEmulation	
Association Publication Update Requests	Public	Public	The Publication references RI Emulation in order to obtain publication messages for distribution.
Unspecified	Publication	RIEmulation	

Connector	Source	Target	Notes
Realization	Public	Public	
Source -> Destination	SRS127.4	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS40.12	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS111.2	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS117.2	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS125.2	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS134.5	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS138.1	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS135.2	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS133.2.1	RIEmulation	
Realization SRS41 to RIEmulation	Public	Public	
Source -> Destination	SRS41	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS119.3	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS142.2.1	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS132.2	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS137.2	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS99	RIEmulation	
Realization SRS40 to RIEmulation	Public	Public	
Source -> Destination	SRS40	RIEmulation	
Realization	Public	Public	
Source -> Destination	SRS118	RIEmulation	

Attributes

Attribute	Notes	Constraints and tags
updateService EnityDataUpdateService Private	The entity data update service associated with Entity Emulation.	*Default:*

Operations

Method	Notes	Parameters
addEntity() void Public	Method provides a means for external addition of entities.	String [in] entityDataType String [in] entityId
addEntityElement() void Public	Method provides a means for external addition of elements of an entity.	String [in] entityDataType String [in] entityId String [in] entityName String [in] entityValue
deleteEntityElement() void Public	Method provides a means for external deletion of an entity.	String [in] entityDataType String [in] entityId String [in] entityName String [in] entityValue
getEntityElement() String Public	Method provides a means for external view of an entity element.	String [in] entityDataType String [in] entityId String [in] entityName String [in] entityValue
getPubMessage() void Public	This method returns the publication message associated with the subscription message that was received and the dialog name provided. If the request could not be properly satisfied by the entity emulation, an exception is thrown.	String [in] dialog The name of the publication dialog that is associated with the message provided.Message [in] subscriptionMessage The subscription message that was sent for this publication.
getRRResponse() Message Public	This method returns the response message associated with the request message that was received. If the request could not be properly satisfied by the entity emulation, an error response message is provided.	String [in] dialog The name of the RequestResponse dialog that is associated with the message provided.Message [in] requestMessage

Method	Notes	Parameters
		The request message that was received from the EC.
getSubResponse() Message Public	This method returns the response message associated with the subscription message that was received. If the request could not be properly satisfied by the entity emulation, an error response message is provided.	String [in] dialog The name of the subscription dialog that is associated with the message provided.Message [in] requestMessage The subscription message that was received from the EC.
initialize() void Public	Method used to trigger the initialization of RI Emulation using the emulation data provided.	RIEmulationParameters [in] emulationParameters
registerForUpdates() EntityDataUpdateService Public	Registers the provided entity listener for any updates to data related to the given dialog.	String [in] dialog EntityDataListener [in] entityListener
unRegisterForUpdates() void Public	Untegisters the provided entity listener for updates to data related to the given dialog.	String [in] dialog EntityDataListener [in] entityListener
updateEntityElement() void Public	Method provides a means for external change of an entity element value.	String [in] entityDataType String [in] entityId String [in] entityName String [in] entityValue

5.4.1.1.2.6 CCTVHandler

Type: **Class** *Package:* cctv

An implementation of the AbstractEntityHandler that processes requests and provides responses for all CCTV entities.

Connections

Connector	Source	Target	Notes
Realization	Public	Public	
Source -> Destination	SRS109	CCTVHandler	
Realization	Public	Public	
Source -> Destination	SRS102	CCTVHandler	

Connector	Source	Target	Notes
Realization	Public	Public	
Source -> Destination	SRS109.1.1	CCTVHandler	
Realization	Public	Public	
Source -> Destination	SRS111.1.1	CCTVHandler	
Realization	Public	Public	
Source -> Destination	SRS109.3	CCTVHandler	
Realization	Public	Public	
Source -> Destination	SRS101	CCTVHandler	
Realization	Public	Public	
Source -> Destination	SRS106	CCTVHandler	
Realization	Public	Public	
Source -> Destination	SRS105	CCTVHandler	
Realization	Public	Public	
Source -> Destination	SRS103	CCTVHandler	
Realization	Public	Public	
Source -> Destination	SRS108	CCTVHandler	
Realization	Public	Public	
Source -> Destination	SRS109.4	CCTVHandler	
Realization	Public	Public	
Source -> Destination	SRS111.1	CCTVHandler	
Realization	Public	Public	
Source -> Destination	SRS110	CCTVHandler	
Realization	Public	Public	
Source -> Destination	SRS109.2	CCTVHandler	
Realization	Public	Public	
Source -> Destination	SRS110.1	CCTVHandler	
Realization	Public	Public	
Source -> Destination	SRS111.2	CCTVHandler	
Realization	Public	Public	
Source -> Destination	SRS109.1	CCTVHandler	
Realization	Public	Public	
Source -> Destination	SRS107	CCTVHandler	
Realization	Public	Public	
Source -> Destination	SRS98	CCTVHandler	
Realization	Public	Public	
Source -> Destination	SRS111	CCTVHandler	

Connector	Source	Target	Notes
Realization	Public	Public	
Source -> Destination	SRS109.5	CCTVHandler	
Realization	Public	Public	
Source -> Destination	SRS111.2.1	CCTVHandler	

Operations

Method	Notes	Parameters
getPubMessage() void Public	When a dlCCTVInventoryUpdate dialog and valid subscription message is received, this method will return either a cCTVInventoryMsg message, or an exception if an error was encountered. When a dlCCTVStatusUpdate dialog and valid subscription message is received, this method will return either a cCTVStatusMsg message, or an exception if an error was encountered.	String [in] dialog The name of the publication dialog that is associated with the message provided.Message [in] subscriptionMessage The subscription message that was sent for this publication.
getRRResponse() Message Public	When a dlCCTVControlRequest dialog and valid cCTVControlRequestMsg message is received, this method will return either a deviceControlResponse message, or an errorReportMsg message if an error was encountered. When a dlCCTVInventoryRequest dialog and valid deviceInformationRequestMsg message is received, this method will return either a cCTVInventory message, or an errorReportMsg message if an error was encountered. When a dlCCTVStatusRequest dialog and valid deviceInformationRequestMsg message is received, this method will return either a cCTVStatus message, or an errorReportMsg message if an error was encountered. When a dlDeviceCancelControlRequest dialog and valid deviceCancelControlRequestMsg message is received, this method will return either a deviceControlResponse message, or an errorReportMsg message if an error was encountered. When a dlDeviceControlStatusRequest dialog and valid deviceControlStatusRequestMsg message is	String [in] dialog The name of the RequestResponse dialog that is associated with the message provided.Message [in] requestMessage The request message that was received from the EC.

Method	Notes	Parameters
	received, this method will return either a deviceControlResponse message, or an errorReportMsg message if an error was encountered.	
getSubResponse() Message Public	When a dlDeviceInformationSubscription dialog and valid deviceInformationRequestMsg message is received, this method will return either a c2cMessageReceipt message, or an errorReportMsg message if an error was encountered.	String [in] dialog The name of the subscription dialog that is associated with the message provided.Message [in] requestMessage The subscription message that was received from the EC.
Initialize() void Public	A concrete implementation of this method should be called when there is a need to initialize (re-initialize) the data associated with this entity type.	

5.4.1.1.2.7 DetectorHandler

Type: **Class** *Package:* detector

An implementation of the AbstractEntityHandler that processes requests and provides responses for all Detector entities.

Connections

Connector	Source	Target	Notes
Realization	Public	Public	
Source -> Destination	SRS113.2	DetectorHandler	
Realization	Public	Public	
Source -> Destination	SRS107	DetectorHandler	
Realization	Public	Public	
Source -> Destination	SRS98	DetectorHandler	
Realization	Public	Public	
Source -> Destination	SRS108	DetectorHandler	
Realization	Public	Public	
Source -> Destination	SRS114.2.1	DetectorHandler	
Realization	Public	Public	
Source -> Destination	SRS114	DetectorHandler	

Connector	Source	Target	Notes
Realization	Public	Public	
Source -> Destination	SRS102	DetectorHandler	
Realization	Public	Public	
Source -> Destination	SRS101	DetectorHandler	
Realization	Public	Public	
Source -> Destination	SRS114.3	DetectorHandler	
Realization	Public	Public	
Source -> Destination	SRS114.1.1	DetectorHandler	
Realization	Public	Public	
Source -> Destination	SRS113.1	DetectorHandler	
Realization	Public	Public	
Source -> Destination	SRS112.2	DetectorHandler	
Realization	Public	Public	
Source -> Destination	SRS114.3.1	DetectorHandler	
Realization	Public	Public	
Source -> Destination	SRS114.1	DetectorHandler	
Realization	Public	Public	
Source -> Destination	SRS114.2	DetectorHandler	
Realization	Public	Public	
Source -> Destination	SRS112.3	DetectorHandler	
Realization	Public	Public	
Source -> Destination	SRS113	DetectorHandler	
Realization	Public	Public	
Source -> Destination	SRS112.5	DetectorHandler	
Realization	Public	Public	
Source -> Destination	SRS112.4	DetectorHandler	
Realization	Public	Public	
Source -> Destination	SRS112	DetectorHandler	
Realization	Public	Public	
Source -> Destination	SRS112.1	DetectorHandler	

Operations

Method	Notes	Parameters
getPubMessage() void Public	When a dlDetectorDataUpdate dialog and valid subscription message is received, this method will return either a detectorDataMsg message, or an exception if an error was encountered.	String [in] dialog The name of the publication dialog that is associated with the message provided.Message [in]

Method	Notes	Parameters
	When a dlDetectorInventoryUpdate dialog and valid subscription message is received, this method will return either a detectorInventoryMsg message, or an exception if an error was encountered. When a dlDetectorStatusUpdate dialog and valid subscription message is received, this method will return either a detectorStatusMsg message, or an exception if an error was encountered.	subscriptionMessage The subscription message that was sent for this publication.
getRRResponse() Message Public	When a dlArchivedDataProcessingDocumentationMetadataRequest dialog and valid archivedDataProcessingDocumentationMetadataRequestMsg message is received, this method will return either a archivedDataProcessingDocumentationMetadata message, or an errorReportMsg message if an error was encountered. When a dlDetectorDataRequest dialog and valid detectorDataRequestMsg message is received, this method will return either a detectorData message, or an errorReportMsg message if an error was encountered. When a dlDetectorInventoryRequest dialog and valid deviceInformationRequestMsg message is received, this method will return either a detectorInventory message, or an errorReportMsg message if an error was encountered. When a dlDetectorMaintenanceHistoryRequest dialog and valid detectorMaintenanceHistoryRequestMsg message is received, this method will return either a detectorMaintenanceHistory message, or an errorReportMsg message if an error was encountered. When a dlDetectorStatusRequest dialog and valid deviceInformationRequestMsg message is received, this method will return either a detectorStatusMsg message, or an errorReportMsg message if an error was encountered.	String [in] dialog The name of the RequestResponse dialog that is associated with the message provided.Message [in] requestMessage The request message that was received from the EC.

Method	Notes	Parameters
getSubResponse() Message Public	When a dlDetectorDataSubscription dialog and valid deviceInformationRequestMsg message is received, this method will return either a c2cMessageReceipt message, or an errorReportMsg message if an error was encountered. When a dlDeviceInformationSubscription dialog and valid deviceInformationRequestMsg message is received, this method will return either a c2cMessageReceipt message, or an errorReportMsg message if an error was encountered.	String [in] dialog The name of the subscription dialog that is associated with the message provided.Message [in] requestMessage The subscription message that was received from the EC.
Initialize() void Public	A concrete implementation of this method should be called when there is a need to initialize (re-initialize) the data associated with this entity type.	

5.4.1.1.2.8 DMSHandler

Type: **Class** *Package:* dms

An implementation of the AbstractEntityHandler that processes requests and provides responses for all DMS entities.

Connections

Connector	Source	Target	Notes
Realization	Public	Public	
Source -> Destination	SRS115.1	DMSHandler	
Realization	Public	Public	
Source -> Destination	SRS116.2	DMSHandler	
Realization	Public	Public	
Source -> Destination	SRS146.3.1	DMSHandler	
Realization	Public	Public	
Source -> Destination	SRS101	DMSHandler	
Realization	Public	Public	
Source -> Destination	SRS115.3	DMSHandler	
Realization	Public	Public	
Source -> Destination	SRS116.1	DMSHandler	

Connector	Source	Target	Notes
Realization	Public	Public	
Source -> Destination	SRS116.4	DMSHandler	
Realization	Public	Public	
Source -> Destination	SRS98	DMSHandler	
Realization	Public	Public	
Source -> Destination	SRS116.6	DMSHandler	
Realization	Public	Public	
Source -> Destination	SRS115.6	DMSHandler	
Realization	Public	Public	
Source -> Destination	SRS146.2.1	DMSHandler	
Realization	Public	Public	
Source -> Destination	SRS115.5	DMSHandler	
Realization	Public	Public	
Source -> Destination	SRS102	DMSHandler	
Realization	Public	Public	
Source -> Destination	SRS103	DMSHandler	
Realization	Public	Public	
Source -> Destination	SRS105	DMSHandler	
Realization	Public	Public	
Source -> Destination	SRS115.8	DMSHandler	
Realization	Public	Public	
Source -> Destination	SRS115.9	DMSHandler	
Realization	Public	Public	
Source -> Destination	SRS108	DMSHandler	
Realization	Public	Public	
Source -> Destination	SRS106	DMSHandler	
Realization	Public	Public	
Source -> Destination	SRS116	DMSHandler	
Realization	Public	Public	
Source -> Destination	SRS146.1	DMSHandler	
Realization	Public	Public	
Source -> Destination	SRS146	DMSHandler	
Realization	Public	Public	
Source -> Destination	SRS115.2	DMSHandler	
Realization	Public	Public	
Source -> Destination	SRS115	DMSHandler	

Connector	Source	Target	Notes
Realization	Public	Public	
Source -> Destination	SRS115.7	DMSHandler	
Realization	Public	Public	
Source -> Destination	SRS146.1.1	DMSHandler	
Realization	Public	Public	
Source -> Destination	SRS115.4	DMSHandler	
Realization	Public	Public	
Source -> Destination	SRS146.3	DMSHandler	
Realization	Public	Public	
Source -> Destination	SRS115.3.1	DMSHandler	
Realization	Public	Public	
Source -> Destination	SRS116.5	DMSHandler	
Realization	Public	Public	
Source -> Destination	SRS107	DMSHandler	
Realization	Public	Public	
Source -> Destination	SRS116.3	DMSHandler	
Realization	Public	Public	
Source -> Destination	SRS146.2	DMSHandler	

Operations

Method	Notes	Parameters
getPubMessage() void Public	When a dlDMSInventoryUpdate dialog and valid subscription message is received, this method will return either a dMSInventoryMsg message, or an exception if an error was encountered. When a dlDMSMessageInventoryUpdate dialog and valid subscription message is received, this method will return either a dMSMessageInventoryRequestMsg message, or an exception if an error was encountered. When a dlDMSStatusUpdate dialog and valid subscription message is received, this method will return either a dMSStatusMsg message, or an exception if an error was encountered.	String [in] dialog The name of the publication dialog that is associated with the message provided.Message [in] subscriptionMessage The subscription message that was sent for this publication.
getRRResponse() Message Public	When a dlDeviceCancelControlRequest dialog and valid deviceCancelControlRequestMsg message is	String [in] dialog The name of the RequestResponse

Method	Notes	Parameters
	received, this method will return either a deviceControlResponse message, or an errorReportMsg message if an error was encountered. When a dlDeviceControlStatusRequest dialog and valid deviceControlStatusRequestMsg message is received, this method will return either a deviceControlResponse message, or an errorReportMsg message if an error was encountered. When a dlDMSControlRequest dialog and valid dMSControlRequestMsg message is received, this method will return either a deviceControlResponse message, or an errorReportMsg message if an error was encountered. When a dlDMSFontTableRequest dialog and valid dMSFontTableRequestMsg message is received, this method will return either a dMSFontTable message, or an errorReportMsg message if an error was encountered. When a dlDMSInventoryRequest dialog and valid deviceInformationRequestMsg message is received, this method will return either a dMSInventory message, or an errorReportMsg message if an error was encountered. When a dlDMSMessageAppearanceRequest dialog and valid dMSMessageAppearanceRequestMsg message is received, this method will return either a dMSMessageAppearance message, or an errorReportMsg message if an error was encountered. When a dlDMSMessageInventoryRequest dialog and valid dMSMessageInventoryRequestMsg message is received, this method will return either a dMSMessageInventory message, or an errorReportMsg message if an error was encountered. When a dlDMSPriorityQueueRequest dialog and valid devicePriorityQueueRequestMsg message is received, this method will return either a dMSPriorityQueue message, or an errorReportMsg message if an error was encountered. When a dlDMSStatusRequest dialog and	dialog that is associated with the message provided.Message [in] requestMessage The request message that was received from the EC.

Method	Notes	Parameters
	valid deviceInformationRequestMsg message is received, this method will return either a dMSStatus message, or an errorReportMsg message if an error was encountered.	
getSubResponse() Message Public	When a dlDeviceInformationSubscription dialog and valid deviceInformationRequestMsg message is received, this method will return either a c2cMessageReceipt message, or an errorReportMsg message if an error was encountered. When a dlDMSMessageInventorySubscription dialog and valid dMSMessageInventoryRequestMsg message is received, this method will return either a c2cMessageReceipt message, or an errorReportMsg message if an error was encountered.	String [in] dialog The name of the subscription dialog that is associated with the message provided.Message [in] requestMessage The subscription message that was received from the EC.
Initialize() void Public	A concrete implementation of this method should be called when there is a need to initialize (re-initialize) the data associated with this entity type.	

5.4.1.1.2.9 ESSHandler

Type: **Class** *Package:* ess

An implementation of the AbstractEntityHandler that processes requests and provides responses for all ESS entities.

Connections

Connector	Source	Target	Notes
Realization	Public	Public	
Source -> Destination	SRS118.1.1	ESSHandler	
Realization	Public	Public	
Source -> Destination	SRS147.1	ESSHandler	
Realization	Public	Public	
Source -> Destination	SRS101	ESSHandler	

Connector	Source	Target	Notes
Realization	Public	Public	
Source -> Destination	SRS98	ESSHandler	
Realization	Public	Public	
Source -> Destination	SRS118.2	ESSHandler	
Realization	Public	Public	
Source -> Destination	SRS117.6	ESSHandler	
Realization	Public	Public	
Source -> Destination	SRS108	ESSHandler	
Realization	Public	Public	
Source -> Destination	SRS117.2	ESSHandler	
Realization	Public	Public	
Source -> Destination	SRS117.1	ESSHandler	
Realization	Public	Public	
Source -> Destination	SRS118.3	ESSHandler	
Realization	Public	Public	
Source -> Destination	SRS118	ESSHandler	
Realization	Public	Public	
Source -> Destination	SRS117.4	ESSHandler	
Realization	Public	Public	
Source -> Destination	SRS102	ESSHandler	
Realization	Public	Public	
Source -> Destination	SRS117	ESSHandler	
Realization	Public	Public	
Source -> Destination	SRS107	ESSHandler	
Realization	Public	Public	
Source -> Destination	SRS117.3	ESSHandler	
Realization	Public	Public	
Source -> Destination	SRS118.2.1	ESSHandler	
Realization	Public	Public	
Source -> Destination	SRS118.3.1	ESSHandler	
Realization	Public	Public	
Source -> Destination	SRS118.1	ESSHandler	
Realization	Public	Public	
Source -> Destination	SRS147	ESSHandler	
Realization	Public	Public	
Source -> Destination	SRS117.5	ESSHandler	

Operations

Method	Notes	Parameters
getPubMessage() void Public	When a dlESSInventoryUpdate dialog and valid subscription message is received, this method will return either a eSSInventoryMsg message, or an exception if an error was encountered. When a dlESSObservationReportUpdate dialog and valid subscription message is received, this method will return either a eSSObservationReportMsg message, or an exception if an error was encountered. When a dlESSStatusUpdate dialog and valid subscription message is received, this method will return either a eSSStatusMsg message, or an exception if an error was encountered.	String [in] dialog The name of the publication dialog that is associated with the message provided.Message [in] subscriptionMessage The subscription message that was sent for this publication.
getRRResponse() Message Public	When a dlESSInventoryRequest dialog and valid deviceInformationRequestMsg message is received, this method will return either a eSSInventory message, or an errorReportMsg message if an error was encountered. When a dlESSObservationMetadataRequest dialog and valid deviceInformationRequestMsg message is received, this method will return either a eSSObservationMetadata message, or an errorReportMsg message if an error was encountered. When a dlESSObservationReportRequest dialog and valid deviceInformationRequestMsg message is received, this method will return either a eSSObservationReport message, or an errorReportMsg message if an error was encountered. When a dlESSStatusRequest dialog and valid deviceInformationRequestMsg message is received, this method will return either a eSSStatus message, or an errorReportMsg message if an error was encountered.	String [in] dialog The name of the RequestResponse dialog that is associated with the message provided.Message [in] requestMessage The request message that was received from the EC.
getSubResponse() Message	When a dlDeviceInformationSubscription dialog and valid	String [in] dialog

Method	Notes	Parameters
Public	deviceInformationRequestMsg message is received, this method will return either a c2cMessageReceipt message, or an errorReportMsg message if an error was encountered.	The name of the subscription dialog that is associated with the message provided.Message [in] requestMessage The subscription message that was received from the EC.
Initialize() void Public	A concrete implementation of this method should be called when there is a need to initialize (re-initialize) the data associated with this entity type.	

5.4.1.1.2.10 EventHandler

Type: **Class** *Package:* events

An implementation of the AbstractEntityHandler that processes requests and provides responses for all Event entities.

Connections

Connector	Source	Target	Notes
Realization	Public	Public	
Source -> Destination	SRS119	EventHandler	
Realization	Public	Public	
Source -> Destination	SRS108	EventHandler	
Realization	Public	Public	
Source -> Destination	SRS121.2.1	EventHandler	
Realization	Public	Public	
Source -> Destination	SRS121.1.1	EventHandler	
Realization	Public	Public	
Source -> Destination	SRS121.2	EventHandler	
Realization	Public	Public	
Source -> Destination	SRS120.1	EventHandler	
Realization	Public	Public	
Source -> Destination	SRS102	EventHandler	
Realization	Public	Public	
Source -> Destination	SRS107	EventHandler	
Realization	Public	Public	
Source -> Destination	SRS121.3	EventHandler	

Connector	Source	Target	Notes
Realization	Public	Public	
Source -> Destination	SRS120	EventHandler	
Realization	Public	Public	
Source -> Destination	SRS121.3.1	EventHandler	
Realization	Public	Public	
Source -> Destination	SRS121	EventHandler	
Realization	Public	Public	
Source -> Destination	SRS119.3	EventHandler	
Realization	Public	Public	
Source -> Destination	SRS119.1	EventHandler	
Realization	Public	Public	
Source -> Destination	SRS101	EventHandler	
Realization	Public	Public	
Source -> Destination	SRS121.1	EventHandler	
Realization	Public	Public	
Source -> Destination	SRS120.2	EventHandler	
Realization	Public	Public	
Source -> Destination	SRS119.2	EventHandler	
Realization	Public	Public	
Source -> Destination	SRS120.3	EventHandler	

Operations

Method	Notes	Parameters
getPubMessage() void Public	When a dlActionLogUpdate dialog and valid subscription message is received, this method will return either a actionLogMsg message, or an exception if an error was encountered. When a dlEventIndexUpdate dialog and valid subscription message is received, this method will return either a eventIndexMsg message, or an exception if an error was encountered. When a dlFullEventUpdateUpdate dialog and valid subscription message is received, this method will return either a fEUMsg message, or an exception if an error was encountered.	String [in] dialog The name of the publication dialog that is associated with the message provided.Message [in] subscriptionMessage The subscription message that was sent for this publication.

Method	Notes	Parameters
getRRResponse() Message Public	When a dlActionLogRequest dialog and valid eventRequestMsg message is received, this method will return either a actionLog message, or an errorReportMsg message if an error was encountered. When a dlEventIndexRequest dialog and valid eventRequestMsg message is received, this method will return either a eventIndex message, or an errorReportMsg message if an error was encountered. When a dlFullEventUpdateRequest dialog and valid eventRequestMsg message is received, this method will return either a fEU message, or an errorReportMsg message if an error was encountered.	String [in] dialog The name of the RequestResponse dialog that is associated with the message provided.Message [in] requestMessage The request message that was received from the EC.
getSubResponse() Message Public	When a dlActionLogSubscription dialog and valid eventRequestMsg message is received, this method will return either a c2cMessageReceipt message, or an errorReportMsg message if an error was encountered. When a dlEventIndexSubscription dialog and valid eventRequestMsg message is received, this method will return either a c2cMessageReceipt message, or an errorReportMsg message if an error was encountered. When a dlFullEventUpdateSubscription dialog and valid eventRequestMsg message is received, this method will return either a c2cMessageReceipt message, or an errorReportMsg message if an error was encountered.	String [in] dialog The name of the subscription dialog that is associated with the message provided.Message [in] requestMessage The subscription message that was received from the EC.
Initialize() void Public	A concrete implementation of this method should be called when there is a need to initialize (re-initialize) the data associated with this entity type.	

5.4.1.1.2.11 GateHandler

Type: **Class** *Package:* gate

An implementation of the AbstractEntityHandler that processes requests and provides responses for all Gate entities.

Connections

Connector	Source	Target	Notes
Realization	Public	Public	
Source -> Destination	SRS122.6	GateHandler	
Realization	Public	Public	
Source -> Destination	SRS124	GateHandler	
Realization	Public	Public	
Source -> Destination	SRS124.1.1	GateHandler	
Realization	Public	Public	
Source -> Destination	SRS102	GateHandler	
Realization	Public	Public	
Source -> Destination	SRS106	GateHandler	
Realization	Public	Public	
Source -> Destination	SRS123.1	GateHandler	
Realization	Public	Public	
Source -> Destination	SRS122.2	GateHandler	
Realization	Public	Public	
Source -> Destination	SRS108	GateHandler	
Realization	Public	Public	
Source -> Destination	SRS124.3	GateHandler	
Realization	Public	Public	
Source -> Destination	SRS124.2	GateHandler	
Realization	Public	Public	
Source -> Destination	SRS122	GateHandler	
Realization	Public	Public	
Source -> Destination	SRS122.5	GateHandler	
Realization	Public	Public	
Source -> Destination	SRS103	GateHandler	
Realization	Public	Public	
Source -> Destination	SRS98	GateHandler	
Realization	Public	Public	
Source -> Destination	SRS122.1	GateHandler	
Realization	Public	Public	
Source -> Destination	SRS124.2.1	GateHandler	

Connector	Source	Target	Notes
Realization	Public	Public	
Source -> Destination	SRS105	GateHandler	
Realization	Public	Public	
Source -> Destination	SRS123	GateHandler	
Realization	Public	Public	
Source -> Destination	SRS122.3.1	GateHandler	
Realization	Public	Public	
Source -> Destination	SRS122.3	GateHandler	
Realization	Public	Public	
Source -> Destination	SRS124.1	GateHandler	
Realization	Public	Public	
Source -> Destination	SRS122.4	GateHandler	
Realization	Public	Public	
Source -> Destination	SRS101	GateHandler	
Realization	Public	Public	
Source -> Destination	SRS107	GateHandler	
Realization	Public	Public	
Source -> Destination	SRS124.3.1	GateHandler	

Operations

Method	Notes	Parameters
getPubMessage() void Public	When a dlGateControlScheduleUpdate dialog and valid subscription message is received, this method will return either a gateControlScheduleMsg message, or an exception if an error was encountered. When a dlGateInventoryUpdate dialog and valid subscription message is received, this method will return either a gateInventoryMsg message, or an exception if an error was encountered. When a dlGateStatusUpdate dialog and valid subscription message is received, this method will return either a gateStatusMsg message, or an exception if an error was encountered.	String [in] dialog The name of the publication dialog that is associated with the message provided.Message [in] subscriptionMessage The subscription message that was sent for this publication.
getRRResponse() Message Public	When a dlDeviceCancelControlRequest dialog and valid deviceCancelControlRequestMsg message is received, this method will return either a	String [in] dialog The name of the RequestResponse dialog that is associated with the

Method	Notes	Parameters
	deviceControlResponse message, or an errorReportMsg message if an error was encountered. When a dlDeviceControlStatusRequest dialog and valid deviceControlStatusRequestMsg message is received, this method will return either a deviceControlResponse message, or an errorReportMsg message if an error was encountered. When a dlGateControlRequest dialog and valid gateControlRequestMsg message is received, this method will return either a deviceControlResponse message, or an errorReportMsg message if an error was encountered. When a dlGateControlScheduleRequest dialog and valid deviceInformationRequestMsg message is received, this method will return either a gateControlSchedule message, or an errorReportMsg message if an error was encountered. When a dlGateInventoryRequest dialog and valid deviceInformationRequestMsg message is received, this method will return either a gateInventory message, or an errorReportMsg message if an error was encountered. When a dlGateStatusRequest dialog and valid deviceInformationRequestMsg message is received, this method will return either a gateStatus message, or an errorReportMsg message if an error was encountered.	message provided.Message [in] requestMessage The request message that was received from the EC.
getSubResponse() Message Public	When a dlDeviceInformationSubscription dialog and valid deviceInformationRequestMsg message is received, this method will return either a c2cMessageReceipt message, or an errorReportMsg message if an error was encountered.	String [in] dialog The name of the subscription dialog that is associated with the message provided.Message [in] requestMessage The subscription message that was received from the EC.

Method	Notes	Parameters
Initialize() void Public	A concrete implementation of this method should be called when there is a need to initialize (re-initialize) the data associated with this entity type.	

5.4.1.1.2.12 HARHandler

Type: **Class** *Package:* har

An implementation of the AbstractEntityHandler that processes requests and provides responses for all HAR entities.

Connections

Connector	Source	Target	Notes
Realization	Public	Public	
Source -> Destination	SRS125.1	HARHandler	
Realization	Public	Public	
Source -> Destination	SRS127	HARHandler	
Realization	Public	Public	
Source -> Destination	SRS101	HARHandler	
Realization	Public	Public	
Source -> Destination	SRS125.8	HARHandler	
Realization	Public	Public	
Source -> Destination	SRS108	HARHandler	
Realization	Public	Public	
Source -> Destination	SRS127.4	HARHandler	
Realization	Public	Public	
Source -> Destination	SRS102	HARHandler	
Realization	Public	Public	
Source -> Destination	SRS127.3	HARHandler	
Realization	Public	Public	
Source -> Destination	SRS125.4	HARHandler	
Realization	Public	Public	
Source -> Destination	SRS103	HARHandler	
Realization	Public	Public	
Source -> Destination	SRS106	HARHandler	
Realization	Public	Public	

Connector	Source	Target	Notes
Source -> Destination	SRS127.2	HARHandler	
Realization	Public	Public	
Source -> Destination	SRS105	HARHandler	
Realization	Public	Public	
Source -> Destination	SRS127.4.1	HARHandler	
Realization	Public	Public	
Source -> Destination	SRS127.2.1	HARHandler	
Realization	Public	Public	
Source -> Destination	SRS125.2	HARHandler	
Realization	Public	Public	
Source -> Destination	SRS98	HARHandler	
Realization	Public	Public	
Source -> Destination	SRS127.3.1	HARHandler	
Realization	Public	Public	
Source -> Destination	SRS125.5	HARHandler	
Realization	Public	Public	
Source -> Destination	SRS127.1.1	HARHandler	
Realization	Public	Public	
Source -> Destination	SRS126.1	HARHandler	
Realization	Public	Public	
Source -> Destination	SRS126	HARHandler	
Realization	Public	Public	
Source -> Destination	SRS125.6	HARHandler	
Realization	Public	Public	
Source -> Destination	SRS125.7	HARHandler	
Realization	Public	Public	
Source -> Destination	SRS127.1	HARHandler	
Realization	Public	Public	
Source -> Destination	SRS125	HARHandler	
Realization	Public	Public	
Source -> Destination	SRS107	HARHandler	
Realization	Public	Public	
Source -> Destination	SRS125.3	HARHandler	
Realization	Public	Public	
Source -> Destination	SRS125.3.1	HARHandler	

Operations

Method	Notes	Parameters
getPubMessage() void Public	When a dlHARControlScheduleUpdate dialog and valid subscription message is received, this method will return either a hARControlScheduleMsg message, or an exception if an error was encountered. When a dlHARInventoryUpdate dialog and valid subscription message is received, this method will return either a hARInventoryMsg message, or an exception if an error was encountered. When a dlHARMessageInventoryUpdate dialog and valid subscription message is received, this method will return either a hARMessageInventoryMsg message, or an exception if an error was encountered. When a dlHARStatusUpdate dialog and valid subscription message is received, this method will return either a hARStatusMsg message, or an exception if an error was encountered.	String [in] dialog The name of the publication dialog that is associated with the message provided.Message [in] subscriptionMessage The subscription message that was sent for this publication.
getRRResponse() Message Public	When a dlDeviceCancelControlRequest dialog and valid deviceCancelControlRequestMsg message is received, this method will return either a deviceControlResponse message, or an errorReportMsg message if an error was encountered. When a dlDeviceControlStatusRequest dialog and valid deviceControlStatusRequestMsg message is received, this method will return either a deviceControlResponse message, or an errorReportMsg message if an error was encountered. When a dlHARControlRequest dialog and valid hARControlRequestMsg message is received, this method will return either a deviceControlResponse message, or an errorReportMsg message if an error was encountered. When a dlHARControlScheduleRequest dialog and valid deviceInformationRequestMsg message is received, this method will return either a hARControlSchedule message, or an	String [in] dialog The name of the RequestResponse dialog that is associated with the message provided.Message [in] requestMessage The request message that was received from the EC.

Method	Notes	Parameters
	errorReportMsg message if an error was encountered. When a dlHARInventoryRequest dialog and valid deviceInformationRequestMsg message is received, this method will return either a hARInventory message, or an errorReportMsg message if an error was encountered. When a dlHARMessageInventoryRequest dialog and valid deviceInformationRequestMsg message is received, this method will return either a hARMessageInventory message, or an errorReportMsg message if an error was encountered. When a dlHARPriorityQueueRequest dialog and valid devicePriorityQueueRequestMsg message is received, this method will return either a hARPriorityQueue message, or an errorReportMsg message if an error was encountered. When a dlHARStatusRequest dialog and valid deviceInformationRequestMsg message is received, this method will return either a hARStatus message, or an errorReportMsg message if an error was encountered.	
getSubResponse() Message Public	When a dlDeviceInformationSubscription dialog and valid deviceInformationRequestMsg message is received, this method will return either a c2cMessageReceipt message, or an errorReportMsg message if an error was encountered.	String [in] dialog The name of the subscription dialog that is associated with the message provided.Message [in] requestMessage The subscription message that was received from the EC.
Initialize() void Public	A concrete implementation of this method should be called when there is a need to initialize (re-initialize) the data associated with this entity type.	

5.4.1.1.2.13 LaneControlSignalHandler

Type: **Class** *Package:* lcs

An implementation of the AbstractEntityHandler that processes requests and provides responses for all Lane Control Signal entities.

Connections

Connector	Source	Target	Notes
Realization	Public	Public	
Source -> Destination	SRS103	LaneControlSignalHandler	
Realization	Public	Public	
Source -> Destination	SRS129	LaneControlSignalHandler	
Realization	Public	Public	
Source -> Destination	SRS128.3.1	LaneControlSignalHandler	
Realization	Public	Public	
Source -> Destination	SRS106	LaneControlSignalHandler	
Realization	Public	Public	
Source -> Destination	SRS105	LaneControlSignalHandler	
Realization	Public	Public	
Source -> Destination	SRS128.4	LaneControlSignalHandler	
Realization	Public	Public	
Source -> Destination	SRS98	LaneControlSignalHandler	
Realization	Public	Public	
Source -> Destination	SRS130	LaneControlSignalHandler	
Realization	Public	Public	
Source -> Destination	SRS128.5	LaneControlSignalHandler	
Realization	Public	Public	
Source -> Destination	SRS130.1.1	LaneControlSignalHandler	
Realization	Public	Public	
Source -> Destination	SRS128	LaneControlSignalHandler	

Connector	Source	Target	Notes
Realization	Public	Public	
Source -> Destination	SRS129.1	LaneControlSignalHandler	
Realization	Public	Public	
Source -> Destination	SRS128.2	LaneControlSignalHandler	
Realization	Public	Public	
Source -> Destination	SRS130.3.1	LaneControlSignalHandler	
Realization	Public	Public	
Source -> Destination	SRS130.2	LaneControlSignalHandler	
Realization	Public	Public	
Source -> Destination	SRS128.3	LaneControlSignalHandler	
Realization	Public	Public	
Source -> Destination	SRS102	LaneControlSignalHandler	
Realization	Public	Public	
Source -> Destination	SRS130.1	LaneControlSignalHandler	
Realization	Public	Public	
Source -> Destination	SRS130.3	LaneControlSignalHandler	
Realization	Public	Public	
Source -> Destination	SRS108	LaneControlSignalHandler	
Realization	Public	Public	
Source -> Destination	SRS130.2.1	LaneControlSignalHandler	
Realization	Public	Public	
Source -> Destination	SRS107	LaneControlSignalHandler	
Realization	Public	Public	
Source -> Destination	SRS101	LaneControlSignalHandler	
Realization	Public	Public	
Source -> Destination	SRS128.1	LaneControlSignalHandler	
Realization	Public	Public	

Connector	Source	Target	Notes
Source -> Destination	SRS128.6	LaneControlSignalHandler	

Operations

Method	Notes	Parameters
getPubMessage() void Public	When a dlLCSControlScheduleUpdate dialog and valid subscription message is received, this method will return either a lCSControlScheduleMsg message, or an exception if an error was encountered. When a dlLCSInventoryUpdate dialog and valid subscription message is received, this method will return either a lCSInventoryMsg message, or an exception if an error was encountered. When a dlLCSStatusUpdate dialog and valid subscription message is received, this method will return either a lCSStatusMsg message, or an exception if an error was encountered.	String [in] dialog The name of the publication dialog that is associated with the message provided.Message [in] subscriptionMessage The subscription message that was sent for this publication.
getRRResponse() Message Public	When a dlDeviceCancelControlRequest dialog and valid deviceCancelControlRequestMsg message is received, this method will return either a deviceControlResponse message, or an errorReportMsg message if an error was encountered. When a dlDeviceControlStatusRequest dialog and valid deviceControlStatusRequestMsg message is received, this method will return either a deviceControlResponse message, or an errorReportMsg message if an error was encountered. When a dlLCSControlRequest dialog and valid lCSControlRequestMsg message is received, this method will return either a deviceControlResponse message, or an errorReportMsg message if an error was encountered. When a dlLCSControlScheduleRequest dialog and valid deviceInformationRequestMsg message is received, this method will return either a lCSControlSchedule message, or an errorReportMsg message if an error was encountered.	String [in] dialog The name of the RequestResponse dialog that is associated with the message provided.Message [in] requestMessage The request message that was received from the EC.

Method	Notes	Parameters
	When a dlLCSInventoryRequest dialog and valid deviceInformationRequestMsg message is received, this method will return either a lCSInventory message, or an errorReportMsg message if an error was encountered. When a dlLCSStatusRequest dialog and valid deviceInformationRequestMsg message is received, this method will return either a lCSStatus message, or an errorReportMsg message if an error was encountered.	
getSubResponse() Message Public	When a dlDeviceInformationSubscription dialog and valid deviceInformationRequestMsg message is received, this method will return either a c2cMessageReceipt message, or an errorReportMsg message if an error was encountered.	String [in] dialog The name of the subscription dialog that is associated with the message provided.Message [in] requestMessage The subscription message that was received from the EC.
Initialize() void Public	A concrete implementation of this method should be called when there is a need to initialize (re-initialize) the data associated with this entity type.	

5.4.1.1.2.14 RampMeterHandler

Type: **Class** *Package:* rampmeter

An implementation of the AbstractEntityHandler that processes requests and provides responses for all Ramp Meter entities.

Connections

Connector	Source	Target	Notes
Realization	Public	Public	
Source -> Destination	SRS105	RampMeterHandler	
Realization	Public	Public	
Source -> Destination	SRS131.3	RampMeterHandler	

Connector	Source	Target	Notes
Realization	Public	Public	
Source -> Destination	SRS131.4	RampMeterHandler	
Realization	Public	Public	
Source -> Destination	SRS132.1	RampMeterHandler	
Realization	Public	Public	
Source -> Destination	SRS133.1	RampMeterHandler	
Realization	Public	Public	
Source -> Destination	SRS103	RampMeterHandler	
Realization	Public	Public	
Source -> Destination	SRS133.3.1	RampMeterHandler	
Realization	Public	Public	
Source -> Destination	SRS133.1.1	RampMeterHandler	
Realization	Public	Public	
Source -> Destination	SRS131.2	RampMeterHandler	
Realization	Public	Public	
Source -> Destination	SRS133	RampMeterHandler	
Realization	Public	Public	
Source -> Destination	SRS101	RampMeterHandler	
Realization	Public	Public	
Source -> Destination	SRS133.3	RampMeterHandler	
Realization	Public	Public	
Source -> Destination	SRS98	RampMeterHandler	
Realization	Public	Public	
Source -> Destination	SRS131	RampMeterHandler	
Realization	Public	Public	
Source -> Destination	SRS106	RampMeterHandler	
Realization	Public	Public	
Source -> Destination	SRS133.4.1	RampMeterHandler	
Realization	Public	Public	
Source -> Destination	SRS133.4	RampMeterHandler	
Realization	Public	Public	
Source -> Destination	SRS131.3.1	RampMeterHandler	
Realization	Public	Public	
Source -> Destination	SRS131.6	RampMeterHandler	
Realization	Public	Public	
Source -> Destination	SRS132.2	RampMeterHandler	

Connector	Source	Target	Notes
Realization	Public	Public	
Source -> Destination	SRS131.5	RampMeterHandler	
Realization	Public	Public	
Source -> Destination	SRS108	RampMeterHandler	
Realization	Public	Public	
Source -> Destination	SRS131.8	RampMeterHandler	
Realization	Public	Public	
Source -> Destination	SRS102	RampMeterHandler	
Realization	Public	Public	
Source -> Destination	SRS107	RampMeterHandler	
Realization	Public	Public	
Source -> Destination	SRS132	RampMeterHandler	
Realization	Public	Public	
Source -> Destination	SRS133.2	RampMeterHandler	
Realization	Public	Public	
Source -> Destination	SRS133.2.1	RampMeterHandler	
Realization	Public	Public	
Source -> Destination	SRS131.1	RampMeterHandler	
Realization	Public	Public	
Source -> Destination	SRS131.7	RampMeterHandler	

Operations

Method	Notes	Parameters
getPubMessage() void Public	When a dlRampMeterControlScheduleUpdate dialog and valid subscription message is received, this method will return either a rampMeterControlScheduleMsg message, or an exception if an error was encountered. When a dlRampMeterInventoryUpdate dialog and valid subscription message is received, this method will return either a rampMeterInventoryMsg message, or an exception if an error was encountered. When a dlRampMeterPlanInventoryUpdate dialog and valid subscription message is received, this method will return either a rampMeterPlanInventoryMsg message, or an exception if an error was encountered. When a dlRampMeterStatusUpdate dialog and valid subscription message is received, this method will return either a	String [in] dialog The name of the publication dialog that is associated with the message provided.Message [in] subscriptionMessage The subscription message that was sent for this publication.

Method	Notes	Parameters
	rampMeterStatusMsg message, or an exception if an error was encountered.	
getRRResponse() Message Public	When a dlDeviceCancelControlRequest dialog and valid deviceCancelControlRequestMsg message is received, this method will return either a deviceControlResponse message, or an errorReportMsg message if an error was encountered. When a dlDeviceControlStatusRequest dialog and valid deviceControlStatusRequestMsg message is received, this method will return either a deviceControlResponse message, or an errorReportMsg message if an error was encountered. When a dlRampMeterControlRequest dialog and valid rampMeterControlRequestMsg message is received, this method will return either a deviceControlResponse message, or an errorReportMsg message if an error was encountered. When a dlRampMeterControlScheduleRequest dialog and valid deviceInformationRequestMsg message is received, this method will return either a rampMeterControlSchedule message, or an errorReportMsg message if an error was encountered. When a dlRampMeterInventoryRequest dialog and valid deviceInformationRequestMsg message is received, this method will return either a rampMeterInventory message, or an errorReportMsg message if an error was encountered. When a dlRampMeterPlanInventoryRequest dialog and valid deviceInformationRequestMsg message is received, this method will return either a rampMeterPlanInventory message, or an errorReportMsg message if an error was encountered. When a dlRampMeterPriorityQueueRequest dialog and valid devicePriorityQueueRequestMsg message is received, this method will return either a	String [in] dialog The name of the RequestResponse dialog that is associated with the message provided.Message [in] requestMessage The request message that was received from the EC.

Method	Notes	Parameters
	rampMeterPriorityQueue message, or an errorReportMsg message if an error was encountered. When a dlRampMeterStatusRequest dialog and valid deviceInformationRequestMsg message is received, this method will return either a rampMeterStatus message, or an errorReportMsg message if an error was encountered.	
getSubResponse() Message Public	When a dlDeviceInformationSubscription dialog and valid deviceInformationRequestMsg message is received, this method will return either a message, or an errorReportMsg message if an error was encountered. When a dlRampMeterPlanInventorySubscription dialog and valid deviceInformationRequestMsg message is received, this method will return either a message, or an errorReportMsg message if an error was encountered.	String [in] dialog The name of the subscription dialog that is associated with the message provided.Message [in] requestMessage The subscription message that was received from the EC.
Initialize() void Public	A concrete implementation of this method should be called when there is a need to initialize (re-initialize) the data associated with this entity type.	

5.4.1.1.2.15 SignalSectionHandler

Type: **Class** *Package:* signalsection

An implementation of the AbstractEntityHandler that processes requests and provides responses for all Signal Section entities.

Connections

Connector	Source	Target	Notes
Realization	Public	Public	
Source -> Destination	SRS135.1	SignalSectionHandler	

Connector	Source	Target	Notes
Realization	Public	Public	
Source -> Destination	SRS134.2	SignalSectionHandler	
Realization	Public	Public	
Source -> Destination	SRS136.1	SignalSectionHandler	
Realization	Public	Public	
Source -> Destination	SRS136	SignalSectionHandler	
Realization	Public	Public	
Source -> Destination	SRS107	SignalSectionHandler	
Realization	Public	Public	
Source -> Destination	SRS101	SignalSectionHandler	
Realization	Public	Public	
Source -> Destination	SRS135	SignalSectionHandler	
Realization	Public	Public	
Source -> Destination	SRS135.2	SignalSectionHandler	
Realization	Public	Public	
Source -> Destination	SRS134.7	SignalSectionHandler	
Realization	Public	Public	
Source -> Destination	SRS134	SignalSectionHandler	
Realization	Public	Public	
Source -> Destination	SRS136.1.1	SignalSectionHandler	
Realization	Public	Public	
Source -> Destination	SRS103	SignalSectionHandler	
Realization	Public	Public	
Source -> Destination	SRS136.2	SignalSectionHandler	
Realization	Public	Public	
Source -> Destination	SRS102	SignalSectionHandler	
Realization	Public	Public	
Source -> Destination	SRS134.1	SignalSectionHandler	
Realization	Public	Public	
Source -> Destination	SRS136.3	SignalSectionHandler	
Realization	Public	Public	
Source -> Destination	SRS136.3.1	SignalSectionHandler	
Realization	Public	Public	
Source -> Destination	SRS134.6	SignalSectionHandler	
Realization	Public	Public	
Source -> Destination	SRS105	SignalSectionHandler	

Connector	Source	Target	Notes
Realization	Public	Public	
Source -> Destination	SRS134.4	SignalSectionHandler	
Realization	Public	Public	
Source -> Destination	SRS134.3	SignalSectionHandler	
Realization	Public	Public	
Source -> Destination	SRS136.2.1	SignalSectionHandler	
Realization	Public	Public	
Source -> Destination	SRS134.5	SignalSectionHandler	
Realization	Public	Public	
Source -> Destination	SRS134.2.1	SignalSectionHandler	
Realization	Public	Public	
Source -> Destination	SRS108	SignalSectionHandler	
Realization	Public	Public	
Source -> Destination	SRS106	SignalSectionHandler	

Operations

Method	Notes	Parameters
getPubMessage() void Public	When a dlSectionControlScheduleUpdate dialog and valid subscription message is received, this method will return either a sectionControlScheduleMsg message, or an exception if an error was encountered. When a dlSectionStatusUpdate dialog and valid subscription message is received, this method will return either a sectionStatusMsg message, or an exception if an error was encountered.	String [in] dialog The name of the publication dialog that is associated with the message provided.Message [in] subscriptionMessage The subscription message that was sent for this publication.
getRRResponse() Message Public	When a dlDeviceCancelControlRequest dialog and valid deviceCancelControlRequestMsg message is received, this method will return either a deviceControlResponse message, or an errorReportMsg message if an error was encountered. When a dlSectionControlRequest dialog and valid sectionControlRequestMsg message is received, this method will return either a sectionControlResponse message, or an errorReportMsg message if an error was encountered. When a dlSectionControlScheduleRequest	String [in] dialog The name of the RequestResponse dialog that is associated with the message provided.Message [in] requestMessage The request message that was received from the EC.

Method	Notes	Parameters
	dialog and valid deviceInformationRequestMsg message is received, this method will return either a sectionControlSchedule message, or an errorReportMsg message if an error was encountered. When a dlSectionControlStatusRequest dialog and valid sectionControlStatusRequestMsg message is received, this method will return either a sectionControlResponse message, or an errorReportMsg message if an error was encountered. When a dlSectionPriorityQueueRequest dialog and valid devicePriorityQueueRequestMsg message is received, this method will return either a sectionPriorityQueue message, or an errorReportMsg message if an error was encountered. When a dlSectionSignalTimingPatternInventoryRequest dialog and valid SectionSignalTimingPatternInventoryRequestMsg message is received, this method will return either a SectionSignalTimingPatternInventory message, or an errorReportMsg message if an error was encountered. When a dlSectionStatusRequest dialog and valid deviceInformationRequestMsg message is received, this method will return either a sectionStatus message, or an errorReportMsg message if an error was encountered.	
getSubResponse() Message Public	When a dlDeviceInformationSubscription dialog and valid deviceInformationRequestMsg message is received, this method will return either a message, or an errorReportMsg message if an error was encountered. When a dlSectionSignalTimingPatternInventorySubscription dialog and valid SectionSignalTimingPatternInventoryRequestMsg message is received, this method will return either a message, or an errorReportMsg message if an error was	String [in] dialog The name of the subscription dialog that is associated with the message provided.Message [in] requestMessage The subscription message that was received from the EC.

Method	Notes	Parameters
	encountered.	
Initialize() void Public	A concrete implementation of this method should be called when there is a need to initialize (re-initialize) the data associated with this entity type.	

5.4.1.1.2.16 TrafficControllerHandler

Type: **Class** *Package:* trafficcontroller

An implementation of the AbstractEntityHandler that processes requests and provides responses for all Traffic Controller entities.

Connections

Connector	Source	Target	Notes
Realization	Public	Public	
Source -> Destination	SRS137.2	TrafficControllerHandle r	
Realization	Public	Public	
Source -> Destination	SRS106	TrafficControllerHandle r	
Realization	Public	Public	
Source -> Destination	SRS139	TrafficControllerHandle r	
Realization	Public	Public	
Source -> Destination	SRS139.1	TrafficControllerHandle r	
Realization	Public	Public	
Source -> Destination	SRS137.4	TrafficControllerHandle r	
Realization	Public	Public	
Source -> Destination	SRS139.3.1	TrafficControllerHandle r	
Realization	Public	Public	
Source -> Destination	SRS137.3	TrafficControllerHandle r	
Realization	Public	Public	
Source -> Destination	SRS139.1.1	TrafficControllerHandle	

Connector	Source	Target	Notes
		r	
Realization	Public	Public	
Source -> Destination	SRS98	TrafficControllerHandle r	
Realization	Public	Public	
Source -> Destination	SRS137.8	TrafficControllerHandle r	
Realization	Public	Public	
Source -> Destination	SRS107	TrafficControllerHandle r	
Realization	Public	Public	
Source -> Destination	SRS137.1	TrafficControllerHandle r	
Realization	Public	Public	
Source -> Destination	SRS103	TrafficControllerHandle r	
Realization	Public	Public	
Source -> Destination	SRS137.6	TrafficControllerHandle r	
Realization	Public	Public	
Source -> Destination	SRS139.3	TrafficControllerHandle r	
Realization	Public	Public	
Source -> Destination	SRS137.7	TrafficControllerHandle r	
Realization	Public	Public	
Source -> Destination	SRS138	TrafficControllerHandle r	
Realization	Public	Public	
Source -> Destination	SRS137.5	TrafficControllerHandle r	
Realization	Public	Public	
Source -> Destination	SRS101	TrafficControllerHandle r	
Realization	Public	Public	
Source -> Destination	SRS108	TrafficControllerHandle r	
Realization	Public	Public	
Source -> Destination	SRS138.2	TrafficControllerHandle r	

Connector	Source	Target	Notes
Realization	Public	Public	
Source -> Destination	SRS137.3.1	TrafficControllerHandle r	
Realization	Public	Public	
Source -> Destination	SRS139.4.1	TrafficControllerHandle r	
Realization	Public	Public	
Source -> Destination	SRS139.4	TrafficControllerHandle r	
Realization	Public	Public	
Source -> Destination	SRS105	TrafficControllerHandle r	
Realization	Public	Public	
Source -> Destination	SRS139.2	TrafficControllerHandle r	
Realization	Public	Public	
Source -> Destination	SRS102	TrafficControllerHandle r	
Realization	Public	Public	
Source -> Destination	SRS139.2.1	TrafficControllerHandle r	
Realization	Public	Public	
Source -> Destination	SRS137	TrafficControllerHandle r	
Realization	Public	Public	
Source -> Destination	SRS138.1	TrafficControllerHandle r	

Operations

Method	Notes	Parameters
getPubMessage() void Public	When a dlIntersectionSignalControlScheduleUpdate dialog and valid subscription message is received, this method will return either a intersectionSignalControlScheduleMsg message, or an exception if an error was encountered. When a dlIntersectionSignalInventoryUpdate dialog and valid subscription message is received, this method will return either a intersectionSignalInventoryMsg message, or an exception if an error was encountered.	String [in] dialog The name of the publication dialog that is associated with the message provided.Message [in] subscriptionMessage The subscription message that was sent for this publication.

Method	Notes	Parameters
	When a dlIntersectionSignalStatusUpdate dialog and valid subscription message is received, this method will return either a intersectionSignalStatusMsg message, or an exception if an error was encountered. When a dlIntersectionSignalTimingPatternInventory Update dialog and valid subscription message is received, this method will return either a intersectionSignalTimingPatternInventoryM sg message, or an exception if an error was encountered.	
getRRResponse() Message Public	When a dlDeviceCancelControlRequest dialog and valid deviceCancelControlRequestMsg message is received, this method will return either a deviceControlResponse message, or an errorReportMsg message if an error was encountered. When a dlDeviceControlStatusRequest dialog and valid deviceControlStatusRequestMsg message is received, this method will return either a deviceControlResponse message, or an errorReportMsg message if an error was encountered. When a dlIntersectionSignalControlRequest dialog and valid intersectionSignalControlRequestMsg message is received, this method will return either a intersectionSignalControlResponse message, or an errorReportMsg message if an error was encountered. When a dlIntersectionSignalControlScheduleRequest dialog and valid deviceInformationRequestMsg message is received, this method will return either a intersectionSignalControlSchedule message, or an errorReportMsg message if an error was encountered. When a dlIntersectionSignalInventoryRequest dialog and valid deviceInformationRequestMsg message is received, this method will return either a intersectionSignalInventory message, or an errorReportMsg message if	String [in] dialog The name of the RequestResponse dialog that is associated with the message provided.Message [in] requestMessage The request message that was received from the EC.

Method	Notes	Parameters
	an error was encountered. When a dlIntersectionSignalPriorityQueueRequest dialog and valid devicePriorityQueueRequestMsg message is received, this method will return either a intersectionSignalPriorityQueue message, or an errorReportMsg message if an error was encountered. When a dlIntersectionSignalStatusRequest dialog and valid deviceInformationRequestMsg message is received, this method will return either a intersectionSignalStatus message, or an errorReportMsg message if an error was encountered. When a dlIntersectionSignalTimingPatternInventory Request dialog and valid intersectionSignalTimingPatternInventoryRe questMsg message is received, this method will return either a intersectionSignalTimingPatternInventory message, or an errorReportMsg message if an error was encountered.	
getSubResponse() Message Public	When a dlDeviceInformationSubscription dialog and valid deviceInformationRequestMsg message is received, this method will return either a c2cMessageReceipt message, or an errorReportMsg message if an error was encountered. When a dlIntersectionSignalTimingPatternInventory Subscription dialog and valid intersectionSignalTimingPatternInventoryRe questMsg message is received, this method will return either a c2cMessageReceipt message, or an errorReportMsg message if an error was encountered.	String [in] dialog The name of the subscription dialog that is associated with the message provided.Message [in] requestMessage The subscription message that was received from the EC.
Initialize() void Public	A concrete implementation of this method should be called when there is a need to initialize (re-initialize) the data associated with this entity type.	

5.4.1.1.2.17 VideoSwitchHandler

Type: **Class** *Package:* videoswitch

An implementation of the AbstractEntityHandler that processes requests and provides responses for all Video Switch entities.

Connections

Connector	Source	Target	Notes
Realization	Public	Public	
Source -> Destination	SRS106	VideoSwitchHandler	
Realization	Public	Public	
Source -> Destination	SRS141	VideoSwitchHandler	
Realization	Public	Public	
Source -> Destination	SRS103	VideoSwitchHandler	
Realization	Public	Public	
Source -> Destination	SRS140.4	VideoSwitchHandler	
Realization	Public	Public	
Source -> Destination	SRS142.2.1	VideoSwitchHandler	
Realization	Public	Public	
Source -> Destination	SRS140	VideoSwitchHandler	
Realization	Public	Public	
Source -> Destination	SRS141.1	VideoSwitchHandler	
Realization	Public	Public	
Source -> Destination	SRS105	VideoSwitchHandler	
Realization	Public	Public	
Source -> Destination	SRS140.1	VideoSwitchHandler	
Realization	Public	Public	
Source -> Destination	SRS140.3.1	VideoSwitchHandler	
Realization	Public	Public	
Source -> Destination	SRS107	VideoSwitchHandler	
Realization	Public	Public	
Source -> Destination	SRS98	VideoSwitchHandler	
Realization	Public	Public	
Source -> Destination	SRS140.2	VideoSwitchHandler	
Realization	Public	Public	
Source -> Destination	SRS102	VideoSwitchHandler	
Realization	Public	Public	

Connector	Source	Target	Notes
Source -> Destination	SRS142	VideoSwitchHandler	
Realization	Public	Public	
Source -> Destination	SRS142.1	VideoSwitchHandler	
Realization	Public	Public	
Source -> Destination	SRS142.2	VideoSwitchHandler	
Realization	Public	Public	
Source -> Destination	SRS101	VideoSwitchHandler	
Realization	Public	Public	
Source -> Destination	SRS140.5	VideoSwitchHandler	
Realization	Public	Public	
Source -> Destination	SRS142.1.1	VideoSwitchHandler	
Realization	Public	Public	
Source -> Destination	SRS108	VideoSwitchHandler	
Realization	Public	Public	
Source -> Destination	SRS140.3	VideoSwitchHandler	

Operations

Method	Notes	Parameters
getPubMessage() void Public	When a dlVideoSwitchInventoryUpdate dialog and valid subscription message is received, this method will return either a videoSwitchInventoryMsg message, or an exception if an error was encountered. When a dlVideoSwitchStatusUpdate dialog and valid subscription message is received, this method will return either a videoSwitchStatusMsg message, or an exception if an error was encountered.	String [in] dialog The name of the publication dialog that is associated with the message provided.Message [in] subscriptionMessage The subscription message that was sent for this publication.
getRRResponse() Message Public	When a dlDeviceCancelControlRequest dialog and valid deviceCancelControlRequestMsg message is received, this method will return either a deviceControlResponse message, or an errorReportMsg message if an error was encountered. When a dlDeviceControlStatusRequest dialog and valid deviceControlStatusRequestMsg message is received, this method will return either a deviceControlResponse message, or an	String [in] dialog The name of the RequestResponse dialog that is associated with the message provided.Message [in] requestMessage The request message that was received from the EC.

Method	Notes	Parameters
	errorReportMsg message if an error was encountered. When a dlVideoSwitchControlRequest dialog and valid videoSwitchControlRequestMsg message is received, this method will return either a deviceControlResponse message, or an errorReportMsg message if an error was encountered. When a dlVideoSwitchInventoryRequest dialog and valid deviceInformationRequestMsg message is received, this method will return either a videoSwitchInventory message, or an errorReportMsg message if an error was encountered. When a dlVideoSwitchStatusRequest dialog and valid deviceInformationRequestMsg message is received, this method will return either a videoSwitchStatus message, or an errorReportMsg message if an error was encountered.	
getSubResponse() Message Public	When a dlDeviceInformationSubscription dialog and valid deviceInformationRequestMsg message is received, this method will return either a c2cMessageReceipt message, or an errorReportMsg message if an error was encountered.	String [in] dialog The name of the subscription dialog that is associated with the message provided.Message [in] requestMessage The subscription message that was received from the EC.
Initialize() void Public	A concrete implementation of this method should be called when there is a need to initialize (re-initialize) the data associated with this entity type.	

5.4.1.1.3 Context Diagram

N/A

5.4.1.1.4 Variability Guide

N/A

5.4.1.1.5 Rationale

N/A

5.4.1.2 Entity

5.4.1.2.1 Primary Presentation

Figure 5-23 Entity Modules

The Entity Modules represent and process the data types for each of the entities specified by an Information Layer Standard. Currently the TMDD entity data types are supported.

5.4.1.2.2 Element Catalog

5.4.1.2.2.1 cctv

5.4.1.2.2.1.1 CCTVInventory

Type: **Class** *Package:* cctv

This class represents the set of CCTV Inventory entity information for the center. The information that is maintained is described in TMDD Volume II - Section 3.2.2.2.

Connections

Connector	Source	Target	Notes
Realization	Public	Public	
Source -> Destination	SRS41.3	CCTVInventory	
Realization	Public	Public	
Source -> Destination	SRS41.5	CCTVInventory	
Realization	Public	Public	
Source -> Destination	SRS41	CCTVInventory	
Realization	Public	Public	
Source -> Destination	SRS109.2	CCTVInventory	
Realization	Public	Public	
Source -> Destination	SRS111.1	CCTVInventory	
Realization	Public	Public	
Source -> Destination	SRS111.1.1	CCTVInventory	
Realization	Public	Public	
Source -> Destination	SRS41.1	CCTVInventory	
Realization	Public	Public	
Source -> Destination	SRS41.7	CCTVInventory	
Realization	Public	Public	
Source -> Destination	SRS41.4	CCTVInventory	
Realization	Public	Public	
Source -> Destination	SRS41.6	CCTVInventory	
Realization	Public	Public	
Source -> Destination	SRS41.2	CCTVInventory	

Operations

Method	Notes	Parameters
addEntity() void	Adds a new entity item to the set of entities contained.	String [in] entityId

Method	Notes	Parameters
Public		
addEntityElement() void Public	Add the new element to the set of elements that are currently specified for this entity.	String [in] entityId String [in] elementName String [in] elementValue
deleteEntity() void Public	Deletes an entity item from the set of entities contained.	String [in] entityId
getEntityElement() void Public	Get the value of a specified entity.	String [in] entityId String [in] entityElement
getEntityElements() Array Public	Gathers up a list of available entity elements that match all of the provided filters.	List [in] filterList
initialize() void Public	Initialize the entity data to the provided value set.	ByteArray [in] entitydata
updateEntityElement() void Public	Update the value associated with the specified entity element.	String [in] entityId String [in] entityElementName String [in] entityElementValue

5.4.1.2.2.1.2 CCTVStatus

Type: **Class** *Package:* cctv

This class represents the set of CCTV Status entity information for the center. The information that is maintained is described in TMDD Volume II - Section 3.2.2.3.

Connections

Connector	Source	Target	Notes
Realization	Public	Public	
Source -> Destination	SRS109.3	CCTVStatus	
Realization	Public	Public	
Source -> Destination	SRS41.2	CCTVStatus	
Realization	Public	Public	
Source -> Destination	SRS111.2.1	CCTVStatus	
Realization	Public	Public	

Connector	Source	Target	Notes
Source -> Destination	SRS41.3	CCTVStatus	
Realization	Public	Public	
Source -> Destination	SRS41	CCTVStatus	
Realization	Public	Public	
Source -> Destination	SRS41.6	CCTVStatus	
Realization	Public	Public	
Source -> Destination	SRS41.4	CCTVStatus	
Realization	Public	Public	
Source -> Destination	SRS41.5	CCTVStatus	
Realization	Public	Public	
Source -> Destination	SRS41.7	CCTVStatus	
Realization	Public	Public	
Source -> Destination	SRS41.1	CCTVStatus	
Realization	Public	Public	
Source -> Destination	SRS111.2	CCTVStatus	
Realization	Public	Public	
Source -> Destination	SRS109.1.1	CCTVStatus	

Operations

Method	Notes	Parameters
addEntity() void Public	Adds a new entity item to the set of entities contained.	String [in] entityId
addEntityElement() void Public	Add the new element to the set of elements that are currently specified for this entity.	String [in] entityId String [in] elementName String [in] elementValue
deleteEntity() void Public	Deletes an entity item from the set of entities contained.	String [in] entityId
getEntityElement() void Public	Get the value of a specified entity.	String [in] entityId String [in] entityElement
getEntityElements() Array Public	Gathers up a list of available entity elements that match all of the provided filters.	List [in] filterList
initialize() void	Initialize the entity data to the provided value set.	ByteArray [in] entitydata

Method	Notes	Parameters
Public		
updateEntityElement() void Public	Update the value associated with the specified entity element.	String [in] entityId String [in] entityElementName String [in] entityElementValue

5.4.1.2.2.2 detector

5.4.1.2.2.2.1 ArchivedDataProcessingDocumentationMetaData

Type: **Class** *Package:* detector

This class represents the set of Archived Data Processing Documentation Metadata entity information for the center. The information that is maintained is described in TMDD Volume II - Section 3.2.1.1.

Connections

Connector	Source	Target	Notes
Realization	Public	Public	
Source -> Destination	SRS41.2	ArchivedDataProcessingDocumentationMetaData	
Realization	Public	Public	
Source -> Destination	SRS41.1	ArchivedDataProcessingDocumentationMetaData	
Realization	Public	Public	
Source -> Destination	SRS41.5	ArchivedDataProcessingDocumentationMetaData	
Realization	Public	Public	
Source -> Destination	SRS41.7	ArchivedDataProcessingDocumentationMetaData	
Realization	Public	Public	
Source -> Destination	SRS41.6	ArchivedDataProcessingDocumentationMetaData	
Realization	Public	Public	
Source -> Destination	SRS41.3	ArchivedDataProcessingDocumentationMetaData	
Realization	Public	Public	

Connector	Source	Target	Notes
Source -> Destination	SRS112.1	ArchivedDataProcessingDocumentationMetaData	
Realization	Public	Public	
Source -> Destination	SRS41	ArchivedDataProcessingDocumentationMetaData	
Realization	Public	Public	
Source -> Destination	SRS41.4	ArchivedDataProcessingDocumentationMetaData	

Operations

Method	Notes	Parameters
addEntity() void Public	Adds a new entity item to the set of entities contained.	String [in] entityId
addEntityElement() void Public	Add the new element to the set of elements that are currently specified for this entity.	String [in] entityId String [in] elementName String [in] elementValue
deleteEntity() void Public	Deletes an entity item from the set of entities contained.	String [in] entityId
getEntityElement() void Public	Get the value of a specified entity.	String [in] entityId String [in] entityElement
getEntityElements() Array Public	Gathers up a list of available entity elements that match all of the provided filters.	List [in] filterList
initialize() void Public	Initialize the entity data to the provided value set.	ByteArray [in] entitydata
updateEntityElement() void Public	Update the value associated with the specified entity element.	String [in] entityId String [in] entityElementName String [in] entityElementValue

5.4.1.2.2.2.2 DetectorData

Type: **Class** *Package:* detector

This class represents the set of Detector Data entity information for the center. The information that is maintained is described in TMDD Volume II - Section 3.2.4.1.

Connections

Connector	Source	Target	Notes
Realization	Public	Public	
Source -> Destination	SRS41.1	DetectorData	
Realization	Public	Public	
Source -> Destination	SRS41.2	DetectorData	
Realization	Public	Public	
Source -> Destination	SRS114.1	DetectorData	
Realization	Public	Public	
Source -> Destination	SRS41.7	DetectorData	
Realization	Public	Public	
Source -> Destination	SRS41.6	DetectorData	
Realization	Public	Public	
Source -> Destination	SRS41.5	DetectorData	
Realization	Public	Public	
Source -> Destination	SRS41	DetectorData	
Realization	Public	Public	
Source -> Destination	SRS114.1.1	DetectorData	
Realization	Public	Public	
Source -> Destination	SRS112.2	DetectorData	
Realization	Public	Public	
Source -> Destination	SRS41.3	DetectorData	
Realization	Public	Public	
Source -> Destination	SRS41.4	DetectorData	
Realization	Public	Public	
Source -> Destination	SRS113.1	DetectorData	

Operations

Method	Notes	Parameters
addEntity() void Public	Adds a new entity item to the set of entities contained.	String [in] entityId
addEntityElement() void Public	Add the new element to the set of elements that are currently specified for this entity.	String [in] entityId String [in] elementName

Method	Notes	Parameters
		String [in] elementValue
deleteEntity() void Public	Deletes an entity item from the set of entities contained.	String [in] entityId
getEntityElement() void Public	Get the value of a specified entity.	String [in] entityId String [in] entityElement
getEntityElements() Array Public	Gathers up a list of available entity elements that match all of the provided filters.	List [in] filterList
initialize() void Public	Initialize the entity data to the provided value set.	ByteArray [in] entitydata
updateEntityElement() void Public	Update the value associated with the specified entity element.	String [in] entityId String [in] entityElementName String [in] entityElementValue

5.4.1.2.2.2.3 DetectorInventory

Type: **Class** *Package:* detector

This class represents the set of Detector Inventory entity information for the center. The information that is maintained is described in TMDD Volume II - Section 3.2.4.3.

Connections

Connector	Source	Target	Notes
Realization	Public	Public	
Source -> Destination	SRS41.4	DetectorInventory	
Realization	Public	Public	
Source -> Destination	SRS41.3	DetectorInventory	
Realization	Public	Public	
Source -> Destination	SRS114.2.1	DetectorInventory	
Realization	Public	Public	
Source -> Destination	SRS114.2	DetectorInventory	
Realization	Public	Public	
Source -> Destination	SRS41.6	DetectorInventory	

Connector	Source	Target	Notes
Realization	Public	Public	
Source -> Destination	SRS41	DetectorInventory	
Realization	Public	Public	
Source -> Destination	SRS112.3	DetectorInventory	
Realization	Public	Public	
Source -> Destination	SRS41.5	DetectorInventory	
Realization	Public	Public	
Source -> Destination	SRS41.1	DetectorInventory	
Realization	Public	Public	
Source -> Destination	SRS41.7	DetectorInventory	
Realization	Public	Public	
Source -> Destination	SRS41.2	DetectorInventory	
Realization	Public	Public	
Source -> Destination	SRS113.2	DetectorInventory	

Operations

Method	Notes	Parameters
addEntity() void Public	Adds a new entity item to the set of entities contained.	String [in] entityId
addEntityElement() void Public	Add the new element to the set of elements that are currently specified for this entity.	String [in] entityId String [in] elementName String [in] elementValue
deleteEntity() void Public	Deletes an entity item from the set of entities contained.	String [in] entityId
getEntityElement() void Public	Get the value of a specified entity.	String [in] entityId String [in] entityElement
getEntityElements() Array Public	Gathers up a list of available entity elements that match all of the provided filters.	List [in] filterList
initialize() void Public	Initialize the entity data to the provided value set.	ByteArray [in] entitydata
updateEntityElement() void Public	Update the value associated with the specified entity element.	String [in] entityId String [in] entityElementName

Method	Notes	Parameters
		String [in] entityElementValue

5.4.1.2.2.2.4 DetectorMaintenanceHistory

Type: **Class** *Package:* detector

This class represents the set of Detector Maintenance History entity information for the center. The information that is maintained is described in TMDD Volume II - Section 3.2.4.5.

Connections

Connector	Source	Target	Notes
Realization	Public	Public	
Source -> Destination	SRS41	DetectorMaintenanceHistory	
Realization	Public	Public	
Source -> Destination	SRS41.6	DetectorMaintenanceHistory	
Realization	Public	Public	
Source -> Destination	SRS41.3	DetectorMaintenanceHistory	
Realization	Public	Public	
Source -> Destination	SRS41.4	DetectorMaintenanceHistory	
Realization	Public	Public	
Source -> Destination	SRS112.4	DetectorMaintenanceHistory	
Realization	Public	Public	
Source -> Destination	SRS114.3	DetectorMaintenanceHistory	
Realization	Public	Public	
Source -> Destination	SRS41.2	DetectorMaintenanceHistory	
Realization	Public	Public	
Source -> Destination	SRS41.1	DetectorMaintenanceHistory	
Realization	Public	Public	
Source -> Destination	SRS114.3.1	DetectorMaintenanceHistory	
Realization	Public	Public	

Connector	Source	Target	Notes
Source -> Destination	SRS41.7	DetectorMaintenanceHistory	
Realization	Public	Public	
Source -> Destination	SRS41.5	DetectorMaintenanceHistory	

Operations

Method	Notes	Parameters
addEntity() void Public	Adds a new entity item to the set of entities contained.	String [in] entityId
addEntityElement() void Public	Add the new element to the set of elements that are currently specified for this entity.	String [in] entityId String [in] elementName String [in] elementValue
deleteEntity() void Public	Deletes an entity item from the set of entities contained.	String [in] entityId
getEntityElement() void Public	Get the value of a specified entity.	String [in] entityId String [in] entityElement
getEntityElements() Array Public	Gathers up a list of available entity elements that match all of the provided filters.	List [in] filterList
initialize() void Public	Initialize the entity data to the provided value set.	ByteArray [in] entitydata
updateEntityElement() void Public	Update the value associated with the specified entity element.	String [in] entityId String [in] entityElementName String [in] entityElementValue

5.4.1.2.2.2.5 DetectorStatus

Type: **Class** *Package:* detector

This class represents the set of Detector Status entity information for the center. The information that is maintained is described in TMDD Volume II - Section 3.2.4.6.

Connections

Connector	Source	Target	Notes
Realization	Public	Public	
Source -> Destination	SRS41.7	DetectorStatus	
Realization	Public	Public	
Source -> Destination	SRS41.5	DetectorStatus	
Realization	Public	Public	
Source -> Destination	SRS116.6	DetectorStatus	
Realization	Public	Public	
Source -> Destination	SRS41.6	DetectorStatus	
Realization	Public	Public	
Source -> Destination	SRS113.2	DetectorStatus	
Realization	Public	Public	
Source -> Destination	SRS41	DetectorStatus	
Realization	Public	Public	
Source -> Destination	SRS41.2	DetectorStatus	
Realization	Public	Public	
Source -> Destination	SRS41.3	DetectorStatus	
Realization	Public	Public	
Source -> Destination	SRS112.5	DetectorStatus	
Realization	Public	Public	
Source -> Destination	SRS41.1	DetectorStatus	
Realization	Public	Public	
Source -> Destination	SRS116.1	DetectorStatus	
Realization	Public	Public	
Source -> Destination	SRS41.4	DetectorStatus	

Operations

Method	Notes	Parameters
addEntity() void Public	Adds a new entity item to the set of entities contained.	String [in] entityId
addEntityElement() void Public	Add the new element to the set of elements that are currently specified for this entity.	String [in] entityId String [in] elementName String [in] elementValue
deleteEntity() void Public	Deletes an entity item from the set of entities contained.	String [in] entityId

Method	Notes	Parameters
getEntityElement() void Public	Get the value of a specified entity.	String [in] entityId String [in] entityElement
getEntityElements() Array Public	Gathers up a list of available entity elements that match all of the provided filters.	List [in] filterList
initialize() void Public	Initialize the entity data to the provided value set.	ByteArray [in] entitydata
updateEntityElement() void Public	Update the value associated with the specified entity element.	String [in] entityId String [in] entityElementName String [in] entityElementValue

5.4.1.2.2.3 dms

5.4.1.2.2.3.1 DMSFontTable

Type: **Class** *Package:* dms

This class represents the set of DMS Font Table entity information for the center. The information that is maintained is described in TMDD Volume II - Section 3.2.6.2.

Connections

Connector	Source	Target	Notes
Realization	Public	Public	
Source -> Destination	SRS41.1	DMSFontTable	
Realization	Public	Public	
Source -> Destination	SRS41	DMSFontTable	
Realization	Public	Public	
Source -> Destination	SRS41.6	DMSFontTable	
Realization	Public	Public	
Source -> Destination	SRS41.4	DMSFontTable	
Realization	Public	Public	
Source -> Destination	SRS115.4	DMSFontTable	
Realization	Public	Public	
Source -> Destination	SRS41.3	DMSFontTable	

Connector	Source	Target	Notes
Realization	Public	Public	
Source -> Destination	SRS41.2	DMSFontTable	
Realization	Public	Public	
Source -> Destination	SRS41.5	DMSFontTable	
Realization	Public	Public	
Source -> Destination	SRS41.7	DMSFontTable	

Operations

Method	Notes	Parameters
addEntity() void Public	Adds a new entity item to the set of entities contained.	String [in] entityId
addEntityElement() void Public	Add the new element to the set of elements that are currently specified for this entity.	String [in] entityId String [in] elementName String [in] elementValue
deleteEntity() void Public	Deletes an entity item from the set of entities contained.	String [in] entityId
getEntityElement() void Public	Get the value of a specified entity.	String [in] entityId String [in] entityElement
getEntityElements() Array Public	Gathers up a list of available entity elements that match all of the provided filters.	List [in] filterList
initialize() void Public	Initialize the entity data to the provided value set.	ByteArray [in] entitydata
updateEntityElement() void Public	Update the value associated with the specified entity element.	String [in] entityId String [in] entityElementName String [in] entityElementValue

5.4.1.2.2.3.2 DMSInventory

Type: **Class** *Package:* dms

This class represents the set of DMS Inventory entity information for the center. The information that is maintained is described in TMDD Volume II - Section 3.2.6.4.

Connections

Connector	Source	Target	Notes
Realization	Public	Public	
Source -> Destination	SRS41.3	DMSInventory	
Realization	Public	Public	
Source -> Destination	SRS41.4	DMSInventory	
Realization	Public	Public	
Source -> Destination	SRS116.1	DMSInventory	
Realization	Public	Public	
Source -> Destination	SRS41	DMSInventory	
Realization	Public	Public	
Source -> Destination	SRS41.5	DMSInventory	
Realization	Public	Public	
Source -> Destination	SRS116.4	DMSInventory	
Realization	Public	Public	
Source -> Destination	SRS41.7	DMSInventory	
Realization	Public	Public	
Source -> Destination	SRS41.6	DMSInventory	
Realization	Public	Public	
Source -> Destination	SRS146.1.1	DMSInventory	
Realization	Public	Public	
Source -> Destination	SRS115.5	DMSInventory	
Realization	Public	Public	
Source -> Destination	SRS146.1	DMSInventory	
Realization	Public	Public	
Source -> Destination	SRS41.2	DMSInventory	
Realization	Public	Public	
Source -> Destination	SRS41.1	DMSInventory	

Operations

Method	Notes	Parameters
addEntity() void Public	Adds a new entity item to the set of entities contained.	String [in] entityId
addEntityElement() void Public	Add the new element to the set of elements that are currently specified for this entity.	String [in] entityId String [in] elementName String [in] elementValue

Method	Notes	Parameters
deleteEntity() void Public	Deletes an entity item from the set of entities contained.	String [in] entityId
getEntityElement() void Public	Get the value of a specified entity.	String [in] entityId String [in] entityElement
getEntityElements() Array Public	Gathers up a list of available entity elements that match all of the provided filters.	List [in] filterList
initialize() void Public	Initialize the entity data to the provided value set.	ByteArray [in] entitydata
updateEntityElement() void Public	Update the value associated with the specified entity element.	String [in] entityId String [in] entityElementName String [in] entityElementValue

5.4.1.2.2.3.3 DMSMessageAppearance

Type: **Class** *Package:* dms

This class represents the set of DMS Message Appearance entity information for the center. The information that is maintained is described in TMDD Volume II - Section 3.2.6.5.

Connections

Connector	Source	Target	Notes
Realization	Public	Public	
Source -> Destination	SRS41.3	DMSMessageAppearance	
Realization	Public	Public	
Source -> Destination	SRS41.6	DMSMessageAppearance	
Realization	Public	Public	
Source -> Destination	SRS41.5	DMSMessageAppearance	
Realization	Public	Public	
Source -> Destination	SRS41.7	DMSMessageAppearance	
Realization	Public	Public	

Connector	Source	Target	Notes
Source -> Destination	SRS41.2	DMSMessageAppearance	
Realization	Public	Public	
Source -> Destination	SRS41.4	DMSMessageAppearance	
Realization	Public	Public	
Source -> Destination	SRS41.1	DMSMessageAppearance	
Realization	Public	Public	
Source -> Destination	SRS115.6	DMSMessageAppearance	
Realization	Public	Public	
Source -> Destination	SRS41	DMSMessageAppearance	

Operations

Method	Notes	Parameters
addEntity() void Public	Adds a new entity item to the set of entities contained.	String [in] entityId
addEntityElement() void Public	Add the new element to the set of elements that are currently specified for this entity.	String [in] entityId String [in] elementName String [in] elementValue
deleteEntity() void Public	Deletes an entity item from the set of entities contained.	String [in] entityId
getEntityElement() void Public	Get the value of a specified entity.	String [in] entityId String [in] entityElement
getEntityElements() Array Public	Gathers up a list of available entity elements that match all of the provided filters.	List [in] filterList
initialize() void Public	Initialize the entity data to the provided value set.	ByteArray [in] entitydata
updateEntityElement() void Public	Update the value associated with the specified entity element.	String [in] entityId String [in] entityElementName String [in] entityElementValue

5.4.1.2.2.3.4 DMSMessageInventory

Type: **Class** *Package:* dms

This class represents the set of DMS Message Inventory entity information for the center. The information that is maintained is described in TMDD Volume II - Section 3.2.6.7.

Connections

Connector	Source	Target	Notes
Realization	Public	Public	
Source -> Destination	SRS41.5	DMSMessageInventory	
Realization	Public	Public	
Source -> Destination	SRS41.3	DMSMessageInventory	
Realization	Public	Public	
Source -> Destination	SRS146.2.1	DMSMessageInventory	
Realization	Public	Public	
Source -> Destination	SRS146.2	DMSMessageInventory	
Realization	Public	Public	
Source -> Destination	SRS116.2	DMSMessageInventory	
Realization	Public	Public	
Source -> Destination	SRS41.7	DMSMessageInventory	
Realization	Public	Public	
Source -> Destination	SRS115.7	DMSMessageInventory	
Realization	Public	Public	
Source -> Destination	SRS41	DMSMessageInventory	
Realization	Public	Public	
Source -> Destination	SRS41.4	DMSMessageInventory	
Realization	Public	Public	
Source -> Destination	SRS116.5	DMSMessageInventory	
Realization	Public	Public	
Source -> Destination	SRS41.6	DMSMessageInventory	
Realization	Public	Public	
Source -> Destination	SRS41.2	DMSMessageInventory	
Realization	Public	Public	
Source -> Destination	SRS41.1	DMSMessageInventory	

Operations

Method	Notes	Parameters

Method	Notes	Parameters
addEntity() void Public	Adds a new entity item to the set of entities contained.	String [in] entityId
addEntityElement() void Public	Add the new element to the set of elements that are currently specified for this entity.	String [in] entityId String [in] elementName String [in] elementValue
deleteEntity() void Public	Deletes an entity item from the set of entities contained.	String [in] entityId
getEntityElement() void Public	Get the value of a specified entity.	String [in] entityId String [in] entityElement
getEntityElements() Array Public	Gathers up a list of available entity elements that match all of the provided filters.	List [in] filterList
initialize() void Public	Initialize the entity data to the provided value set.	ByteArray [in] entitydata
updateEntityElement() void Public	Update the value associated with the specified entity element.	String [in] entityId String [in] entityElementName String [in] entityElementValue

5.4.1.2.2.3.5 DMSStatus

Type: **Class** *Package:* dms

This class represents the set of DMS Status entity information for the center. The information that is maintained is described in TMDD Volume II - Section 3.2.6.10.

Connections

Connector	Source	Target	Notes
Realization	Public	Public	
Source -> Destination	SRS41.2	DMSStatus	
Realization	Public	Public	
Source -> Destination	SRS115.3.1	DMSStatus	
Realization	Public	Public	
Source -> Destination	SRS115.9	DMSStatus	

Connector	Source	Target	Notes
Realization	Public	Public	
Source -> Destination	SRS41.4	DMSStatus	
Realization	Public	Public	
Source -> Destination	SRS41.7	DMSStatus	
Realization	Public	Public	
Source -> Destination	SRS146.3	DMSStatus	
Realization	Public	Public	
Source -> Destination	SRS41.6	DMSStatus	
Realization	Public	Public	
Source -> Destination	SRS146.3.1	DMSStatus	
Realization	Public	Public	
Source -> Destination	SRS41.3	DMSStatus	
Realization	Public	Public	
Source -> Destination	SRS41	DMSStatus	
Realization	Public	Public	
Source -> Destination	SRS41.5	DMSStatus	
Realization	Public	Public	
Source -> Destination	SRS41.1	DMSStatus	
Realization	Public	Public	
Source -> Destination	SRS116.1	DMSStatus	

Operations

Method	Notes	Parameters
addEntity() void Public	Adds a new entity item to the set of entities contained.	String [in] entityId
addEntityElement() void Public	Add the new element to the set of elements that are currently specified for this entity.	String [in] entityId String [in] elementName String [in] elementValue
deleteEntity() void Public	Deletes an entity item from the set of entities contained.	String [in] entityId
getEntityElement() void Public	Get the value of a specified entity.	String [in] entityId String [in] entityElement
getEntityElements() Array	Gathers up a list of available entity elements	List [in] filterList

Method	Notes	Parameters
Public	that match all of the provided filters.	
initialize() void Public	Initialize the entity data to the provided value set.	ByteArray [in] entitydata
updateEntityElement() void Public	Update the value associated with the specified entity element.	String [in] entityId String [in] entityElementName String [in] entityElementValue

5.4.1.2.2.4 ess

5.4.1.2.2.4.1 ESSInventory

Type: **Class** *Package:* ess

This class represents the set of ESS Inventory entity information for the center. The information that is maintained is described in TMDD Volume II - Section 3.2.7.1.

Connections

Connector	Source	Target	Notes
Realization	Public	Public	
Source -> Destination	SRS117.1	ESSInventory	
Realization	Public	Public	
Source -> Destination	SRS41	ESSInventory	
Realization	Public	Public	
Source -> Destination	SRS147.1	ESSInventory	
Realization	Public	Public	
Source -> Destination	SRS41.5	ESSInventory	
Realization	Public	Public	
Source -> Destination	SRS41.1	ESSInventory	
Realization	Public	Public	
Source -> Destination	SRS41.6	ESSInventory	
Realization	Public	Public	
Source -> Destination	SRS41.7	ESSInventory	
Realization	Public	Public	
Source -> Destination	SRS118.1	ESSInventory	
Realization	Public	Public	

Connector	Source	Target	Notes
Source -> Destination	SRS118.1.1	ESSInventory	
Realization	Public	Public	
Source -> Destination	SRS117.6	ESSInventory	
Realization	Public	Public	
Source -> Destination	SRS41.2	ESSInventory	
Realization	Public	Public	
Source -> Destination	SRS41.4	ESSInventory	
Realization	Public	Public	
Source -> Destination	SRS41.3	ESSInventory	

Operations

Method	Notes	Parameters
addEntity() void Public	Adds a new entity item to the set of entities contained.	String [in] entityId
addEntityElement() void Public	Add the new element to the set of elements that are currently specified for this entity.	String [in] entityId String [in] elementName String [in] elementValue
deleteEntity() void Public	Deletes an entity item from the set of entities contained.	String [in] entityId
getEntityElement() void Public	Get the value of a specified entity.	String [in] entityId String [in] entityElement
getEntityElements() Array Public	Gathers up a list of available entity elements that match all of the provided filters.	List [in] filterList
initialize() void Public	Initialize the entity data to the provided value set.	ByteArray [in] entitydata
updateEntityElement() void Public	Update the value associated with the specified entity element.	String [in] entityId String [in] entityElementName String [in] entityElementValue

5.4.1.2.2.4.2 ESSObservationMetadata

Type: **Class** *Package:* ess

This class represents the set of ESS Observation Report entity information for the center. The information that is maintained is described in TMDD Volume II - Section 3.2.7.2.

Connections

Connector	Source	Target	Notes
Realization	Public	Public	
Source -> Destination	SRS41.7	ESSObservationMetadata	
Realization	Public	Public	
Source -> Destination	SRS41	ESSObservationMetadata	
Realization	Public	Public	
Source -> Destination	SRS41.3	ESSObservationMetadata	
Realization	Public	Public	
Source -> Destination	SRS41.4	ESSObservationMetadata	
Realization	Public	Public	
Source -> Destination	SRS41.1	ESSObservationMetadata	
Realization	Public	Public	
Source -> Destination	SRS41.6	ESSObservationMetadata	
Realization	Public	Public	
Source -> Destination	SRS117.2	ESSObservationMetadata	
Realization	Public	Public	
Source -> Destination	SRS41.2	ESSObservationMetadata	
Realization	Public	Public	
Source -> Destination	SRS41.5	ESSObservationMetadata	

Operations

Method	Notes	Parameters
addEntity() void Public	Adds a new entity item to the set of entities contained.	String [in] entityId
addEntityElement() void Public	Add the new element to the set of elements that are currently specified for this entity.	String [in] entityId String [in] elementName

Method	Notes	Parameters
		String [in] elementValue
deleteEntity() void Public	Deletes an entity item from the set of entities contained.	String [in] entityId
getEntityElement() void Public	Get the value of a specified entity.	String [in] entityId String [in] entityElement
getEntityElements() Array Public	Gathers up a list of available entity elements that match all of the provided filters.	List [in] filterList
initialize() void Public	Initialize the entity data to the provided value set.	ByteArray [in] entitydata
updateEntityElement() void Public	Update the value associated with the specified entity element.	String [in] entityId String [in] entityElementName String [in] entityElementValue

5.4.1.2.2.4.3 ESSObservationReport

Type: **Class** *Package:* ess

This class represents the set of ESS Observation Report entity information for the center. The information that is maintained is described in TMDD Volume II - Section 3.2.7.3.

Connections

Connector	Source	Target	Notes
Realization	Public	Public	
Source -> Destination	SRS117.3	ESSObservationReport	
Realization	Public	Public	
Source -> Destination	SRS41.2	ESSObservationReport	
Realization	Public	Public	
Source -> Destination	SRS41.6	ESSObservationReport	
Realization	Public	Public	
Source -> Destination	SRS41	ESSObservationReport	
Realization	Public	Public	
Source -> Destination	SRS41.4	ESSObservationReport	

Connector	Source	Target	Notes
Realization	Public	Public	
Source -> Destination	SRS118.2	ESSObservationReport	
Realization	Public	Public	
Source -> Destination	SRS41.7	ESSObservationReport	
Realization	Public	Public	
Source -> Destination	SRS41.1	ESSObservationReport	
Realization	Public	Public	
Source -> Destination	SRS117.6	ESSObservationReport	
Realization	Public	Public	
Source -> Destination	SRS118.2.1	ESSObservationReport	
Realization	Public	Public	
Source -> Destination	SRS41.3	ESSObservationReport	
Realization	Public	Public	
Source -> Destination	SRS41.5	ESSObservationReport	

Operations

Method	Notes	Parameters
addEntity() void Public	Adds a new entity item to the set of entities contained.	String [in] entityId
addEntityElement() void Public	Add the new element to the set of elements that are currently specified for this entity.	String [in] entityId String [in] elementName String [in] elementValue
deleteEntity() void Public	Deletes an entity item from the set of entities contained.	String [in] entityId
getEntityElement() void Public	Get the value of a specified entity.	String [in] entityId String [in] entityElement
getEntityElements() Array Public	Gathers up a list of available entity elements that match all of the provided filters.	List [in] filterList
initialize() void Public	Initialize the entity data to the provided value set.	ByteArray [in] entitydata
updateEntityElement() void Public	Update the value associated with the specified entity element.	String [in] entityId String [in] entityElementName

Method	Notes	Parameters
		String [in] entityElementValue

5.4.1.2.2.4.4 ESSStatus

Type: **Class** *Package:* ess

This class represents the set of ESS Status entity information for the center. The information that is maintained is described in TMDD Volume II - Section 3.2.7.4.

Connections

Connector	Source	Target	Notes
Realization	Public	Public	
Source -> Destination	SRS41.2	ESSStatus	
Realization	Public	Public	
Source -> Destination	SRS41.6	ESSStatus	
Realization	Public	Public	
Source -> Destination	SRS41.4	ESSStatus	
Realization	Public	Public	
Source -> Destination	SRS118.3.1	ESSStatus	
Realization	Public	Public	
Source -> Destination	SRS41.7	ESSStatus	
Realization	Public	Public	
Source -> Destination	SRS147.1	ESSStatus	
Realization	Public	Public	
Source -> Destination	SRS41	ESSStatus	
Realization	Public	Public	
Source -> Destination	SRS118.3	ESSStatus	
Realization	Public	Public	
Source -> Destination	SRS117.6	ESSStatus	
Realization	Public	Public	
Source -> Destination	SRS117.4	ESSStatus	
Realization	Public	Public	
Source -> Destination	SRS41.3	ESSStatus	
Realization	Public	Public	
Source -> Destination	SRS41.1	ESSStatus	
Realization	Public	Public	

Connector	Source	Target	Notes
Source -> Destination	SRS41.5	ESSStatus	

Operations

Method	Notes	Parameters
addEntity() void Public	Adds a new entity item to the set of entities contained.	String [in] entityId
addEntityElement() void Public	Add the new element to the set of elements that are currently specified for this entity.	String [in] entityId String [in] elementName String [in] elementValue
deleteEntity() void Public	Deletes an entity item from the set of entities contained.	String [in] entityId
getEntityElement() void Public	Get the value of a specified entity.	String [in] entityId String [in] entityElement
getEntityElements() Array Public	Gathers up a list of available entity elements that match all of the provided filters.	List [in] filterList
initialize() void Public	Initialize the entity data to the provided value set.	ByteArray [in] entitydata
updateEntityElement() void Public	Update the value associated with the specified entity element.	String [in] entityId String [in] entityElementName String [in] entityElementValue

5.4.1.2.2.5 events

5.4.1.2.2.5.1 EventUpdates

Type: **Class** *Package:* events

This class represents the set of Event Updates entity information for the center. The information that is maintained is described in TMDD Volume II - Section 3.2.8.4.

Connections

Connector	Source	Target	Notes
Realization	Public	Public	

Connector	Source	Target	Notes
Source -> Destination	SRS41.6	EventUpdates	
Realization	Public	Public	
Source -> Destination	SRS121.3.1	EventUpdates	
Realization	Public	Public	
Source -> Destination	SRS120.3	EventUpdates	
Realization	Public	Public	
Source -> Destination	SRS41.3	EventUpdates	
Realization	Public	Public	
Source -> Destination	SRS41.2	EventUpdates	
Realization	Public	Public	
Source -> Destination	SRS41	EventUpdates	
Realization	Public	Public	
Source -> Destination	SRS121.3	EventUpdates	
Realization	Public	Public	
Source -> Destination	SRS41.5	EventUpdates	
Realization	Public	Public	
Source -> Destination	SRS41.1	EventUpdates	
Realization	Public	Public	
Source -> Destination	SRS41.7	EventUpdates	
Realization	Public	Public	
Source -> Destination	SRS41.4	EventUpdates	
Realization	Public	Public	
Source -> Destination	SRS119.3	EventUpdates	

Operations

Method	Notes	Parameters
addEntity() void Public	Adds a new entity item to the set of entities contained.	String [in] entityId
addEntityElement() void Public	Add the new element to the set of elements that are currently specified for this entity.	String [in] entityId String [in] elementName String [in] elementValue
deleteEntity() void Public	Deletes an entity item from the set of entities contained.	String [in] entityId
getEntityElement() void	Get the value of a specified entity.	String [in] entityId

Method	Notes	Parameters
Public		String [in] entityElement
getEntityElements() Array Public	Gathers up a list of available entity elements that match all of the provided filters.	List [in] filterList
initialize() void Public	Initialize the entity data to the provided value set.	ByteArray [in] entitydata
updateEntityElement() void Public	Update the value associated with the specified entity element.	String [in] entityId String [in] entityElementName String [in] entityElementValue

5.4.1.2.2.5.2 EventsActionLog

Type: **Class** *Package:* events

This class represents the set of Events Action Log entity information for the center. The information that is maintained is described in TMDD Volume II - Section 3.2.8.1.

Connections

Connector	Source	Target	Notes
Realization	Public	Public	
Source -> Destination	SRS121.1	EventsActionLog	
Realization	Public	Public	
Source -> Destination	SRS41.7	EventsActionLog	
Realization	Public	Public	
Source -> Destination	SRS41.3	EventsActionLog	
Realization	Public	Public	
Source -> Destination	SRS120.1	EventsActionLog	
Realization	Public	Public	
Source -> Destination	SRS41.2	EventsActionLog	
Realization	Public	Public	
Source -> Destination	SRS119.1	EventsActionLog	
Realization	Public	Public	
Source -> Destination	SRS121.1.1	EventsActionLog	
Realization	Public	Public	
Source -> Destination	SRS41.4	EventsActionLog	

Connector	Source	Target	Notes
Realization	Public	Public	
Source -> Destination	SRS41.1	EventsActionLog	
Realization	Public	Public	
Source -> Destination	SRS41	EventsActionLog	
Realization	Public	Public	
Source -> Destination	SRS41.5	EventsActionLog	
Realization	Public	Public	
Source -> Destination	SRS41.6	EventsActionLog	

Operations

Method	Notes	Parameters
addEntity() void Public	Adds a new entity item to the set of entities contained.	String [in] entityId
addEntityElement() void Public	Add the new element to the set of elements that are currently specified for this entity.	String [in] entityId String [in] elementName String [in] elementValue
deleteEntity() void Public	Deletes an entity item from the set of entities contained.	String [in] entityId
getEntityElement() void Public	Get the value of a specified entity.	String [in] entityId String [in] entityElement
getEntityElements() Array Public	Gathers up a list of available entity elements that match all of the provided filters.	List [in] filterList
initialize() void Public	Initialize the entity data to the provided value set.	ByteArray [in] entitydata
updateEntityElement() void Public	Update the value associated with the specified entity element.	String [in] entityId String [in] entityElementName String [in] entityElementValue

5.4.1.2.2.5.3 EventsIndex

Type: **Class** *Package:* events

This class represents the set of Events Index entity information for the center. The information that is maintained is described in TMDD Volume II - Section 3.2.8.2.

Connections

Connector	Source	Target	Notes
Realization	Public	Public	
Source -> Destination	SRS41.5	EventsIndex	
Realization	Public	Public	
Source -> Destination	SRS120.2	EventsIndex	
Realization	Public	Public	
Source -> Destination	SRS41.1	EventsIndex	
Realization	Public	Public	
Source -> Destination	SRS41.3	EventsIndex	
Realization	Public	Public	
Source -> Destination	SRS121.2	EventsIndex	
Realization	Public	Public	
Source -> Destination	SRS41.6	EventsIndex	
Realization	Public	Public	
Source -> Destination	SRS119.2	EventsIndex	
Realization	Public	Public	
Source -> Destination	SRS41.4	EventsIndex	
Realization	Public	Public	
Source -> Destination	SRS41.2	EventsIndex	
Realization	Public	Public	
Source -> Destination	SRS41.7	EventsIndex	
Realization	Public	Public	
Source -> Destination	SRS41	EventsIndex	
Realization	Public	Public	
Source -> Destination	SRS121.2.1	EventsIndex	

Operations

Method	Notes	Parameters
addEntity() void Public	Adds a new entity item to the set of entities contained.	String [in] entityId
addEntityElement() void Public	Add the new element to the set of elements that are currently specified for this entity.	String [in] entityId String [in] elementName String [in] elementValue

Method	Notes	Parameters
deleteEntity() void Public	Deletes an entity item from the set of entities contained.	String [in] entityId
getEntityElement() void Public	Get the value of a specified entity.	String [in] entityId String [in] entityElement
getEntityElements() Array Public	Gathers up a list of available entity elements that match all of the provided filters.	List [in] filterList
initialize() void Public	Initialize the entity data to the provided value set.	ByteArray [in] entitydata
updateEntityElement() void Public	Update the value associated with the specified entity element.	String [in] entityId String [in] entityElementName String [in] entityElementValue

5.4.1.2.2.6 gate

5.4.1.2.2.6.1 GateControlSchedule

Type: **Class** *Package:* gate

This class represents the set of Gate Inventory entity information for the center. The information that is maintained is described in TMDD Volume II - Section 3.2.9.2.

Connections

Connector	Source	Target	Notes
Realization	Public	Public	
Source -> Destination	SRS124.1.1	GateControlSchedule	
Realization	Public	Public	
Source -> Destination	SRS123.1	GateControlSchedule	
Realization	Public	Public	
Source -> Destination	SRS41.2	GateControlSchedule	
Realization	Public	Public	
Source -> Destination	SRS122.4	GateControlSchedule	
Realization	Public	Public	
Source -> Destination	SRS41.6	GateControlSchedule	

Connector	Source	Target	Notes
Realization	Public	Public	
Source -> Destination	SRS41	GateControlSchedule	
Realization	Public	Public	
Source -> Destination	SRS41.1	GateControlSchedule	
Realization	Public	Public	
Source -> Destination	SRS41.3	GateControlSchedule	
Realization	Public	Public	
Source -> Destination	SRS124.1	GateControlSchedule	
Realization	Public	Public	
Source -> Destination	SRS41.5	GateControlSchedule	
Realization	Public	Public	
Source -> Destination	SRS41.7	GateControlSchedule	
Realization	Public	Public	
Source -> Destination	SRS41.4	GateControlSchedule	

Operations

Method	Notes	Parameters
addEntity() void Public	Adds a new entity item to the set of entities contained.	String [in] entityId
addEntityElement() void Public	Add the new element to the set of elements that are currently specified for this entity.	String [in] entityId String [in] elementName String [in] elementValue
deleteEntity() void Public	Deletes an entity item from the set of entities contained.	String [in] entityId
getEntityElement() void Public	Get the value of a specified entity.	String [in] entityId String [in] entityElement
getEntityElements() Array Public	Gathers up a list of available entity elements that match all of the provided filters.	List [in] filterList
initialize() void Public	Initialize the entity data to the provided value set.	ByteArray [in] entitydata
updateEntityElement() void Public	Update the value associated with the specified entity element.	String [in] entityId String [in] entityElementName

Method	Notes	Parameters
		String [in] entityElementValue

5.4.1.2.2.6.2 GateInventory

Type: **Class** *Package:* gate

This class represents the set of Gate Inventory entity information for the center. The information that is maintained is described in TMDD Volume II - Section 3.2.9.3.

Connections

Connector	Source	Target	Notes
Realization	Public	Public	
Source -> Destination	SRS123.1	GateInventory	
Realization	Public	Public	
Source -> Destination	SRS124.2.1	GateInventory	
Realization	Public	Public	
Source -> Destination	SRS41.6	GateInventory	
Realization	Public	Public	
Source -> Destination	SRS41.1	GateInventory	
Realization	Public	Public	
Source -> Destination	SRS122.5	GateInventory	
Realization	Public	Public	
Source -> Destination	SRS41.2	GateInventory	
Realization	Public	Public	
Source -> Destination	SRS41.7	GateInventory	
Realization	Public	Public	
Source -> Destination	SRS41.3	GateInventory	
Realization	Public	Public	
Source -> Destination	SRS41	GateInventory	
Realization	Public	Public	
Source -> Destination	SRS41.4	GateInventory	
Realization	Public	Public	
Source -> Destination	SRS124.2	GateInventory	
Realization	Public	Public	
Source -> Destination	SRS41.5	GateInventory	

Operations

Method	Notes	Parameters
addEntity() void Public	Adds a new entity item to the set of entities contained.	String [in] entityId
addEntityElement() void Public	Add the new element to the set of elements that are currently specified for this entity.	String [in] entityId String [in] elementName String [in] elementValue
deleteEntity() void Public	Deletes an entity item from the set of entities contained.	String [in] entityId
getEntityElement() void Public	Get the value of a specified entity.	String [in] entityId String [in] entityElement
getEntityElements() Array Public	Gathers up a list of available entity elements that match all of the provided filters.	List [in] filterList
initialize() void Public	Initialize the entity data to the provided value set.	ByteArray [in] entitydata
updateEntityElement() void Public	Update the value associated with the specified entity element.	String [in] entityId String [in] entityElementName String [in] entityElementValue

5.4.1.2.2.6.3 GateStatus

Type: **Class** *Package:* gate

This class represents the set of Gate Status entity information for the center. The information that is maintained is described in TMDD Volume II - Section 3.2.9.4.

Connections

Connector	Source	Target	Notes
Realization	Public	Public	
Source -> Destination	SRS41.6	GateStatus	
Realization	Public	Public	
Source -> Destination	SRS41.4	GateStatus	

Connector	Source	Target	Notes
Realization	Public	Public	
Source -> Destination	SRS41.3	GateStatus	
Realization	Public	Public	
Source -> Destination	SRS41.5	GateStatus	
Realization	Public	Public	
Source -> Destination	SRS122.3.1	GateStatus	
Realization	Public	Public	
Source -> Destination	SRS122.6	GateStatus	
Realization	Public	Public	
Source -> Destination	SRS124.3.1	GateStatus	
Realization	Public	Public	
Source -> Destination	SRS41.7	GateStatus	
Realization	Public	Public	
Source -> Destination	SRS123.1	GateStatus	
Realization	Public	Public	
Source -> Destination	SRS41.1	GateStatus	
Realization	Public	Public	
Source -> Destination	SRS41	GateStatus	
Realization	Public	Public	
Source -> Destination	SRS41.2	GateStatus	
Realization	Public	Public	
Source -> Destination	SRS124.3	GateStatus	

Operations

Method	Notes	Parameters
addEntity() void Public	Adds a new entity item to the set of entities contained.	String [in] entityId
addEntityElement() void Public	Add the new element to the set of elements that are currently specified for this entity.	String [in] entityId String [in] elementName String [in] elementValue
deleteEntity() void Public	Deletes an entity item from the set of entities contained.	String [in] entityId
getEntityElement() void Public	Get the value of a specified entity.	String [in] entityId String [in] entityElement

Method	Notes	Parameters
getEntityElements() Array Public	Gathers up a list of available entity elements that match all of the provided filters.	List [in] filterList
initialize() void Public	Initialize the entity data to the provided value set.	ByteArray [in] entitydata
updateEntityElement() void Public	Update the value associated with the specified entity element.	String [in] entityId String [in] entityElementName String [in] entityElementValue

5.4.1.2.2.7 har

5.4.1.2.2.7.1 HARControlSchedule

Type: **Class** *Package:* har

This class represents the set of HAR Inventory entity information for the center. The information that is maintained is described in TMDD Volume II - Section 3.2.10.2.

Connections

Connector	Source	Target	Notes
Realization	Public	Public	
Source -> Destination	SRS41	HARControlSchedule	
Realization	Public	Public	
Source -> Destination	SRS127.1.1	HARControlSchedule	
Realization	Public	Public	
Source -> Destination	SRS41.6	HARControlSchedule	
Realization	Public	Public	
Source -> Destination	SRS127.1	HARControlSchedule	
Realization	Public	Public	
Source -> Destination	SRS41.3	HARControlSchedule	
Realization	Public	Public	
Source -> Destination	SRS41.1	HARControlSchedule	
Realization	Public	Public	
Source -> Destination	SRS41.7	HARControlSchedule	
Realization	Public	Public	

Connector	Source	Target	Notes
Source -> Destination	SRS126.1	HARControlSchedule	
Realization	Public	Public	
Source -> Destination	SRS125.4	HARControlSchedule	
Realization	Public	Public	
Source -> Destination	SRS41.4	HARControlSchedule	
Realization	Public	Public	
Source -> Destination	SRS41.2	HARControlSchedule	
Realization	Public	Public	
Source -> Destination	SRS41.5	HARControlSchedule	

Operations

Method	Notes	Parameters
addEntity() void Public	Adds a new entity item to the set of entities contained.	String [in] entityId
addEntityElement() void Public	Add the new element to the set of elements that are currently specified for this entity.	String [in] entityId String [in] elementName String [in] elementValue
deleteEntity() void Public	Deletes an entity item from the set of entities contained.	String [in] entityId
getEntityElement() void Public	Get the value of a specified entity.	String [in] entityId String [in] entityElement
getEntityElements() Array Public	Gathers up a list of available entity elements that match all of the provided filters.	List [in] filterList
initialize() void Public	Initialize the entity data to the provided value set.	ByteArray [in] entitydata
updateEntityElement() void Public	Update the value associated with the specified entity element.	String [in] entityId String [in] entityElementName String [in] entityElementValue

5.4.1.2.2.7.2 HARInventory

Type: **Class** *Package:* har

This class represents the set of HAR Inventory entity information for the center. The information that is maintained is described in TMDD Volume II - Section 3.2.10.3.

Connections

Connector	Source	Target	Notes
Realization	Public	Public	
Source -> Destination	SRS41.1	HARInventory	
Realization	Public	Public	
Source -> Destination	SRS127.2	HARInventory	
Realization	Public	Public	
Source -> Destination	SRS127.2.1	HARInventory	
Realization	Public	Public	
Source -> Destination	SRS41.4	HARInventory	
Realization	Public	Public	
Source -> Destination	SRS126.1	HARInventory	
Realization	Public	Public	
Source -> Destination	SRS41.6	HARInventory	
Realization	Public	Public	
Source -> Destination	SRS41.5	HARInventory	
Realization	Public	Public	
Source -> Destination	SRS41.2	HARInventory	
Realization	Public	Public	
Source -> Destination	SRS125.5	HARInventory	
Realization	Public	Public	
Source -> Destination	SRS41	HARInventory	
Realization	Public	Public	
Source -> Destination	SRS41.7	HARInventory	
Realization	Public	Public	
Source -> Destination	SRS41.3	HARInventory	

Operations

Method	Notes	Parameters
addEntity() void Public	Adds a new entity item to the set of entities contained.	String [in] entityId
addEntityElement() void Public	Add the new element to the set of elements that are currently specified for this entity.	String [in] entityId String [in] elementName

Method	Notes	Parameters
		String [in] elementValue
deleteEntity() void Public	Deletes an entity item from the set of entities contained.	String [in] entityId
getEntityElement() void Public	Get the value of a specified entity.	String [in] entityId String [in] entityElement
getEntityElements() Array Public	Gathers up a list of available entity elements that match all of the provided filters.	List [in] filterList
initialize() void Public	Initialize the entity data to the provided value set.	ByteArray [in] entitydata
updateEntityElement() void Public	Update the value associated with the specified entity element.	String [in] entityId String [in] entityElementName String [in] entityElementValue

5.4.1.2.2.7.3 HARMessageInventory

Type: **Class** *Package:* har

This class represents the set of HAR Message Inventory entity information for the center. The information that is maintained is described in TMDD Volume II - Section 3.2.10.4.

Connections

Connector	Source	Target	Notes
Realization	Public	Public	
Source -> Destination	SRS41.6	HARMessageInventory	
Realization	Public	Public	
Source -> Destination	SRS41.4	HARMessageInventory	
Realization	Public	Public	
Source -> Destination	SRS41	HARMessageInventory	
Realization	Public	Public	
Source -> Destination	SRS41.7	HARMessageInventory	
Realization	Public	Public	
Source -> Destination	SRS126.1	HARMessageInventory	

Connector	Source	Target	Notes
Realization	Public	Public	
Source -> Destination	SRS41.3	HARMessageInventory	
Realization	Public	Public	
Source -> Destination	SRS41.2	HARMessageInventory	
Realization	Public	Public	
Source -> Destination	SRS127.3.1	HARMessageInventory	
Realization	Public	Public	
Source -> Destination	SRS41.5	HARMessageInventory	
Realization	Public	Public	
Source -> Destination	SRS125.6	HARMessageInventory	
Realization	Public	Public	
Source -> Destination	SRS41.1	HARMessageInventory	
Realization	Public	Public	
Source -> Destination	SRS127.3	HARMessageInventory	

Operations

Method	Notes	Parameters
addEntity() void Public	Adds a new entity item to the set of entities contained.	String [in] entityId
addEntityElement() void Public	Add the new element to the set of elements that are currently specified for this entity.	String [in] entityId String [in] elementName String [in] elementValue
deleteEntity() void Public	Deletes an entity item from the set of entities contained.	String [in] entityId
getEntityElement() void Public	Get the value of a specified entity.	String [in] entityId String [in] entityElement
getEntityElements() Array Public	Gathers up a list of available entity elements that match all of the provided filters.	List [in] filterList
initialize() void Public	Initialize the entity data to the provided value set.	ByteArray [in] entitydata
updateEntityElement() void Public	Update the value associated with the specified entity element.	String [in] entityId String [in] entityElementName

Method	Notes	Parameters
		String [in] entityElementValue

5.4.1.2.2.7.4 HARStatus

Type: **Class** *Package:* har

This class represents the set of HAR Status entity information for the center. The information that is maintained is described in TMDD Volume II - Section 3.2.10.6.

Connections

Connector	Source	Target	Notes
Realization	Public	Public	
Source -> Destination	SRS126.1	HARStatus	
Realization	Public	Public	
Source -> Destination	SRS125.8	HARStatus	
Realization	Public	Public	
Source -> Destination	SRS127.4	HARStatus	
Realization	Public	Public	
Source -> Destination	SRS41.2	HARStatus	
Realization	Public	Public	
Source -> Destination	SRS41.7	HARStatus	
Realization	Public	Public	
Source -> Destination	SRS127.4.1	HARStatus	
Realization	Public	Public	
Source -> Destination	SRS41.4	HARStatus	
Realization	Public	Public	
Source -> Destination	SRS41.1	HARStatus	
Realization	Public	Public	
Source -> Destination	SRS41.6	HARStatus	
Realization	Public	Public	
Source -> Destination	SRS41.5	HARStatus	
Realization	Public	Public	
Source -> Destination	SRS41.3	HARStatus	
Realization	Public	Public	
Source -> Destination	SRS125.3.1	HARStatus	
Realization	Public	Public	

Connector	Source	Target	Notes
Source -> Destination	SRS41	HARStatus	

Operations

Method	Notes	Parameters
addEntity() void Public	Adds a new entity item to the set of entities contained.	String [in] entityId
addEntityElement() void Public	Add the new element to the set of elements that are currently specified for this entity.	String [in] entityId String [in] elementName String [in] elementValue
deleteEntity() void Public	Deletes an entity item from the set of entities contained.	String [in] entityId
getEntityElement() void Public	Get the value of a specified entity.	String [in] entityId String [in] entityElement
getEntityElements() Array Public	Gathers up a list of available entity elements that match all of the provided filters.	List [in] filterList
initialize() void Public	Initialize the entity data to the provided value set.	ByteArray [in] entitydata
updateEntityElement() void Public	Update the value associated with the specified entity element.	String [in] entityId String [in] entityElementName String [in] entityElementValue

5.4.1.2.2.8 lcs

5.4.1.2.2.8.1 LaneControlSignalControlSchedule

Type: **Class** *Package:* lcs

This class represents the set of Lane Control Signal Control Schedule entity information for the center. The information that is maintained is described in TMDD Volume II - Section 3.2.12.2.

Connections

Connector	Source	Target	Notes
Realization	Public	Public	

Connector	Source	Target	Notes
Source -> Destination	SRS41.5	LaneControlSignalControlSchedule	
Realization	Public	Public	
Source -> Destination	SRS41.6	LaneControlSignalControlSchedule	
Realization	Public	Public	
Source -> Destination	SRS130.1	LaneControlSignalControlSchedule	
Realization	Public	Public	
Source -> Destination	SRS41.7	LaneControlSignalControlSchedule	
Realization	Public	Public	
Source -> Destination	SRS41.4	LaneControlSignalControlSchedule	
Realization	Public	Public	
Source -> Destination	SRS41.3	LaneControlSignalControlSchedule	
Realization	Public	Public	
Source -> Destination	SRS41	LaneControlSignalControlSchedule	
Realization	Public	Public	
Source -> Destination	SRS130.1.1	LaneControlSignalControlSchedule	
Realization	Public	Public	
Source -> Destination	SRS41.1	LaneControlSignalControlSchedule	
Realization	Public	Public	
Source -> Destination	SRS41.2	LaneControlSignalControlSchedule	
Realization	Public	Public	
Source -> Destination	SRS129.1	LaneControlSignalControlSchedule	
Realization	Public	Public	
Source -> Destination	SRS128.4	LaneControlSignalControlSchedule	

Operations

Method	Notes	Parameters
addEntity() void Public	Adds a new entity item to the set of entities contained.	String [in] entityId

Method	Notes	Parameters
addEntityElement() void Public	Add the new element to the set of elements that are currently specified for this entity.	String [in] entityId String [in] elementName String [in] elementValue
deleteEntity() void Public	Deletes an entity item from the set of entities contained.	String [in] entityId
getEntityElement() void Public	Get the value of a specified entity.	String [in] entityId String [in] entityElement
getEntityElements() Array Public	Gathers up a list of available entity elements that match all of the provided filters.	List [in] filterList
initialize() void Public	Initialize the entity data to the provided value set.	ByteArray [in] entitydata
updateEntityElement() void Public	Update the value associated with the specified entity element.	String [in] entityId String [in] entityElementName String [in] entityElementValue

5.4.1.2.2.8.2 LaneControlSignalInventory

Type: **Class** *Package:* lcs

This class represents the set of Lane Control Signal Inventory entity information for the center. The information that is maintained is described in TMDD Volume II - Section 3.2.12.3.

Connections

Connector	Source	Target	Notes
Realization	Public	Public	
Source -> Destination	SRS41.1	LaneControlSignalInventory	
Realization	Public	Public	
Source -> Destination	SRS41.7	LaneControlSignalInventory	
Realization	Public	Public	
Source -> Destination	SRS130.2.1	LaneControlSignalInventory	

Connector	Source	Target	Notes
Realization	Public	Public	
Source -> Destination	SRS129.1	LaneControlSignalInventory	
Realization	Public	Public	
Source -> Destination	SRS41.2	LaneControlSignalInventory	
Realization	Public	Public	
Source -> Destination	SRS41.5	LaneControlSignalInventory	
Realization	Public	Public	
Source -> Destination	SRS130.2	LaneControlSignalInventory	
Realization	Public	Public	
Source -> Destination	SRS41.4	LaneControlSignalInventory	
Realization	Public	Public	
Source -> Destination	SRS41.3	LaneControlSignalInventory	
Realization	Public	Public	
Source -> Destination	SRS41	LaneControlSignalInventory	
Realization	Public	Public	
Source -> Destination	SRS128.5	LaneControlSignalInventory	
Realization	Public	Public	
Source -> Destination	SRS41.6	LaneControlSignalInventory	

Operations

Method	Notes	Parameters
addEntity() void Public	Adds a new entity item to the set of entities contained.	String [in] entityId
addEntityElement() void Public	Add the new element to the set of elements that are currently specified for this entity.	String [in] entityId String [in] elementName String [in] elementValue
deleteEntity() void Public	Deletes an entity item from the set of entities contained.	String [in] entityId

Method	Notes	Parameters
getEntityElement() void Public	Get the value of a specified entity.	String [in] entityId String [in] entityElement
getEntityElements() Array Public	Gathers up a list of available entity elements that match all of the provided filters.	List [in] filterList
initialize() void Public	Initialize the entity data to the provided value set.	ByteArray [in] entitydata
updateEntityElement() void Public	Update the value associated with the specified entity element.	String [in] entityId String [in] entityElementName String [in] entityElementValue

5.4.1.2.2.8.3 LaneControlSignalStatus

Type: **Class** *Package:* lcs

This class represents the set of Lane Control Signal Status entity information for the center. The information that is maintained is described in TMDD Volume II - Section 3.2.12.4.

Connections

Connector	Source	Target	Notes
Realization Source -> Destination	Public SRS128.6	Public LaneControlSignalStatus	
Realization Source -> Destination	Public SRS41.7	Public LaneControlSignalStatus	
Realization Source -> Destination	Public SRS41.2	Public LaneControlSignalStatus	
Realization Source -> Destination	Public SRS129.1	Public LaneControlSignalStatus	
Realization Source -> Destination	Public SRS41	Public LaneControlSignalStatus	
Realization Source -> Destination	Public SRS41.6	Public LaneControlSignalStatu	

Connector	Source	Target	Notes
		s	
Realization	Public	Public	
Source -> Destination	SRS41.1	LaneControlSignalStatus	
Realization	Public	Public	
Source -> Destination	SRS128.3.1	LaneControlSignalStatus	
Realization	Public	Public	
Source -> Destination	SRS41.5	LaneControlSignalStatus	
Realization	Public	Public	
Source -> Destination	SRS41.4	LaneControlSignalStatus	
Realization	Public	Public	
Source -> Destination	SRS130.3.1	LaneControlSignalStatus	
Realization	Public	Public	
Source -> Destination	SRS130.3	LaneControlSignalStatus	
Realization	Public	Public	
Source -> Destination	SRS41.3	LaneControlSignalStatus	

Operations

Method	Notes	Parameters
addEntity() void Public	Adds a new entity item to the set of entities contained.	String [in] entityId
addEntityElement() void Public	Add the new element to the set of elements that are currently specified for this entity.	String [in] entityId String [in] elementName String [in] elementValue
deleteEntity() void Public	Deletes an entity item from the set of entities contained.	String [in] entityId
getEntityElement() void Public	Get the value of a specified entity.	String [in] entityId String [in] entityElement
getEntityElements() Array Public	Gathers up a list of available entity elements that match all of the provided filters.	List [in] filterList

Method	Notes	Parameters
initialize() void Public	Initialize the entity data to the provided value set.	ByteArray [in] entitydata
updateEntityElement() void Public	Update the value associated with the specified entity element.	String [in] entityId String [in] entityElementName String [in] entityElementValue

5.4.1.2.2.9 rampmeter

5.4.1.2.2.9.1 RampMeterControlSchedule

Type: **Class** *Package:* rampmeter

This class represents the set of Ramp Meter Control Schedule entity information for the center. The information that is maintained is described in TMDD Volume II - Section 3.2.16.2.

Connections

Connector	Source	Target	Notes
Realization	Public	Public	
Source -> Destination	SRS41.1	RampMeterControlSchedule	
Realization	Public	Public	
Source -> Destination	SRS133.1.1	RampMeterControlSchedule	
Realization	Public	Public	
Source -> Destination	SRS41.3	RampMeterControlSchedule	
Realization	Public	Public	
Source -> Destination	SRS41.5	RampMeterControlSchedule	
Realization	Public	Public	
Source -> Destination	SRS41	RampMeterControlSchedule	
Realization	Public	Public	
Source -> Destination	SRS131.4	RampMeterControlSchedule	
Realization	Public	Public	
Source -> Destination	SRS41.4	RampMeterControlSche	

Connector	Source	Target	Notes
		dule	
Realization	Public	Public	
Source -> Destination	SRS41.7	RampMeterControlSchedule	
Realization	Public	Public	
Source -> Destination	SRS41.2	RampMeterControlSchedule	
Realization	Public	Public	
Source -> Destination	SRS133.1	RampMeterControlSchedule	
Realization	Public	Public	
Source -> Destination	SRS41.6	RampMeterControlSchedule	
Realization	Public	Public	
Source -> Destination	SRS132.1	RampMeterControlSchedule	

Operations

Method	Notes	Parameters
addEntity() void Public	Adds a new entity item to the set of entities contained.	String [in] entityId
addEntityElement() void Public	Add the new element to the set of elements that are currently specified for this entity.	String [in] entityId String [in] elementName String [in] elementValue
deleteEntity() void Public	Deletes an entity item from the set of entities contained.	String [in] entityId
getEntityElement() void Public	Get the value of a specified entity.	String [in] entityId String [in] entityElement
getEntityElements() Array Public	Gathers up a list of available entity elements that match all of the provided filters.	List [in] filterList
initialize() void Public	Initialize the entity data to the provided value set.	ByteArray [in] entitydata
updateEntityElement() void Public	Update the value associated with the specified entity element.	String [in] entityId String [in] entityElementName

Method	Notes	Parameters
		String [in] entityElementValue

5.4.1.2.2.9.2 RampMeterInventory

Type: **Class** *Package:* rampmeter

This class represents the set of Ramp Meter Inventory entity information for the center. The information that is maintained is described in TMDD Volume II - Section 3.2.16.3.

Connections

Connector	Source	Target	Notes
Realization	Public	Public	
Source -> Destination	SRS41.5	RampMeterInventory	
Realization	Public	Public	
Source -> Destination	SRS133.2	RampMeterInventory	
Realization	Public	Public	
Source -> Destination	SRS132.1	RampMeterInventory	
Realization	Public	Public	
Source -> Destination	SRS41.7	RampMeterInventory	
Realization	Public	Public	
Source -> Destination	SRS41.6	RampMeterInventory	
Realization	Public	Public	
Source -> Destination	SRS41.2	RampMeterInventory	
Realization	Public	Public	
Source -> Destination	SRS41.3	RampMeterInventory	
Realization	Public	Public	
Source -> Destination	SRS131.5	RampMeterInventory	
Realization	Public	Public	
Source -> Destination	SRS41	RampMeterInventory	
Realization	Public	Public	
Source -> Destination	SRS41.4	RampMeterInventory	
Realization	Public	Public	
Source -> Destination	SRS41.1	RampMeterInventory	
Realization	Public	Public	
Source -> Destination	SRS133.2.1	RampMeterInventory	

Operations

Method	Notes	Parameters
addEntity() void Public	Adds a new entity item to the set of entities contained.	String [in] entityId
addEntityElement() void Public	Add the new element to the set of elements that are currently specified for this entity.	String [in] entityId String [in] elementName String [in] elementValue
deleteEntity() void Public	Deletes an entity item from the set of entities contained.	String [in] entityId
getEntityElement() void Public	Get the value of a specified entity.	String [in] entityId String [in] entityElement
getEntityElements() Array Public	Gathers up a list of available entity elements that match all of the provided filters.	List [in] filterList
initialize() void Public	Initialize the entity data to the provided value set.	ByteArray [in] entitydata
updateEntityElement() void Public	Update the value associated with the specified entity element.	String [in] entityId String [in] entityElementName String [in] entityElementValue

5.4.1.2.2.9.3 RampMeterPlanInventory

Type: **Class** *Package:* rampmeter

This class represents the set of Ramp Meter Plan Inventory entity information for the center. The information that is maintained is described in TMDD Volume II - Section 3.2.16.4.

Connections

Connector	Source	Target	Notes
Realization	Public	Public	
Source -> Destination	SRS41.6	RampMeterPlanInventory	
Realization	Public	Public	
Source -> Destination	SRS131.6	RampMeterPlanInventor	

Connector	Source	Target	Notes
		y	
Realization	Public	Public	
Source -> Destination	SRS41.5	RampMeterPlanInventory	
Realization	Public	Public	
Source -> Destination	SRS41.7	RampMeterPlanInventory	
Realization	Public	Public	
Source -> Destination	SRS41.2	RampMeterPlanInventory	
Realization	Public	Public	
Source -> Destination	SRS133.3	RampMeterPlanInventory	
Realization	Public	Public	
Source -> Destination	SRS41	RampMeterPlanInventory	
Realization	Public	Public	
Source -> Destination	SRS41.3	RampMeterPlanInventory	
Realization	Public	Public	
Source -> Destination	SRS41.4	RampMeterPlanInventory	
Realization	Public	Public	
Source -> Destination	SRS41.1	RampMeterPlanInventory	
Realization	Public	Public	
Source -> Destination	SRS132.2	RampMeterPlanInventory	
Realization	Public	Public	
Source -> Destination	SRS133.3.1	RampMeterPlanInventory	

Operations

Method	Notes	Parameters
addEntity() void Public	Adds a new entity item to the set of entities contained.	String [in] entityId
addEntityElement() void Public	Add the new element to the set of elements that are currently specified for this entity.	String [in] entityId String [in] elementName String [in] elementValue

Method	Notes	Parameters
deleteEntity() void Public	Deletes an entity item from the set of entities contained.	String [in] entityId
getEntityElement() void Public	Get the value of a specified entity.	String [in] entityId String [in] entityElement
getEntityElements() Array Public	Gathers up a list of available entity elements that match all of the provided filters.	List [in] filterList
initialize() void Public	Initialize the entity data to the provided value set.	ByteArray [in] entitydata
updateEntityElement() void Public	Update the value associated with the specified entity element.	String [in] entityId String [in] entityElementName String [in] entityElementValue

5.4.1.2.2.9.4 RampMeterStatus

Type: **Class** *Package:* rampmeter

This class represents the set of Ramp Meter Status entity information for the center. The information that is maintained is described in TMDD Volume II - Section 3.2.16.6.

Connections

Connector	Source	Target	Notes
Realization	Public	Public	
Source -> Destination	SRS41.6	RampMeterStatus	
Realization	Public	Public	
Source -> Destination	SRS133.4.1	RampMeterStatus	
Realization	Public	Public	
Source -> Destination	SRS41.3	RampMeterStatus	
Realization	Public	Public	
Source -> Destination	SRS41	RampMeterStatus	
Realization	Public	Public	
Source -> Destination	SRS132.1	RampMeterStatus	
Realization	Public	Public	

Connector	Source	Target	Notes
Source -> Destination	SRS41.4	RampMeterStatus	
Realization	Public	Public	
Source -> Destination	SRS41.1	RampMeterStatus	
Realization	Public	Public	
Source -> Destination	SRS133.4	RampMeterStatus	
Realization	Public	Public	
Source -> Destination	SRS41.5	RampMeterStatus	
Realization	Public	Public	
Source -> Destination	SRS131.3.1	RampMeterStatus	
Realization	Public	Public	
Source -> Destination	SRS131.8	RampMeterStatus	
Realization	Public	Public	
Source -> Destination	SRS41.7	RampMeterStatus	
Realization	Public	Public	
Source -> Destination	SRS41.2	RampMeterStatus	

Operations

Method	Notes	Parameters
addEntity() void Public	Adds a new entity item to the set of entities contained.	String [in] entityId
addEntityElement() void Public	Add the new element to the set of elements that are currently specified for this entity.	String [in] entityId String [in] elementName String [in] elementValue
deleteEntity() void Public	Deletes an entity item from the set of entities contained.	String [in] entityId
getEntityElement() void Public	Get the value of a specified entity.	String [in] entityId String [in] entityElement
getEntityElements() Array Public	Gathers up a list of available entity elements that match all of the provided filters.	List [in] filterList
initialize() void Public	Initialize the entity data to the provided value set.	ByteArray [in] entitydata
updateEntityElement() void	Update the value associated with the	String [in] entityId

Method	Notes	Parameters
Public	specified entity element.	String [in] entityElementName String [in] entityElementValue

5.4.1.2.2.10 signalsection

5.4.1.2.2.10.1 SectionControlSchedule

Type: **Class** *Package:* signalsection

This class represents the set of Section Control Schedule entity information for the center. The information that is maintained is described in TMDD Volume II - Section 3.2.18.3.

Connections

Connector	Source	Target	Notes
Realization	Public	Public	
Source -> Destination	SRS41.3	SectionControlSchedule	
Realization	Public	Public	
Source -> Destination	SRS134.3	SectionControlSchedule	
Realization	Public	Public	
Source -> Destination	SRS41.1	SectionControlSchedule	
Realization	Public	Public	
Source -> Destination	SRS136.1	SectionControlSchedule	
Realization	Public	Public	
Source -> Destination	SRS41.4	SectionControlSchedule	
Realization	Public	Public	
Source -> Destination	SRS135.1	SectionControlSchedule	
Realization	Public	Public	
Source -> Destination	SRS41.2	SectionControlSchedule	
Realization	Public	Public	
Source -> Destination	SRS41.6	SectionControlSchedule	
Realization	Public	Public	
Source -> Destination	SRS136.1.1	SectionControlSchedule	
Realization	Public	Public	
Source -> Destination	SRS41	SectionControlSchedule	
Realization	Public	Public	
Source -> Destination	SRS41.5	SectionControlSchedule	

Connector	Source	Target	Notes
Realization	Public	Public	
Source -> Destination	SRS41.7	SectionControlSchedule	

Operations

Method	Notes	Parameters
addEntity() void Public	Adds a new entity item to the set of entities contained.	String [in] entityId
addEntityElement() void Public	Add the new element to the set of elements that are currently specified for this entity.	String [in] entityId String [in] elementName String [in] elementValue
deleteEntity() void Public	Deletes an entity item from the set of entities contained.	String [in] entityId
getEntityElement() void Public	Get the value of a specified entity.	String [in] entityId String [in] entityElement
getEntityElements() Array Public	Gathers up a list of available entity elements that match all of the provided filters.	List [in] filterList
initialize() void Public	Initialize the entity data to the provided value set.	ByteArray [in] entitydata
updateEntityElement() void Public	Update the value associated with the specified entity element.	String [in] entityId String [in] entityElementName String [in] entityElementValue

5.4.1.2.2.10.2 SectionSignalTimingPatternInventory

Type: **Class** *Package:* signalsection

This class represents the set of Section Signal Timing Pattern Inventory entity information for the center. The information that is maintained is described in TMDD Volume II - Section 3.2.18.7.

Connections

Connector	Source	Target	Notes
Realization	Public	Public SectionSignalTimingPat	

Connector	Source	Target	Notes
Source -> Destination	SRS41.4	ternInventory	
Realization	Public	Public	
Source -> Destination	SRS41.1	SectionSignalTimingPattern Inventory	
Realization	Public	Public	
Source -> Destination	SRS41.6	SectionSignalTimingPattern Inventory	
Realization	Public	Public	
Source -> Destination	SRS136.2.1	SectionSignalTimingPattern Inventory	
Realization	Public	Public	
Source -> Destination	SRS41.7	SectionSignalTimingPattern Inventory	
Realization	Public	Public	
Source -> Destination	SRS134.6	SectionSignalTimingPattern Inventory	
Realization	Public	Public	
Source -> Destination	SRS135.2	SectionSignalTimingPattern Inventory	
Realization	Public	Public	
Source -> Destination	SRS136.2	SectionSignalTimingPattern Inventory	
Realization	Public	Public	
Source -> Destination	SRS41.2	SectionSignalTimingPattern Inventory	
Realization	Public	Public	
Source -> Destination	SRS41.3	SectionSignalTimingPattern Inventory	
Realization	Public	Public	
Source -> Destination	SRS41.5	SectionSignalTimingPattern Inventory	
Realization	Public	Public	
Source -> Destination	SRS41	SectionSignalTimingPattern Inventory	

Operations

Method	Notes	Parameters
addEntity() void Public	Adds a new entity item to the set of entities contained.	String [in] entityId

Method	Notes	Parameters
addEntityElement() void Public	Add the new element to the set of elements that are currently specified for this entity.	String [in] entityId String [in] elementName String [in] elementValue
deleteEntity() void Public	Deletes an entity item from the set of entities contained.	String [in] entityId
getEntityElement() void Public	Get the value of a specified entity.	String [in] entityId String [in] entityElement
getEntityElements() Array Public	Gathers up a list of available entity elements that match all of the provided filters.	List [in] filterList
initialize() void Public	Initialize the entity data to the provided value set.	ByteArray [in] entitydata
updateEntityElement() void Public	Update the value associated with the specified entity element.	String [in] entityId String [in] entityElementName String [in] entityElementValue

5.4.1.2.2.10.3 SectionStatus

Type: **Class** *Package:* signalsection

This class represents the set of Section Status entity information for the center. The information that is maintained is described in TMDD Volume II - Section 3.2.18.6.

Connections

Connector	Source	Target	Notes
Realization	Public	Public	
Source -> Destination	SRS134.7	SectionStatus	
Realization	Public	Public	
Source -> Destination	SRS41.2	SectionStatus	
Realization	Public	Public	
Source -> Destination	SRS134.2.1	SectionStatus	
Realization	Public	Public	
Source -> Destination	SRS41.3	SectionStatus	

Connector	Source	Target	Notes
Realization	Public	Public	
Source -> Destination	SRS135.1	SectionStatus	
Realization	Public	Public	
Source -> Destination	SRS41.1	SectionStatus	
Realization	Public	Public	
Source -> Destination	SRS41.7	SectionStatus	
Realization	Public	Public	
Source -> Destination	SRS41.4	SectionStatus	
Realization	Public	Public	
Source -> Destination	SRS41.5	SectionStatus	
Realization	Public	Public	
Source -> Destination	SRS136.3.1	SectionStatus	
Realization	Public	Public	
Source -> Destination	SRS41	SectionStatus	
Realization	Public	Public	
Source -> Destination	SRS41.6	SectionStatus	
Realization	Public	Public	
Source -> Destination	SRS136.3	SectionStatus	

Operations

Method	Notes	Parameters
addEntity() void Public	Adds a new entity item to the set of entities contained.	String [in] entityId
addEntityElement() void Public	Add the new element to the set of elements that are currently specified for this entity.	String [in] entityId String [in] elementName String [in] elementValue
deleteEntity() void Public	Deletes an entity item from the set of entities contained.	String [in] entityId
getEntityElement() void Public	Get the value of a specified entity.	String [in] entityId String [in] entityElement
getEntityElements() Array Public	Gathers up a list of available entity elements that match all of the provided filters.	List [in] filterList

Method	Notes	Parameters
initialize() void Public	Initialize the entity data to the provided value set.	ByteArray [in] entitydata
updateEntityElement() void Public	Update the value associated with the specified entity element.	String [in] entityId String [in] entityElementName String [in] entityElementValue

5.4.1.2.2.11 trafficcontroller

5.4.1.2.2.11.1 IntersectionSignalControllerControlSchedule

Type: **Class** *Package:* trafficcontroller

This class represents the set of Intersection Signal Controller Control Schedule entity information for the center. The information that is maintained is described in TMDD Volume II - Section 3.2.11.3.

Connections

Connector	Source	Target	Notes
Realization	Public	Public	
Source -> Destination	SRS41.2	IntersectionSignalControllerControlSchedule	
Realization	Public	Public	
Source -> Destination	SRS41	IntersectionSignalControllerControlSchedule	
Realization	Public	Public	
Source -> Destination	SRS139.1	IntersectionSignalControllerControlSchedule	
Realization	Public	Public	
Source -> Destination	SRS139.1.1	IntersectionSignalControllerControlSchedule	
Realization	Public	Public	
Source -> Destination	SRS41.6	IntersectionSignalControllerControlSchedule	
Realization	Public	Public	
Source -> Destination	SRS41.7	IntersectionSignalControllerControlSchedule	
Realization	Public	Public	
Source -> Destination	SRS41.5	IntersectionSignalControllerControlSchedule	

Connector	Source	Target	Notes
Realization	Public	Public	
Source -> Destination	SRS137.4	IntersectionSignalControllerControlSchedule	
Realization	Public	Public	
Source -> Destination	SRS41.4	IntersectionSignalControllerControlSchedule	
Realization	Public	Public	
Source -> Destination	SRS41.3	IntersectionSignalControllerControlSchedule	
Realization	Public	Public	
Source -> Destination	SRS41.1	IntersectionSignalControllerControlSchedule	
Realization	Public	Public	
Source -> Destination	SRS138.1	IntersectionSignalControllerControlSchedule	

Operations

Method	Notes	Parameters
addEntity() void Public	Adds a new entity item to the set of entities contained.	String [in] entityId
addEntityElement() void Public	Add the new element to the set of elements that are currently specified for this entity.	String [in] entityId String [in] elementName String [in] elementValue
deleteEntity() void Public	Deletes an entity item from the set of entities contained.	String [in] entityId
getEntityElement() void Public	Get the value of a specified entity.	String [in] entityId String [in] entityElement
getEntityElements() Array Public	Gathers up a list of available entity elements that match all of the provided filters.	List [in] filterList
initialize() void Public	Initialize the entity data to the provided value set.	ByteArray [in] entitydata
updateEntityElement() void Public	Update the value associated with the specified entity element.	String [in] entityId String [in] entityElementName String [in] entityElementValue

Method	Notes	Parameters

5.4.1.2.2.11.2 IntersectionSignalControllerInventory

Type: **Class** *Package:* trafficcontroller

This class represents the set of IntersectionSignal Controller Inventory entity information for the center. The information that is maintained is described in TMDD Volume II - Section 3.2.11.4.

Connections

Connector	Source	Target	Notes
Realization	Public	Public	
Source -> Destination	SRS139.2	IntersectionSignalControllerInventory	
Realization	Public	Public	
Source -> Destination	SRS41.7	IntersectionSignalControllerInventory	
Realization	Public	Public	
Source -> Destination	SRS137.5	IntersectionSignalControllerInventory	
Realization	Public	Public	
Source -> Destination	SRS139.2.1	IntersectionSignalControllerInventory	
Realization	Public	Public	
Source -> Destination	SRS138.1	IntersectionSignalControllerInventory	
Realization	Public	Public	
Source -> Destination	SRS41.1	IntersectionSignalControllerInventory	
Realization	Public	Public	
Source -> Destination	SRS41.2	IntersectionSignalControllerInventory	
Realization	Public	Public	
Source -> Destination	SRS41.5	IntersectionSignalControllerInventory	
Realization	Public	Public	
Source -> Destination	SRS41.6	IntersectionSignalControllerInventory	
Realization	Public	Public	
Source -> Destination	SRS41.3	IntersectionSignalContr	

Connector	Source	Target	Notes
		ollerInventory	
Realization	Public	Public	
Source -> Destination	SRS41.4	IntersectionSignalControllerInventory	
Realization	Public	Public	
Source -> Destination	SRS41	IntersectionSignalControllerInventory	

Operations

Method	Notes	Parameters
addEntity() void Public	Adds a new entity item to the set of entities contained.	String [in] entityId
addEntityElement() void Public	Add the new element to the set of elements that are currently specified for this entity.	String [in] entityId String [in] elementName String [in] elementValue
deleteEntity() void Public	Deletes an entity item from the set of entities contained.	String [in] entityId
getEntityElement() void Public	Get the value of a specified entity.	String [in] entityId String [in] entityElement
getEntityElements() Array Public	Gathers up a list of available entity elements that match all of the provided filters.	List [in] filterList
initialize() void Public	Initialize the entity data to the provided value set.	ByteArray [in] entitydata
updateEntityElement() void Public	Update the value associated with the specified entity element.	String [in] entityId String [in] entityElementName String [in] entityElementValue

5.4.1.2.2.11.3 IntersectionSignalControllerStatus

Type: **Class** *Package:* trafficcontroller

This class represents the set of IntersectionSignal Controller Status entity information for the center. The information that is maintained is described in TMDD Volume II - Section 3.2.11.6.

Connections

Connector	Source	Target	Notes
Realization	Public	Public	
Source -> Destination	SRS139.3	IntersectionSignalControllerStatus	
Realization	Public	Public	
Source -> Destination	SRS41.6	IntersectionSignalControllerStatus	
Realization	Public	Public	
Source -> Destination	SRS138.1	IntersectionSignalControllerStatus	
Realization	Public	Public	
Source -> Destination	SRS41.4	IntersectionSignalControllerStatus	
Realization	Public	Public	
Source -> Destination	SRS41.2	IntersectionSignalControllerStatus	
Realization	Public	Public	
Source -> Destination	SRS41	IntersectionSignalControllerStatus	
Realization	Public	Public	
Source -> Destination	SRS137.7	IntersectionSignalControllerStatus	
Realization	Public	Public	
Source -> Destination	SRS41.3	IntersectionSignalControllerStatus	
Realization	Public	Public	
Source -> Destination	SRS41.5	IntersectionSignalControllerStatus	
Realization	Public	Public	
Source -> Destination	SRS137.3.1	IntersectionSignalControllerStatus	
Realization	Public	Public	
Source -> Destination	SRS139.3.1	IntersectionSignalControllerStatus	
Realization	Public	Public	
Source -> Destination	SRS41.7	IntersectionSignalControllerStatus	
Realization	Public	Public	
Source -> Destination	SRS41.1	IntersectionSignalContr	

Connector	Source	Target	Notes
		ollerStatus	

Operations

Method	Notes	Parameters
addEntity() void Public	Adds a new entity item to the set of entities contained.	String [in] entityId
addEntityElement() void Public	Add the new element to the set of elements that are currently specified for this entity.	String [in] entityId String [in] elementName String [in] elementValue
deleteEntity() void Public	Deletes an entity item from the set of entities contained.	String [in] entityId
getEntityElement() void Public	Get the value of a specified entity.	String [in] entityId String [in] entityElement
getEntityElements() Array Public	Gathers up a list of available entity elements that match all of the provided filters.	List [in] filterList
initialize() void Public	Initialize the entity data to the provided value set.	ByteArray [in] entitydata
updateEntityElement() void Public	Update the value associated with the specified entity element.	String [in] entityId String [in] entityElementName String [in] entityElementValue

5.4.1.2.2.11.4 IntersectionSignalControllerTimingPatternInventory

Type: **Class** *Package:* trafficcontroller

This class represents the set of IntersectionSignal Controller Timing Pattern Inventory entity information for the center. The information that is maintained is described in TMDD Volume II - Section 3.2.11.7.

Connections

Connector	Source	Target	Notes
Realization	Public	Public	
Source -> Destination	SRS139.4.1	IntersectionSignalContr ollerTimingPatternInven	

Connector	Source	Target	Notes
		tory	
Realization	Public	Public	
Source -> Destination	SRS138.2	IntersectionSignalControllerTimingPatternInventory	
Realization	Public	Public	
Source -> Destination	SRS139.4	IntersectionSignalControllerTimingPatternInventory	
Realization	Public	Public	
Source -> Destination	SRS137.8	IntersectionSignalControllerTimingPatternInventory	

Operations

Method	Notes	Parameters
addEntity() void Public	Adds a new entity item to the set of entities contained.	String [in] entityId
addEntityElement() void Public	Add the new element to the set of elements that are currently specified for this entity.	String [in] entityId String [in] elementName String [in] elementValue
deleteEntity() void Public	Deletes an entity item from the set of entities contained.	String [in] entityId
getEntityElement() void Public	Get the value of a specified entity.	String [in] entityId String [in] entityElement
getEntityElements() Array Public	Gathers up a list of available entity elements that match all of the provided filters.	List [in] filterList
initialize() void Public	Initialize the entity data to the provided value set.	ByteArray [in] entitydata
updateEntityElement() void Public	Update the value associated with the specified entity element.	String [in] entityId String [in] entityElementName String [in] entityElementValue

5.4.1.2.2.12 videoswitch

5.4.1.2.2.12.1 VideoSwitchInventory

Type: **Class** *Package:* videoswitch

This class represents the set of Video Switch Inventory entity information for the center. The information that is maintained is described in TMDD Volume II - Section 3.2.20.2.

Connections

Connector	Source	Target	Notes
Realization	Public	Public	
Source -> Destination	SRS142.1	VideoSwitchInventory	
Realization	Public	Public	
Source -> Destination	SRS142.1.1	VideoSwitchInventory	
Realization	Public	Public	
Source -> Destination	SRS141.1	VideoSwitchInventory	
Realization	Public	Public	
Source -> Destination	SRS140.4	VideoSwitchInventory	

Operations

Method	Notes	Parameters
addEntity() void Public	Adds a new entity item to the set of entities contained.	String [in] entityId
addEntityElement() void Public	Add the new element to the set of elements that are currently specified for this entity.	String [in] entityId String [in] elementName String [in] elementValue
deleteEntity() void Public	Deletes an entity item from the set of entities contained.	String [in] entityId
getEntityElement() void Public	Get the value of a specified entity.	String [in] entityId String [in] entityElement
getEntityElements() Array Public	Gathers up a list of available entity elements that match all of the provided filters.	List [in] filterList
initialize() void Public	Initialize the entity data to the provided value set.	ByteArray [in] entitydata
updateEntityElement() void	Update the value associated with the	String [in] entityId

Method	Notes	Parameters
Public	specified entity element.	String [in] entityElementName String [in] entityElementValue

5.4.1.2.2.12.2 VideoSwitchStatus

Type: **Class** *Package:* videoswitch

This class represents the set of Video Switch Status entity information for the center. The information that is maintained is described in TMDD Volume II - Section 3.2.20.3.

Connections

Connector	Source	Target	Notes
Realization	Public	Public	
Source -> Destination	SRS141.1	VideoSwitchStatus	
Realization	Public	Public	
Source -> Destination	SRS140.3.1	VideoSwitchStatus	
Realization	Public	Public	
Source -> Destination	SRS140.5	VideoSwitchStatus	
Realization	Public	Public	
Source -> Destination	SRS142.2	VideoSwitchStatus	
Realization	Public	Public	
Source -> Destination	SRS142.2.1	VideoSwitchStatus	

Operations

Method	Notes	Parameters
addEntity() void Public	Adds a new entity item to the set of entities contained.	String [in] entityId
addEntityElement() void Public	Add the new element to the set of elements that are currently specified for this entity.	String [in] entityId String [in] elementName String [in] elementValue
deleteEntity() void Public	Deletes an entity item from the set of entities contained.	String [in] entityId
getEntityElement() void Public	Get the value of a specified entity.	String [in] entityId String [in] entityElement

Method	Notes	Parameters
getEntityElements() Array Public	Gathers up a list of available entity elements that match all of the provided filters.	List [in] filterList
initialize() void Public	Initialize the entity data to the provided value set.	ByteArray [in] entitydata
updateEntityElement() void Public	Update the value associated with the specified entity element.	String [in] entityId String [in] entityElementName String [in] entityElementValue

5.4.1.2.3 Context Diagram

N/A

5.4.1.2.4 Variability Guide

N/A

5.4.1.2.5 Rationale

N/A

5.4.1.3 Parameter

5.4.1.3.1 Primary Presentation

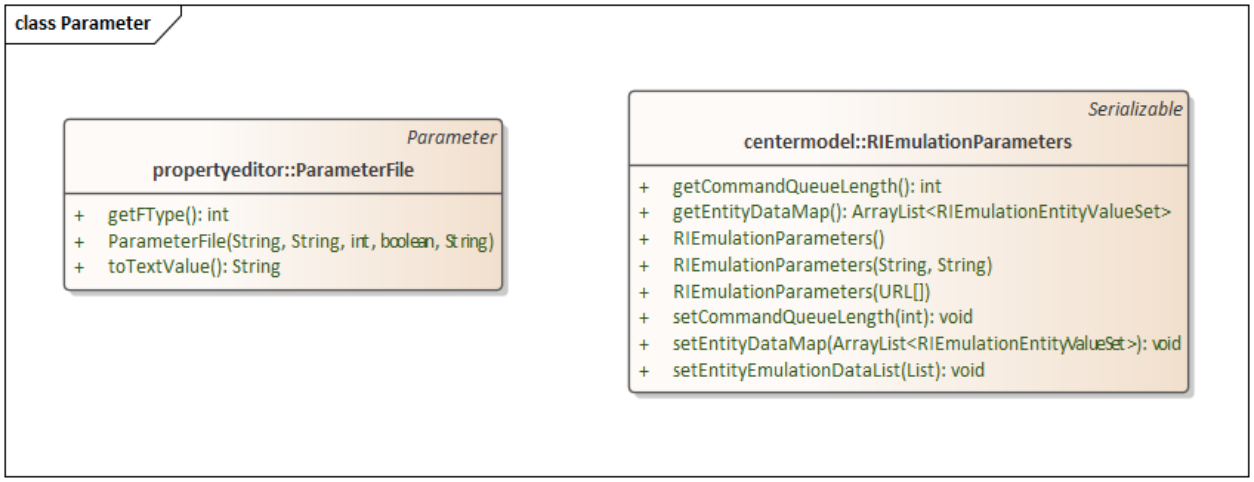

Figure 5-24 Parameter Modules

5.4.1.3.2 Element Catalog

5.4.1.3.2.1 RIEmulationParameters

Type: **Class** *Package:* centermodel

The set of parameters that are specified for entity emulation.

Connections

Connector	Source	Target	Notes
Realization	Public	Public	
Source -> Destination	SRS98	RIEmulationParameters	

Attributes

Attribute	Notes	Constraints and tags
commandQueueLength int Private	The depth of the command queue that will be used during entity emulation.	*Default:*
entityEmulationDataList List Private	A list containing each entity data name and associated data byte array.	*Default:*

5.4.1.3.2.2 ParameterFile

Type: **Class** Parameter *Package:* propertyeditor

The Class ParameterFile.

Connections

Connector	Source	Target	Notes
Generalization	Public	Public	
Source -> Destination	ParameterFile	Parameter	
Association	Public	Package p	
Source -> Destination	EditorPanelFileChooser	ParameterFile	
Realization SRS26 to ParameterFile	Public	Public	
Source -> Destination	SRS26	ParameterFile	

Attributes

Attribute	Notes	Constraints and tags
DIR int Public	The Constant DIR.	*Default:* JFileChooser.DIRECTORIES_ONLY

Attribute	Notes	Constraints and tags
Static Const		
FILE int Public	The Constant FILE.	*Default:* JFileChooser.FILES_ONLY
Static Const		
ftype int Package	The ftype.	*Default:*

Operations

Method	Notes	Parameters
getFType() int Public	Gets the f type. @return the f type	
ParameterFile() Public	Instantiates a new parameter file. Pre-Conditions: N/A Post-Conditions: N/A	String [in] name the nameString [in] value the valueint [in] ftype the ftypeboolean [in] editable the editableString [in] doc the doc
toTextValue() String Public	To text value. Pre-Conditions: N/A Post-Conditions: N/A @return the string	

5.4.1.3.3 Context Diagram

N/A

5.4.1.3.4 Variability Guide

N/A

5.4.1.3.5 Rationale

N/A

5.4.1.4 Entity Namer (1104)

The entity namer functionality has not been implemented due to lack of requirements for it. Entity names are currently provided by the user as part of entity data definitions.

5.4.2 Behavioral

5.4.2.1 Connection

The connection functionality has not been implemented due to lack of requirements for it.

5.4.2.2 Authentication

5.4.2.2.1 Primary Presentation

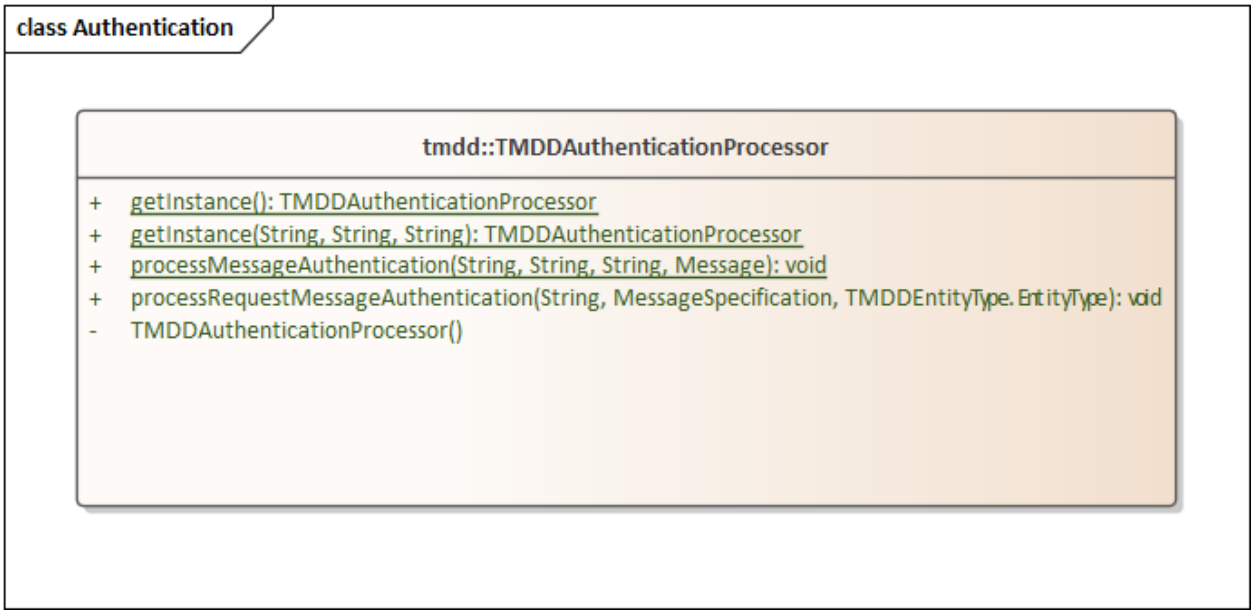

Figure 5-25 Authentication Modules

5.4.2.2.2 Element Catalog

5.4.2.2.2.1 TMDDAuthenticationProcessor

Type: **Class** *Package:* tmddv303c

The Class TMDDAuthenticationProcessor inspects message content to determine whether authentication requirements have been satisfied.

Connections

Connector	Source	Target	Notes
Realization SRS92 to TMDDAuthenticationProcessor Source -> Destination	Public SRS92	Public TMDDAuthenticationProcessor	
Realization SRS8.1.1 to TMDDAuthenticationProc	Public	Public	

Connector	Source	Target	Notes
essor Source -> Destination	SRS8.1.1	TMDDAuthenticationProcessor	

Operations

Method	Notes	Parameters
Static **processMessageAuthentication()** void Public	Process message authentication. Pre-Conditions: N/A Post-Conditions: N/A	String [in] username the usernameString [in] password the passwordString [in] organizationId the organization idMessage [in] requestMessage the request message

5.4.2.2.3 Context Diagram

N/A

5.4.2.2.4 Variability Guide

N/A

5.4.2.2.5 Rationale

N/A

5.4.2.3 Event Model

The Event Model functionality has been implemented within the Entity Manager and Entity elements.

5.5 Message Manager

5.5.1 C2C Message Manager

5.5.1.1 Primary Presentation

Figure 5-26 C2C Message Manager Modules

5.5.1.2 Element Catalog

5.5.1.2.1 Message

Type: **Class** *Package:* messagemanager

The Class Message represents a message that is exchanged between centers.

Connections

Connector	Source	Target	Notes
Association Source -> Destination	Public Message	Private messageBody TemporaryMessageFile	
Association Source -> Destination	Public Message	Private enumeratedMessageType MESSAGETYPE	
Association Source -> Destination	Public Message	Private messageSpecification MessageSpecification	
Realization SRS37.4 to Message Source -> Destination	Public SRS37.4	Public Message	
Realization SRS37.4.1 to Message Source -> Destination	Public SRS37.4.1	Public Message	
Realization SRS37.4.2 to Message Source -> Destination	Public SRS37.4.2	Public Message	
Realization SRS37.3 to Message Source -> Destination	Public SRS37.3	Public Message	
Realization SRS37.4.3 to Message Source -> Destination	Public SRS37.4.3	Public Message	

Attributes

Attribute	Notes	Constraints and tags
ASN1_MESSAGE_ENCODING String Public Static	The AS n1_ messag e_ encoding.	*Default:* "ASN1"

Attribute	Notes	Constraints and tags
encodedMessageType String Private	The encoded message type.	*Default:* "Unknown"
enumeratedMessageType MESSAGETYPE Private	The enumerated message type.	*Default:*
FTP_TRANSPORT_PROTOCOL String Public Static	The ftp transport protocol.	*Default:* "FTP"
GZIPPED_XML_MESSAGE_ENCODING String Public Static	The gzipped xml message encoding.	*Default:* "GZipXML"
HTTP_TRANSPORT_PROTOCOL String Public Static	The http transport protocol.	*Default:* "HTTP"
httpHeaders HashMap<String, String> Private	The http headers.	*Default:*
HTTPS_TRANSPORT_PROTOCOL String Public Static	The https transport protocol.	*Default:* "HTTPS"
messageBody TemporaryMessageFile Private	The message body.	*Default:*
messageBodyParts ArrayList<TemporaryMessageFile> Private	The message body parts.	*Default:* new ArrayList<>()
messageDestinationAddress String Private	The message destination address.	*Default:* ""

Attribute	Notes	Constraints and tags
messageEncoding String Private	The message encoding.	*Default:* ""
messageName String Private	The message name.	*Default:* ""
messageSourceAddress String Private	The message source address.	*Default:* ""
messageSpecification MessageSpecification Private	The message specification.	*Default:*
messageType String Private	The message type.	*Default:* ""
parentDialog String Private	The parent dialog.	*Default:* ""
relatedCommand String Private	GET (RETRV), POST	*Default:* ""
REQUEST_MESSAGE_TYPE String Public Static	The request message type.	*Default:* "Request"
RESPONSE_MESSAGE_TYPE String Public Static	The response message type.	*Default:* "Response"
skipApplicationLayerEncoding boolean Private	The skip application layer encoding.	*Default:*
SOAP_XML_MESSAGE_ENCODING String	The soap xml message encoding.	*Default:* "SOAP_XML"

Attribute	Notes	Constraints and tags
Public Static		
soapHeaders HashMap<String, String> Private	The soap headers.	*Default:*
storageTime Date Private	The storage time.	*Default:*
transportTimeInMillis long Private	The transport time in millis.	*Default:* -1
viaTransportProtocol String Private	The via transport protocol.	*Default:* ""
XML_MESSAGE_ENCODING String Public Static	The xml message encoding.	*Default:* "XML"

Operations

Method	Notes	Parameters
getEncodedMessageType() String Public	Gets the encoded message type. @return the encoded message type	
getEnumeratedMessageType() MESSAGETYPE Public	Gets the enumerated message type. @return the enumerated message type	
getHttpHeaders() HashMap<String, String> Public	Gets the http headers. @return the http headers	
getMessageBody() byte Public	Gets the message body. @return the message body	

Method	Notes	Parameters
getMessageBodyParts() ArrayList<byte[]> Public	Gets the message body parts. @return the message body parts	
getMessageDestinationAddress() String Public	Gets the message destination address. @return the message destination address	
getMessageEncoding() String Public	Gets the message encoding. @return the message encoding	
getMessageName() String Public	Gets the message name. @return the message name	
getMessageSourceAddress() String Public	Gets the message source address. @return the message source address	
getMessageSpecification() MessageSpecification Public	Gets the message specification. @return the message specification	
getMessageType() String Public	Gets the message type. @return the message type	
getParentDialog() String Public	Gets the parent dialog. @return the parent dialog	
getRelatedCommand() String Public	Gets the related command. @return the related command	

Method	Notes	Parameters
getSoapHeaders() HashMap<String, String> Public	Gets the soap headers. @return the soap headers	
getStorageTime() Date Public	Gets the storage time. @return the storage time	
getTransportTimeInMillis() long Public	Gets the transport time in millis. @return the transport time in millis	
getViaTransportProtocol() String Public	Gets the via transport protocol. @return the via transport protocol	
isSkipApplicationLayerEncoding() boolean Public	Checks if is skip application layer encoding. Pre-Conditions: N/A Post-Conditions: N/A @return true, if is skip application layer encoding	
Message() Protected	This object should be created by the MessageManager.	String [in] parentDialog the parent dialogString [in] messageType the message typeString [in] messageName the message nameHashMap [in] httpHeaders the http headersHashMap [in] soapHeaders the soap headersbyte[] [in] messageBody the message bodyString [in] messageEncoding the message encodingString [in] messageSourceAddress the message source addressString

Method	Notes	Parameters
		[in] messageDestinationAddress the message destination addressString [in] viaTransportProtocol the via transport protocolString [in] relatedCommand the related commandDate [in] storageTime the storage time
Message() Protected	This object should be created by the MessageManager.	String [in] parentDialog the parent dialog
Message() Private	Disable the default constructor.	
prettyPrint() String Private	Pretty print. Pre-Conditions: N/A Post-Conditions: N/A @return the string	String [in] xmlInput the xml input
setEncodedMessageType() void Public	Sets the encoded message type.	String [in] encodedMessageType the new encoded message type
setEnumeratedMessageType() void Public	Sets the enumerated message type.	MESSAGETYPE [in] enumeratedMessageType the new enumerated message type
setHttpHeaders() void Public	Sets the http headers. Pre-Conditions: N/A Post-Conditions: N/A	HashMap<String, String> [in] httpHeaders the http headers
setMessageBody() void Public	Sets the message body.	byte[] [in] messageBody the new message body
setMessageBodyParts() void Public	Sets the message body parts.	ArrayList<byte[]> [in] messageBodyParts the new message body parts
setMessageDestinationAddress() void Public	Sets the message destination address.	String [in] messageDestinationAddress the new message destination address
setMessageEncoding() void Public	Sets the message encoding.	String [in] messageEncoding the new message encoding
setMessageName() void Public	Sets the message name.	String [in] messageName the new message name

Method	Notes	Parameters
setMessageSourceAddress() void Public	Sets the message source address.	String [in] messageSourceAddress the new message source address
setMessageSpecification() void Public	Sets the message specification.	MessageSpecification [in] messageSpecification the new message specification
setMessageType() void Public	Sets the message type.	String [in] messageType the new message type
setParentDialog() void Public	Sets the parent dialog.	String [in] parentDialog the new parent dialog
setRelatedCommand() void Public	Sets the related command.	String [in] relatedCommand the new related command
setSkipApplicationLayerEncoding() void Public	Sets the skip application layer encoding.	boolean [in] skipApplicationLayerEncoding the new skip application layer encoding
setSoapHeaders() void Public	Sets the soap headers. Pre-Conditions: N/A Post-Conditions: N/A	HashMap<String, String> [in] soapHeaders the soap headers
setStorageTime() void Public	Sets the storage time.	Date [in] storageTime the new storage time
setTransportTimeInMillis() void Public	Sets the transport time in millis.	long [in] transportTimeInMillis the new transport time in millis
setViaTransportProtocol() void Public	Sets the via transport protocol.	String [in] viaTransportProtocol the new via transport protocol
stripInvalidCharacters() String Private	Strip invalid characters. Pre-Conditions: N/A Post-Conditions: N/A @return the string	byte[] [in] message the message
toXML() String Public	To xml. Pre-Conditions: N/A Post-Conditions: N/A @return the string	

5.5.1.3 Context Diagram

N/A

5.5.1.4 Variability Guide

N/A

5.5.1.5 Rationale

N/A

5.5.2 Validator

The Validator functionality has been implemented within the application layer standard and information layer standard functions.

5.5.3 Parser

The parser functionality has been implemented within the application layer standard and information layer standard functions.

5.5.4 Distributer

The Distributer functionality has not been implemented due to lack of requirements for it.

5.5.5 Creator

5.5.5.1 Primary Presentation

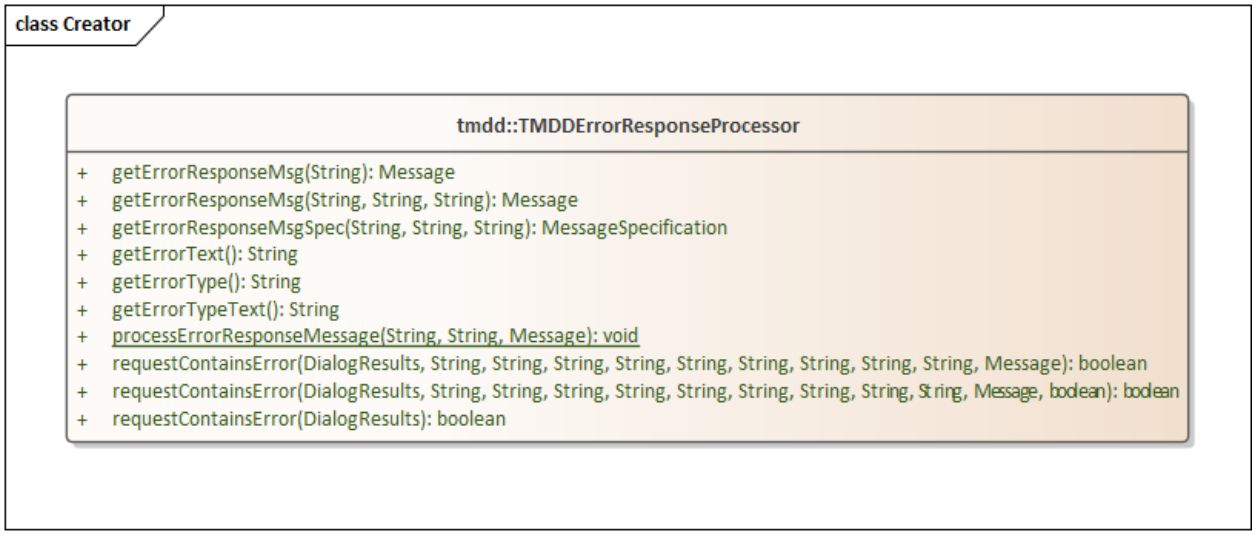

Figure 5-27 Creator Modules

5.5.5.2 Element Catalog

5.5.5.2.1 TMDDErrorResponseProcessor

Type: **Class** *Package:* tmddv303c

The Class TMDDErrorResponseProcessor generates error responses as defined within TMDD and inspects error response messages received from other centers.

Connections

Connector	Source	Target	Notes
Realization SRS92 to TMDDErrorResponseProcessor Source -> Destination	Public SRS92	Public TMDDErrorResponseProcessor	
Realization SRS8.1.1 to TMDDErrorResponseProcessor Source -> Destination	Public SRS8.1.1	Public TMDDErrorResponseProcessor	

Attributes

Attribute	Notes	Constraints and tags
ERRORREPORTCODE String Private Static	The errorreportcode.	*Default:* "tmdd:errorReportMsg.error-code = #ERRORCODE#"
ERRORREPORTTEXT String Private Static	The errorreporttext.	*Default:* "tmdd:errorReportMsg.error-text = #ERRORTEXT#"
errorText String Private	The error text.	*Default:* ""
errorType String Private	The error type.	*Default:* "0"
errorTypeText String Private	The error type text.	*Default:* ""
ORGANIZATIONINFORMATION String	The organizationinformation.	*Default:* "tmdd:errorReportMsg.organization-

Attribute	Notes	Constraints and tags
Private Static		information.organization-id = C2C RI"
ORGANIZATIONREQUESTING String Private Static	The organizationrequesting.	*Default:* "tmdd:errorReportMsg.organization-requesting.organization-id = SUT"
RESTRICTIONS String Private Static	The restrictions.	*Default:* "tmdd:errorReportMsg.restrictions.organization-information-forwarding-restrictions = 1"

Operations

Method	Notes	Parameters
getErrorResponseMsg() Message Public	Gets the error response msg. @return the error response msg	String [in] messageName the message name
getErrorResponseMsg() Message Public	Gets the error response msg. @return the error response msg	String [in] messageName the message nameString [in] errorCode the error codeString [in] errorText the error text
getErrorText() String Public	Gets the error text. @return the error text	
getErrorType() String Public	Gets the error type. @return the error type	
getErrorTypeText() String Public	Gets the error type text. @return the error type text	
Static **processErrorResponseMessage()** void	Process error response message. Pre-Conditions: N/A Post-Conditions: N/A	String [in] errorResponseExpected the error response expectedString [in] errorTypeExpected

Method	Notes	Parameters
Public		the error type expectedMessage [in] responseMessage the response message
requestContainsError() boolean Public	Request contains error. Pre-Conditions: N/A Post-Conditions: N/A @return true, if successful	DialogResults [in] dialogResults the dialog resultsString [in] authenticationCheck the authentication checkString [in] allowAccess the allow accessString [in] errorResponseExpected the error response expectedString [in] errorTypeExpected the error type expectedString [in] dataAvailable the data availableString [in] userName the user nameString [in] password the passwordString [in] organizationId the organization idString [in] testCaseID the test case idMessage [in] requestMessage the request message
requestContainsError() boolean Public	Request contains error. Pre-Conditions: N/A Post-Conditions: N/A @return true, if successful	DialogResults [in] dialogResults the dialog resultsString [in] authenticationCheck the authentication checkString [in] allowAccess the allow accessString [in] errorResponseExpected the error response expectedString [in] errorTypeExpected the error type expectedString [in] dataAvailable the data availableString [in] userName the user nameString [in] password the passwordString [in] organizationId the organization idString [in] testCaseID

Method	Notes	Parameters
		the test case idMessage [in] requestMessage the request messageboolean [in] periodicUpdateError the periodic update error

5.5.5.3 Context Diagram

N/A

5.5.5.4 Variability Guide

N/A

5.5.5.5 Rationale

N/A

5.6 Logger

5.6.1 Log Manager

5.6.1.1 Primary Presentation

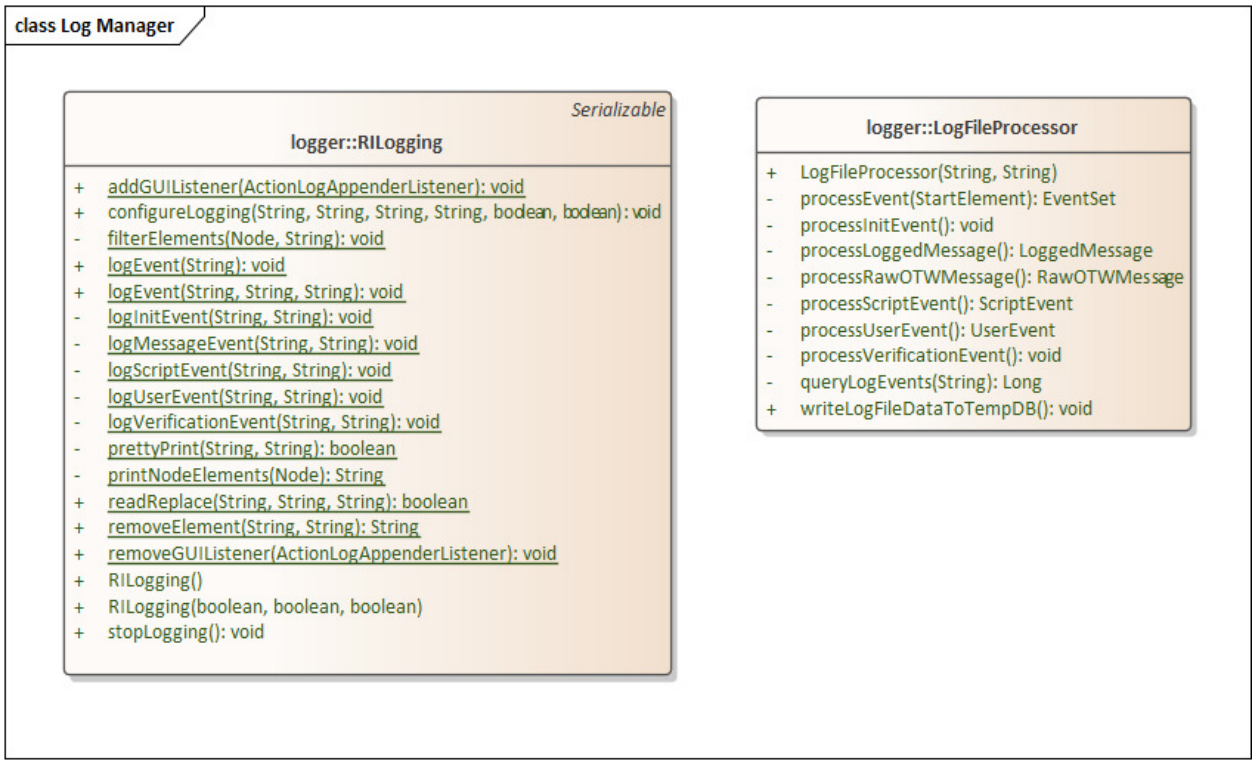

Figure 5-28 Log Manager Modules

5.6.1.2 Element Catalog

5.6.1.2.1 RILogging

Type: **Class** Serializable *Package:* logger

Maintains the RI logging settings to be used during RI operation and testing. It also provides for log4j configuration for logging to the RI log file.

Connections

Connector	Source	Target	Notes
Realization SRS32.2 to RILogging Source -> Destination	Public SRS32.2	Public RILogging	
Realization SRS37 to	Public	Public	

Connector	Source	Target	Notes
RILogging Source -> Destination	SRS37	RILogging	
Realization SRS32.5 to RILogging Source -> Destination	Public SRS32.5	Public RILogging	
Realization SRS32.1 to RILogging Source -> Destination	Public SRS32.1	Public RILogging	
Realization SRS32.4 to RILogging Source -> Destination	Public SRS32.4	Public RILogging	
Realization SRS32.6 to RILogging Source -> Destination	Public SRS32.6	Public RILogging	
Realization SRS32.3 to RILogging Source -> Destination	Public SRS32.3	Public RILogging	
Association Source -> Destination	Public RITestEngine	Private logger RILogging	

Attributes

Attribute	Notes	Constraints and tags
log Logger Private	The log2.	*Default:*
log2 Logger Private	The log2.	*Default:*
logFile String Private	The log file.	*Default:*
RI_INIT_EVENT String Public Static	The ri init event.	*Default:* "InitEvent"
RI_MESSAGE_EVENT String Public Static	The ri message event.	*Default:* "MessageEvent"

Attribute	Notes	Constraints and tags
RI_SCRIPT_EVENT String Public Static	The ri script event.	*Default:* "ScriptEvent"
RI_USER_EVENT String Public Static	The ri user event.	*Default:* "UserEvent"
RI_VERIFICATION_EVENT String Public Static	The ri verification event.	*Default:* "VerificationEvent"
riAppender FileAppender Private	The ri appender.	*Default:*

Operations

Method	Notes	Parameters
configureLogging() void Public	Sets up log4j logging parameters for the test. log4j can be set to log various test levels. Based on the logging parameters these levels will be set appropriately. Next this method, logs the current Test Configuration in the Test Log logs the name of the Test Operator in the Test Log logs the unique test name in the Test Log logs the current date and time in the Test Log logs any descriptive test notes in the Test Log	String [in] logFileName the log file nameString [in] configFileName the config file nameString [in] logDescription the log descriptionString [in] checkSum the check sum
Static **filterElements()** void Private	Filter elements. Pre-Conditions: N/A Post-Conditions: N/A	Node [in] parent the parentString [in] filter the filter
Static **logEvent()** void Public	Log event. Pre-Conditions: N/A Post-Conditions: N/A	String [in] logName the log nameString [in] eventType the event typeString [in] theEvent the the event
Static **logEvent()** void Public	Log event. Pre-Conditions: N/A Post-Conditions: N/A	String [in] theEvent the the event
Static **logInitEvent()** void Private	Log init event. Pre-Conditions: N/A Post-Conditions: N/A	String [in] logName the log nameString [in] theEvent the the event

Method	Notes	Parameters
Static **logMessageEvent()** void Private	Log message event. Pre-Conditions: N/A Post-Conditions: N/A	String [in] logName the log nameString [in] theEvent the the event
Static **logScriptEvent()** void Private	Log script event. Pre-Conditions: N/A Post-Conditions: N/A	String [in] logName the log nameString [in] theEvent the the event
Static **logUserEvent()** void Private	Log user event. Pre-Conditions: N/A Post-Conditions: N/A	String [in] logName the log nameString [in] theEvent the the event
Static **logVerificationEvent()** void Private	Log verification event. Pre-Conditions: N/A Post-Conditions: N/A	String [in] logName the log nameString [in] theEvent the the event
Static **prettyPrint()** void Private	Pretty print. Pre-Conditions: N/A Post-Conditions: N/A	String [in] xmlInput the xml inputString [in] xmlOutputFile the xml output file
Static **printNodeElements()** String Private	Prints the node elements. Pre-Conditions: N/A Post-Conditions: N/A @return the string	Node [in] node the node
Static **readReplace()** void Public	Replaces the log4j: namespace references from the log file and creates a new version which is a well formed XML Document.	String [in] fname the fnameString [in] oldPattern the old patternString [in] replPattern the repl pattern
Static **removeElement()** String Public	Removes the element. Pre-Conditions: N/A Post-Conditions: N/A @return the string	String [in] inputXML the input xmlString [in] targetElement the target element
RILogging() Public	Instantiates a new rI logging. Pre-Conditions: N/A Post-Conditions: N/A	boolean [in] messagesOnly the messages onlyboolean [in] failuresOnly the failures onlyboolean [in] actionsAndMessages the actions and messages
RILogging()	Instantiates a new rI logging. Pre-Conditions: N/A Post-Conditions: N/A	

Method	Notes	Parameters
Public		
stopLogging() void Public	Stop logging. Pre-Conditions: N/A Post-Conditions: N/A	

5.6.1.2.2 LogFileProcessor

Type: **Class** *Package:* logger

The Class LogFileProcessor processes the contents of a RI Test Log file and places the contents into the c2cri database for report generation.

Connections

Connector	Source	Target	Notes
Association Source -> Destination	Public LogFileProcessor	Private initEvent InitEvent	
Association Source -> Destination	Public LogFileProcessor	Private scriptEventDAO ScriptEventDAO	
Association Source -> Destination	Public LogFileProcessor	Private rawOTWMessageDAO RawOTWMessageDAO	
Realization SRS50 to LogFileProcessor Source -> Destination	Public SRS50	Public LogFileProcessor	

Attributes

Attribute	Notes	Constraints and tags
APPLICATIONLEVELSTANDARD String Private Static	The applicationlevelstandard.	*Default:* "applicationLevelStandard"
C2CRIVERSION String Private Static	The C2cri version tag.	*Default:* "c2criVersion"
CHECKSUM String Private Static	The checksum tag.	*Default:* "checksum"
COLUMN String	The column tag.	*Default:* "column"

Attribute	Notes	Constraints and tags
Private		
Static		
CONFIGFILENAME String	The configuration tag.	*Default:* "configFileName"
Private		
Static		
CONFIGURATIONAUTHOR String	The configurationauthor.	*Default:* "configurationAuthor"
Private		
Static		
CONFIGVERSION String	The configversion.	*Default:* "configVersion"
Private		
Static		
CONFORMANCE String	The conformance.	*Default:* "conformance"
Private		
Static		
conn Connection	The conn.	*Default:* null
Private		
CONNECTIONNAME String	The connectionname tag.	*Default:* "connectionName"
Private		
Static		
CREATOR String	The creator tag.	*Default:* "creator"
Private		
Static		
DESCRIPTION String	The description tag.	*Default:* "description"
Private		
Static		
DESTINATIONADDRESS String	The destinationaddress tag.	*Default:* "DestinationAddress"
Private		
Static		
ERRORDESCRIPTION String	The errordescription.	*Default:* "errorDescription"
Private		
Static		
event XMLEvent	The event.	*Default:* null
Private		

Attribute	Notes	Constraints and tags
EVENT String Private Static	The event tag.	*Default:* "event"
eventId int Private	The event id.	*Default:* 1
eventReader XMLEventReader Private	The event reader.	*Default:* null
EXECUTIONTIME String Private Static	The executiontime tag.	*Default:* "execution-time"
EXECUTIONTIMEMILLIS String Private Static	The executiontimemillis tag.	*Default:* "execution-time-millis"
EXTENSION String Private Static	The extension.	*Default:* "extension"
EXTERNALCENTEROPERATION String Private Static	The externalcenteroperation.	*Default:* "externalCenterOperation"
FILE String Private Static	The file tag.	*Default:* "file"
FILENAME String Private Static	The filename tag.	*Default:* "fileName"
FUNCTIONID String Private Static	The functionid tag.	*Default:* "functionId"
INFORMATIONLEVELSTANDARD String Private	The informationlevelstandard.	*Default:* "informationLevelStandard"

Attribute	Notes	Constraints and tags
Static		
initEvent InitEvent	The init event.	*Default:* new InitEvent()
Private		
INITEVENT String	The initevent tag.	*Default:* "initEvent"
Private		
Static		
LEVEL String	The level attribute.	*Default:* "level"
Private		
Static		
LINE String	The line tag.	*Default:* "line"
Private		
Static		
logFile File	The log file.	*Default:*
Private		
LOGGEDMESSAGE String	The loggedmessage tag.	*Default:* "loggedMessage"
Private		
Static		
LOGGER String	The logger attribute.	*Default:* "logger"
Private		
Static		
MESSAGE String	The message tag.	*Default:* "message"
Private		
Static		
MESSAGEBODY String	The messagebody tag.	*Default:* "MessageBody"
Private		
Static		
MESSAGEDESTINATIONADDRESS String	The messagedestinationaddress tag.	*Default:* "messageDestinationAddress"
Private		
Static		
MESSAGEENCODING String	The messageencoding tag.	*Default:* "messageEncoding"
Private		
Static		

Attribute	Notes	Constraints and tags
MESSAGENAME String Private Static	The messagename tag.	*Default:* "messageName"
MESSAGESOURCEADDRESS String Private Static	The messagesourceaddress tag.	*Default:* "messageSourceAddress"
MESSAGETYPE String Private Static	The messagetype tag.	*Default:* "messageType"
NEED String Private Static	The need.	*Default:* "need"
OTHERREQ String Private Static	The otherreq.	*Default:* "otherReq"
OUTCOME String Private Static	The outcome tag.	*Default:* "outcome"
outdb File Private	The output database file.	*Default:*
PARENTDIALOG String Private Static	The parentdialog tag.	*Default:* "parentDialog"
PROCESSTYPE String Private Static	The processtype tag.	*Default:* "ProcessType"
pstmt PreparedStatement Private	The pstmt.	*Default:* null
RAWOTWMESSAGE String Private Static	The rawotwmessage tag.	*Default:* "rawOTWMessage"
rawOTWMessageDAO RawOTWMessageDAO	The raw otw message dao.	*Default:*

Attribute	Notes	Constraints and tags
Private		
RELATEDCOMMAND String Private Static	The relatedcommand tag.	*Default:* "relatedCommand"
REQID String Private Static	The reqid.	*Default:* "reqID"
REQTEXT String Private Static	The reqtext.	*Default:* "reqText"
REQUIREMENT String Private Static	The requirement.	*Default:* "requirement"
RESULT String Private Static	The result.	*Default:* "result"
RESULTS String Private Static	The results.	*Default:* "results"
SCRIPTEVENT String Private Static	The scriptevent tag.	*Default:* "scriptEvent"
scriptEventDAO ScriptEventDAO Private	The script event dao.	*Default:*
scriptEventID int Private	The script event id.	*Default:* 0
SELECTEDAPPLAYERTESTSUITE String Private Static	The selectedapplayertestsuite.	*Default:* "selectedAppLayerTestSuite"
SELECTEDINFOLAYERTESTSUITE String	The selectedinfolayertestsuite.	*Default:* "selectedInfoLayerTestSuite"

Attribute	Notes	Constraints and tags
Private		
Static		
SEQUENCECOUNT String	The sequencecount attribute.	*Default:* "sequenceCount"
Private		
Static		
SOURCEADDRESS String	The sourceaddress tag.	*Default:* "SourceAddress"
Private		
Static		
SRC String	The src tag.	*Default:* "src"
Private		
Static		
STARTTIME String	The starttime tag.	*Default:* "startTime"
Private		
Static		
SUPPORTED String	The supported.	*Default:* "supported"
Private		
Static		
TAG String	The tag tag.	*Default:* "tag"
Private		
Static		
TESTCASE String	The testcase tag.	*Default:* "testCase"
Private		
Static		
TESTCASEID String	The testcaseid.	*Default:* "testCaseID"
Private		
Static		
TESTCASENAME String	The testcasename tag.	*Default:* "test-case-name"
Private		
Static		
TESTRESULTS String	The testresult tags.	*Default:* "testResults"
Private		
Static		
THREAD String	The thread attribute.	*Default:* "thread"
Private		
Static		
TIMESTAMP String	The timestamp attribute.	*Default:* "timestamp"

Attribute	Notes	Constraints and tags
Private		
Static		
TIMESTAMPINMILLIS String	The timestampinmillis attribute.	*Default:* "timestampInMillis"
Private		
Static		
TYPE String	The type tag.	*Default:* "type"
Private		
Static		
UNID String	The unid.	*Default:* "unID"
Private		
Static		
UNSELECTED String	The unselected.	*Default:* "unSelected"
Private		
Static		
USEREVENT String	The userevent tag.	*Default:* "userEvent"
Private		
Static		
userEventList ArrayList<UserEvent>	The user event list.	*Default:* new ArrayList<UserEvent>()
Private		
USERNEED String	The userneed.	*Default:* "userNeed"
Private		
Static		
VERIFICATIONEVENT String	The verificationevent tag.	*Default:* "verificationEvent"
Private		
Static		
VERIFICATIONTESTCASE String	The verificationtestcase.	*Default:* "testCase"
Private		
Static		
VERIFICATIONTIMESTAMP String	The verificationtimestamp.	*Default:* "timeStamp"
Private		
Static		

Attribute	Notes	Constraints and tags
VIATRANSPORTPROTOCOL String Private Static	The viatransportprotocol tag.	*Default:* "viaTransportProtocol"

Operations

Method	Notes	Parameters
LogFileProcessor() Public	Instantiates a new log file processor. Pre-Conditions: N/A Post-Conditions: N/A	String [in] logFileName the log file nameString [in] databaseName the database name
processEvent() EventSet Private	Process event tags. Pre-Conditions: N/A Post-Conditions: N/A @return the event set	StartElement [in] thisEvent the this event
processInitEvent() void Private	Process init event tags. Pre-Conditions: N/A Post-Conditions: N/A	
processLoggedMessage() LoggedMessage Private	Process the loggedmessage tag. Pre-Conditions: N/A Post-Conditions: N/A @return the logged message	
processRawOTWMessage() RawOTWMessage Private	Process rawotwmessage tags. Pre-Conditions: N/A Post-Conditions: N/A @return the raw otw message	
processScriptEvent() ScriptEvent Private	Process script event tags. Pre-Conditions: N/A Post-Conditions: N/A @return the script event	
processUserEvent() UserEvent Private	Process user event tags. Pre-Conditions: N/A Post-Conditions: N/A @return the user event	

Method	Notes	Parameters
processVerificationEvent() void Private	Process verificationevent tags. Pre-Conditions: N/A Post-Conditions: N/A	
queryLogEvents() Long Private	Determines the number of events contained within the log file. This method currently uses xQuery to quickly run through the log file and get a count of event tags. Pre-Conditions: N/A Post-Conditions: N/A @return the long	String [in] logFileName the log file name
writeLogFileDataToTempDB() void Public	Write out log data to temporary SQLite database.	

5.6.1.3 Context Diagram

N/A

5.6.1.4 Variability Guide

N/A

5.6.1.5 Rationale

N/A

5.6.2 Logger

5.6.2.1 Primary Presentation

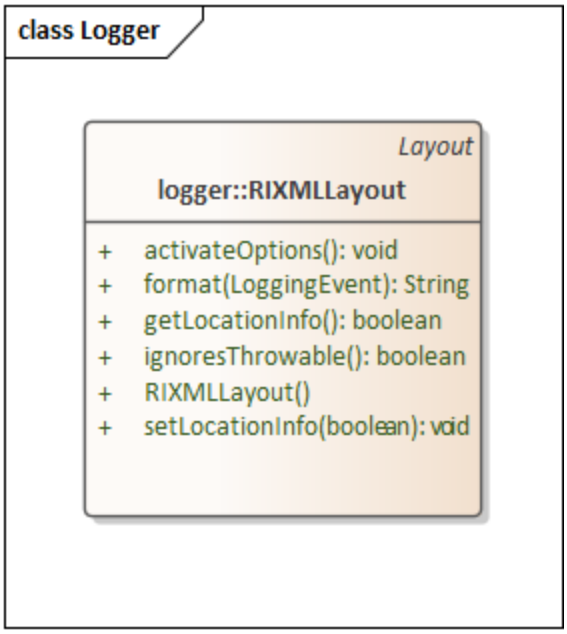

Figure 5-29 Logger Modules

5.6.2.2 Element Catalog

5.6.2.2.1 RIXMLLayout

Type: **Class** Layout *Package:* logger

The output of the XMLLayout consists of a series of log4j:event elements as defined in the log4j.dtd . It does not output a complete well-formed XML file. The output is designed to be included as an external entity in a separate file to form a correct XML file.

For example, if <code>abc</code> is the name of the file where the XMLLayout ouput goes, then a well-formed XML file would be:

<pre> <?xml version="1.0" ?> <!DOCTYPE log4j:eventSet SYSTEM "log4j.dtd" [<!ENTITY data SYSTEM "abc">]> <log4j:eventSet version="1.2" xmlns:log4j="http://jakarta.apache.org/log4j/"> &data </log4j:eventSet>

</pre>

This approach enforces the independence of the XMLLayout and the appender where it is embedded.

The <code>version</code> attribute helps components to correctly intrepret output generated by XMLLayout. The value of this attribute should be "1.1" for output generated by log4j versions prior to log4j 1.2 (final release) and "1.2" for relase 1.2 and later. Contributors: Mathias Bogaert

@since 0.9.0

Connections

Connector	Source	Target	Notes
Realization SRS37.1 to RIXMLLayout Source -> Destination	Public SRS37.1	Public RIXMLLayout	

Attributes

Attribute	Notes	Constraints and tags
locationInfo boolean Private		*Default:* false

Operations

Method	Notes	Parameters
activateOptions() void Public	No options to activate.	
format() String Public	Formats a {@link LoggingEvent}in conformance with the log4j.dtd.	LoggingEvent [in] event
getLocationInfo() boolean Public	Returns the current value of the **LocationInfo** option.	
ignoresThrowable() boolean Public		
RIXMLLayout() Public	Default constructor. @since 1.3	
setLocationInfo() void Public	The **LocationInfo** option takes a boolean value. By default, it is set to false which means there will be no location information output by this layout. If the the option is set to true, then the file name and line number of the statement at the origin of the log statement will be output. If you are embedding this layout within an {@link	boolean [in] flag

Method	Notes	Parameters
	org.apache.log4j.net.SMTPAppender} then make sure to set the **LocationInfo** option of that appender as well.	

5.6.2.3 Context Diagram

N/A

5.6.2.4 Variability Guide

N/A

5.6.2.5 Rationale

N/A

5.6.3 Handler

5.6.3.1 Primary Presentation

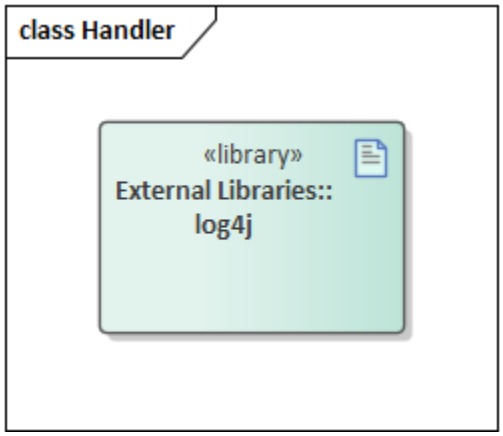

Figure 5-30 Handler Modules

5.6.3.2 Element Catalog

5.6.3.2.1 log4j

Type: **Class** *Package:* Logical View

http://logging.apache.org/log4j/1.2/

Apache log4j™ 1.2

Welcome to Apache log4j, a logging library for Java. Apache log4j is an Apache Software Foundation Project and developed by a dedicated team of Committers of the Apache Software Foundation. For more info, please see The Apache Software Foundation. Apache log4j is also part of a project which is known as Apache Logging. Please see the License.

If you are interested in the recent changes, visit our changes report.

Why logging?

Inserting log statements into your code is a low-tech method for debugging it. It may also be the only way because debuggers are not always available or applicable. This is often the case for distributed applications.

On the other hand, some people argue that log statements pollute source code and decrease legibility. (We believe that the contrary is true). In the Java language where a preprocessor is not available, log statements increase the size of the code and reduce its speed, even when logging is turned off. Given that a reasonably sized application may contain thousands of log statements, speed is of particular importance.

Why log4j?

With log4j it is possible to enable logging at runtime without modifying the application binary. The log4j package is designed so that these statements can remain in shipped code without incurring a heavy performance cost. Logging behavior can be controlled by editing a configuration file, without touching the application binary.

Logging equips the developer with detailed context for application failures. On the other hand, testing provides quality assurance and confidence in the application. Logging and testing should not be confused. They are complementary. When logging is wisely used, it can prove to be an essential tool.

One of the distinctive features of log4j is the notion of inheritance in loggers. Using a logger hierarchy it is possible to control which log statements are output at arbitrarily fine granularity but also great ease. This helps to reduce the volume of logged output and the cost of logging.

The target of the log output can be a file, an OutputStream, a java.io.Writer, a remote log4j server, a remote Unix Syslog daemon, or many other output targets.

Performance

On an AMD Duron clocked at 800Mhz running JDK 1.3.1, it costs about 5 nanoseconds to determine if a logging statement should be logged or not. Actual logging is also quite fast, ranging from 21 microseconds using the SimpleLayout, 37 microseconds using the TTCCLayout. The performance of the PatternLayout is almost as good as the dedicated layouts, except that it is much more flexible.

Connections

Connector	Source	Target	Notes
Realization Source -> Destination	Public SRS37.2	Public log4j	

Connector	Source	Target	Notes
Realization SRS37.2 to log4j Source -> Destination	Public SRS37.2	Public log4j	

5.6.3.3 Context Diagram

N/A

5.6.3.4 Variability Guide

N/A

5.6.3.5 Rationale

N/A

5.6.4 Reader

The Reader functionality has been implemented as part of the Report generation functionality.

5.7 Reporter

5.7.1 Report Manager

5.7.1.1 Primary Presentation

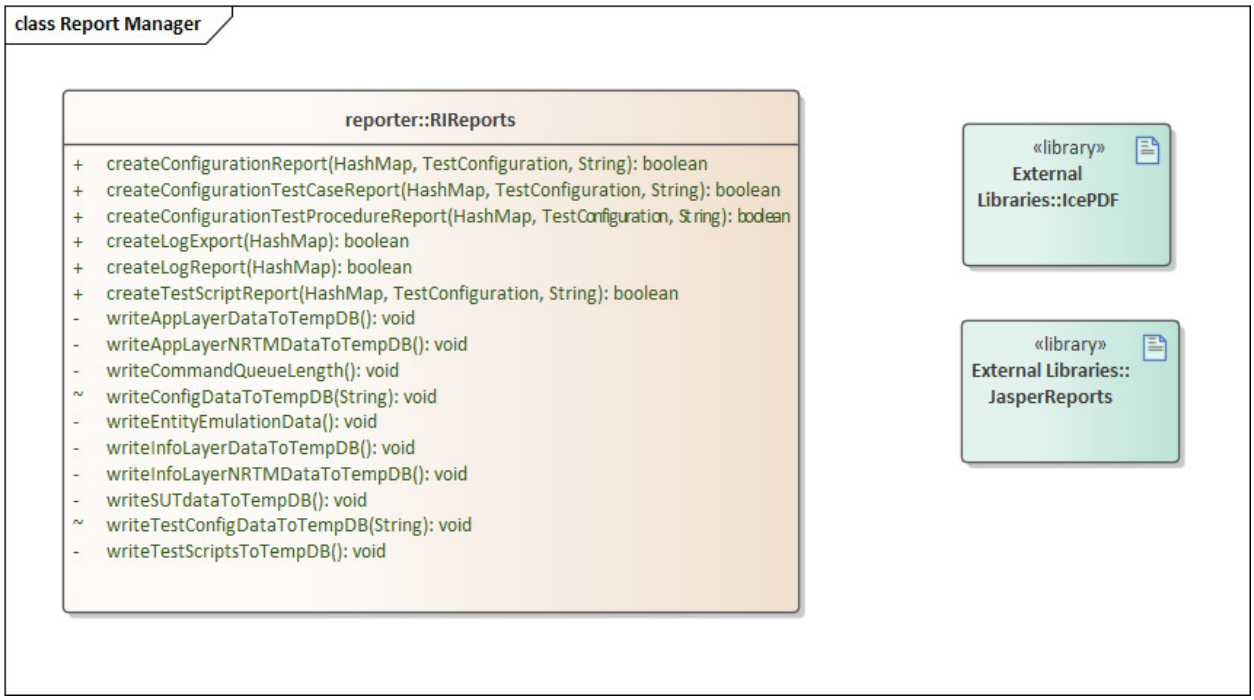

Figure 5-31 Report Manager Modules

5.7.1.2 Element Catalog

5.7.1.2.1 IcePDF

Type: **Class** *Package:* Logical View

http://www.icesoft.org/java/projects/ICEpdf/overview.jsf

ICEpdf is an open source PDF engine for viewing, printing, and annotating PDF documents. The ICEpdf API is 100% Java, lightweight, fast, efficient, and very easy to use.

ICEpdf can be used as standalone open source Java PDF viewer, or can be easily embedded in any Java application to seamlessly load or capture PDF documents. Beyond PDF document rendering, ICEpdf is extremely versatile, and can be used in a multitude of innovative ways, including:

- PDF to image conversion
- PDF annotation
- PDF text and/or image extraction

- PDF search
- PDF printing

Connections

Connector	Source	Target	Notes
Realization Source -> Destination	Public SRS57	Public IcePDF	
Realization SRS57 to IcePDF Source -> Destination	Public SRS57	Public IcePDF	
Realization Source -> Destination	Public SRS56.2	Public IcePDF	
Realization Source -> Destination	Public SRS56	Public IcePDF	
Realization SRS56 to IcePDF Source -> Destination	Public SRS56	Public IcePDF	
Realization Source -> Destination	Public SRS56.1	Public IcePDF	
Realization SRS56.1 to IcePDF Source -> Destination	Public SRS56.1	Public IcePDF	

5.7.1.2.2 JasperReports

Type: **Class** *Package:* Logical View

http://community.jaspersoft.com/project/jasperreports-library

What is JasperReports Library?

JasperReports Library is a Java Library that offers an interface to JasperReports Library reporting engine. This library is the engine at the core of the iReport Designer, Jaspersoft Studio, and JasperReports Server. In addition, it is embedded in thousands of applications around the world.

See a list of JasperReports Library Features.

System Requirements

JasperReports Library has few external dependencies.

All that is required is:

- Java JDK 1.6 or higher
- JDBC 2.1 driver (if RDBMS used)

Installation

JasperReports Library is just a .jar file - installation is completed by adding the jar file to your Java classpath along with other required jar files that support the JasperReports Library. For more information see JasperReports Library - First Steps and

Configuration

Optional Configurations

Most of the configuration of JasperReports Library is done within the JasperReports Library.properties file (for global settings) or at the report level by setting the appropriate settings within the JRXML.

- JasperReports Library Configuration Reference

Using JasperReports Library

It is highly recommended when you first get started with JasperReports Library that you leverage the Desktop Report Designer for JasperReports Library, iReport Designer. Many successfully use XML editors to create and modify the JRXML consumed by JasperReports Library, however it is much more effective in the majority of cases to work within iReport Designer.

Connections

Connector	Source	Target	Notes
Realization Source -> Destination	Public SRS57.3	Public JasperReports	
Realization Source -> Destination	Public SRS59	Public JasperReports	
Realization Source -> Destination	Public SRS57.4	Public JasperReports	
Realization Source -> Destination	Public SRS54	Public JasperReports	
Realization Source -> Destination	Public SRS57.1	Public JasperReports	
Realization SRS57.1 to JasperReports Source -> Destination	Public SRS57.1	Public JasperReports	

Connector	Source	Target	Notes
Realization SRS57.3 to JasperReports Source -> Destination	Public SRS57.3	Public JasperReports	
Realization SRS54 to JasperReports Source -> Destination	Public SRS54	Public JasperReports	
Realization SRS59 to JasperReports Source -> Destination	Public SRS59	Public JasperReports	
Realization SRS57.4 to JasperReports Source -> Destination	Public SRS57.4	Public JasperReports	
Realization SRS49 to JasperReports Source -> Destination	Public SRS49	Public JasperReports	
Realization Source -> Destination	Public SRS57.2	Public JasperReports	
Realization Source -> Destination	Public SRS49	Public JasperReports	
Realization Source -> Destination	Public SRS50	Public JasperReports	
Realization SRS57.2 to JasperReports Source -> Destination	Public SRS57.2	Public JasperReports	
Realization SRS50 to JasperReports Source -> Destination	Public SRS50	Public JasperReports	

5.7.1.2.3 RIReports

Type: **Class** *Package:* reporter

This class provides the RI's interface with JasperReports and initiates/manages report generation.

Connections

Connector	Source	Target	Notes
Association Source -> Destination	Public RIReports	Package selectedTestConfigObj TestConfiguration	
Realization SRS50.1 to RIReports Source -> Destination	Public SRS50.1	Public RIReports	
Realization Source -> Destination	Public SRS50.9	Public RIReports	
Realization Source -> Destination	Public SRS50.7	Public RIReports	
Realization Source -> Destination	Public SRS49.3	Public RIReports	
Realization SRS50.4 to RIReports Source -> Destination	Public SRS50.4	Public RIReports	
Realization SRS49 to RIReports Source -> Destination	Public SRS49	Public RIReports	
Realization SRS49.1 to RIReports Source -> Destination	Public SRS49.1	Public RIReports	
Realization SRS51 to RIReports Source -> Destination	Public SRS51	Public RIReports	
Realization SRS50.6 to RIReports Source -> Destination	Public SRS50.6	Public RIReports	
Realization SRS60 to RIReports Source -> Destination	Public SRS60	Public RIReports	
Realization SRS50 to RIReports Source -> Destination	Public SRS50	Public RIReports	
Realization SRS50.5 to RIReports Source -> Destination	Public SRS50.5	Public RIReports	
Realization SRS49.2 to RIReports	Public	Public	

Connector	Source	Target	Notes
Source -> Destination	SRS49.2	RIReports	
Realization SRS54 to RIReports Source -> Destination	Public SRS54	Public RIReports	
Realization SRS50.2 to RIReports Source -> Destination	Public SRS50.2	Public RIReports	
Realization SRS53 to RIReports Source -> Destination	Public SRS53	Public RIReports	
Realization SRS50.3 to RIReports Source -> Destination	Public SRS50.3	Public RIReports	
Realization Source -> Destination	Public SRS50.8	Public RIReports	

Attributes

Attribute	Notes	Constraints and tags
All_Configreport_Source String Public Static	The All_ configreport_ source.	*Default:* "./reports/c2cRIReport.jasper"
All_Logreport_Source String Public Static	The All_ logreport_ source.	*Default:* "./reports/TestScriptActionLogReport.jrxml"
All_TestScripts_Source String Public Static	The All_ test scripts_ source.	*Default:* "./reports/TestScriptsReport.jasper"
C2C_RI_VERSION_PARAMETER String Public Static	The C2 c_ r i_ versio n_ parameter.	*Default:* "C2CRIVERSION"
CONFIG_CHECKSUM_PARAMETER String Public Static	The config checksum parameter.	*Default:* "CONFIGURATIONCHECKSUM"
Conformance_Logreport_So	The Conformance_ logreport_ source.	*Default:*

Attribute	Notes	Constraints and tags
urce String Public Static		"./reports/ConformanceReport.jrxml"
conn Connection Package	The conn.	*Default:* null
Default_Reports_Path String Public Static	The Default_ reports_ path.	*Default:* "./reports/"
MsgDetail_Logreport_Source String Public Static	The Msg detail_ logreport_ source.	*Default:* "./reports/MessageDetailReport.jrxml"
MsgSummary_Logreport_Source String Public Static	The Msg summary_ logreport_ source.	*Default:* "./reports/MessageSummaryReport.jrxml"
pstmt PreparedStatement Package	The pstmt.	*Default:* null
selectedTestConfigObj TestConfiguration Package	The selected test config obj.	*Default:* null
TestCaseDetails_Logreport_Source String Public Static	The Test case details_ logreport_ source.	*Default:* "./reports/TestCaseDetailReport.jrxml"
TestCaseSummary_Logreport_Source String Public Static	The Test case summary_ logreport_ source.	*Default:* "./reports/TestCaseSummaryReport.jrxml"

Operations

Method	Notes	Parameters
createConfigurationReport() boolean Public	Generate a C2C RI test configuration Jasper report. @return true, if successful	HashMap [in] reportParameters the report parametersTestConfiguration [in] testConfig

Method	Notes	Parameters
		Selected TestConfiguration objectString [in] configFileName Test Configuration file name
createLogExport() boolean Public	method called when the user wants to create a Test Log CSV Export. The method then provides the specified test log information to Jasper Reports.. This method utilizes the ProgresUI to inform the user that the report is running. After the report is complete the ProgressUI is closed. Log reports will be exported as CSV files. @return true, if successful	HashMap [in] reportParameters the report parameters
createLogReport() boolean Public	method called when the user wants to create a Test Log Report. The method then provides the specified test log information to Jasper Reports.. This method utilizes the ProgresUI to inform the user that the report is running. After the report is complete the ProgressUI is closed. Log reports will be exported as PDF files. @return true, if successful	HashMap [in] reportParameters the report parameters
createTestScriptReport() boolean Public	Generate a C2C RI test configuration Jasper report. @return true, if successful	HashMap [in] reportParameters the report parametersTestConfiguration [in] testConfig Selected TestConfiguration objectString [in] configFileName Test Configuration file name
writeAppLayerDataToTempDB() void Private	Write out AppLayer data to temporary SQLite database.	
writeAppLayerNRTMDataToTempDB() void Private	Write out AppLayer data to temporary SQLite database.	
writeConfigDataToTempDB() void Package	Write out configuration data to temporary SQLite database.	String [in] configFileName the config file name
writeInfoLayerDataToTempDB() void	Write out InfoLayer data to temporary SQLite database.	

Method	Notes	Parameters
Private		
writeInfoLayerNRTMDataToTempDB() void	Write out InfoLayer data to temporary SQLite database.	
Private		
writeSUTdataToTempDB() void	Write out SUT data to temporary SQLite database.	
Private		
writeTestConfigDataToTempDB() void	Store TestConfiguration data for reporting purpose.	String [in] configFileName the config file name
Package		
writeTestScriptsToTempDB() void	Write out test script data to temporary SQLite database.	
Private		

5.7.1.3 Context Diagram

N/A

5.7.1.4 Variability Guide

N/A

5.7.1.5 Rationale

N/A

5.7.2 Test Config Report

The Test Config Report functionality has been implemented by jasper report files.

5.7.3 PassFail Report

The PassFail Report functionality has been implemented by jasper report files.

5.7.4 Test Activity Report

The Test Activity Report functionality has been implemented by jasper report files.

5.8 C2C SUT Interface

5.8.1 Center Services

5.8.1.1 Primary Presentation

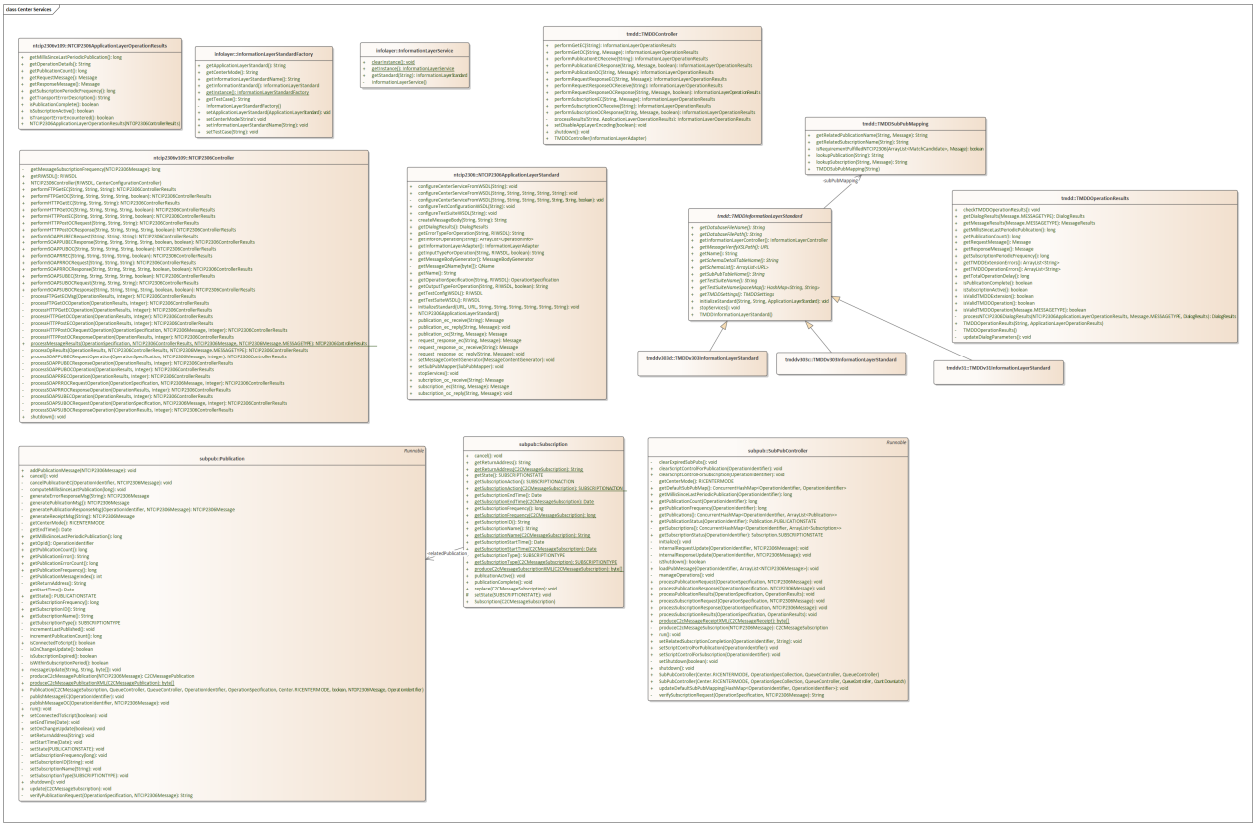

Figure 5-32 Center Services Modules

5.8.1.2 Element Catalog

5.8.1.2.1 InformationLayerService

Type: **Class** *Package:* infolayer

The Class InformationLayerService returns the Information Layer Standard plugin with a given name.

Connections

Connector	Source	Target	Notes
Realization SRS64 to InformationLayerService Source -> Destination	Public SRS64	Public InformationLayerServic e	

Attributes

Attribute	Notes	Constraints and tags
loader ServiceLoader<InformationLayerStandard> Private	The loader.	*Default:*
service InformationLayerService Private Static	The service.	*Default:*

Operations

Method	Notes	Parameters
Static **clearInstance()** void Public	Clear the singleton static instance of InformationLayerService.	
Static **getInstance()** InformationLayerService Public	Retrieve the singleton static instance of ApplicationLayerService. @return single instance of InformationLayerService	
getStandard() InformationLayerStandard Public	Retrieve the information layer standard from the first provider that matches the provided standard name. @return the standard	String [in] standard the standard
InformationLayerService() Private	Creates a new instance of DictionaryService.	

5.8.1.2.2 InformationLayerStandardFactory

Type: **Class** *Package:* infolayer

The factory for creating InformationLayerStandard objects.

Connections

Connector	Source	Target	Notes
Association Source -> Destination	Public InformationLayerStanda	Private applicationLayerStandar	

Connector	Source	Target	Notes
	rdFactory	d ApplicationLayerStandard	
Association Source -> Destination	Public InformationLayerStandardFactory	Private informationLayerStandard InformationLayerStandard	
Sequence Source -> Destination	Public InformationLayerStandardFactory	Public C2CRISessionTag	
Sequence Source -> Destination	Public InformationLayerStandardFactory	Public C2CRISessionTag	
Sequence getInstance() Source -> Destination	Public C2CRISessionTag	Public InformationLayerStandardFactory	
Realization SRS64 to InformationLayerStandard Factory Source -> Destination	Public SRS64	Public InformationLayerStandardFactory	
Sequence getInformationStandard() Source -> Destination	Public C2CRISessionTag	Public InformationLayerStandardFactory	

Attributes

Attribute	Notes	Constraints and tags
applicationLayerStandard ApplicationLayerStandard Private	The application layer standard.	*Default:*
centerMode String Private	The center mode.	*Default:*
informationLayerStandard InformationLayerStandard Private	The information layer standard.	*Default:*
informationLayerStandardName String	The information layer standard name.	*Default:*

Attribute	Notes	Constraints and tags
Private		
log Logger Protected Static	The log.	*Default:* Logger.getLogger(InformationLayerStandardFactory.class.getName())
testCase String Private	The test case.	*Default:*
thisFactory InformationLayerStandardFactory Private Static	The this factory.	*Default:*

Operations

Method	Notes	Parameters
getApplicationLayerStandard() String Public	Gets the application layer standard. @return the application layer standard	
getCenterMode() String Public	Gets the center mode. @return the center mode	
getInformationLayerStandardName() String Public	Gets the information layer standard name. @return the information layer standard name	
getInformationStandard() InformationLayerStandard Public	Gets the information standard. @return the information standard	
Static **getInstance()** InformationLayerStandardFactory Public	Gets the single instance of InformationLayerStandardFactory. Pre-Conditions: N/A Post-Conditions: N/A @return single instance of InformationLayerStandardFactory	

Method	Notes	Parameters
getTestCase() String Public	Gets the test case. @return the test case	
InformationLayerStandardFactory() Private	Instantiates a new information layer standard factory. Pre-Conditions: N/A Post-Conditions: N/A	
setApplicationLayerStandard() void Public	Sets the application layer standard.	ApplicationLayerStandard [in] applicationLayerStandard the new application layer standard
setCenterMode() void Public	Sets the center mode.	String [in] centerMode the new center mode
setInformationLayerStandardName() void Public	Sets the information layer standard name.	String [in] informationLayerStandardName the new information layer standard name
setTestCase() void Public	Sets the test case.	String [in] testCase the new test case

5.8.1.2.3 NTCIP2306ApplicationLayerOperationResults

Type: **Class** *Package:* ntcip2306v109

This class represents the NTCIP 2306 application layer operation results that are provided as an ApplicationLayerOperationResults object.

Connections

Connector	Source	Target	Notes
Realization Source -> Destination	Public NTCIP2306ApplicationLayerOperationResults	Public ApplicationLayerOperationResults	
Association Source -> Destination	Public NTCIP2306ApplicationLayerOperationResults	Private results NTCIP2306ControllerResults	
Realization SRS8.2 to NTCIP2306ApplicationLayerOperationResults Source -> Destination	Public SRS8.2	Public NTCIP2306ApplicationLayerOperationResults	

Connector	Source	Target	Notes
Realization SRS8.2.1 to NTCIP2306ApplicationLayerOperationResults Source -> Destination	Public SRS8.2.1	Public NTCIP2306Application LayerOperationResults	
Realization SRS8.2.2 to NTCIP2306ApplicationLayerOperationResults Source -> Destination	Public SRS8.2.2	Public NTCIP2306Application LayerOperationResults	

Attributes

Attribute	Notes	Constraints and tags
results NTCIP2306ControllerResults Private	The results.	*Default:*

Operations

Method	Notes	Parameters
getMillisSinceLastPeriodicPublication() long Public	Gets the millis since last periodic publication. @return the time in milliseconds since the previous publication.	
getOperationDetails() String Public	Gets the operation details. @return the operation details.	
getPublicationCount() long Public	Gets the publication count. @return the number of publications that have occurred.	
getRequestMessage() Message Public	Gets the request message. @return the request message received in the dialog.	
getResponseMessage() Message	Gets the response message. @return the response message received in	

Method	Notes	Parameters
Public	the dialog.	
getSubscriptionPeriodicFrequency() long Public	Gets the subscription periodic frequency. @return the publication frequency requested in the subscription.	
getTransportErrorDescription() String Public	Gets the transport error description. @return the description for any transport errors.	
isPublicationComplete() boolean Public	Checks if is publication complete. Pre-Conditions: N/A Post-Conditions: N/A @return whether the publication has completed.	
isSubscriptionActive() boolean Public	Checks if is subscription active. Pre-Conditions: N/A Post-Conditions: N/A @return whether the subscription is active.	
isTransportErrorEncountered() boolean Public	Checks if is transport error encountered. Pre-Conditions: N/A Post-Conditions: N/A @return whether a transport error was encountered during the dialog.	
NTCIP2306ApplicationLayerOperationResults() Public	Constructor.	NTCIP2306ControllerResults [in] results the results

5.8.1.2.4 NTCIP2306Controller

Type: **Class** *Package:* ntcip2306v109

Provides the main controller function for NTCIP 2306 related operations. It is provides a controlled interface to the CenterConfigurationController object.

Connections

Connector	Source	Target	Notes
Association	Public	Private ccc	
Source -> Destination	NTCIP2306Controller	CenterConfigurationController	
Association	Public	Private riWSDL	
Source -> Destination	NTCIP2306Controller	RIWSDL	
Association	Public	Private controller	
Source -> Destination	NTCIP2306InformationLayerAdapter	NTCIP2306Controller	
Realization SRS8.2.2 to NTCIP2306Controller	Public	Public	
Source -> Destination	SRS8.2.2	NTCIP2306Controller	
Association	Public	Private theService	
Source -> Destination	NTCIP2306SessionTag	NTCIP2306Controller	

Attributes

Attribute	Notes	Constraints and tags
ccc CenterConfigurationController Private	the referenced Center Configuration Controller object.	*Default:*
log Logger Protected Static	The log.	*Default:* Logger.getLogger(NTCIP2306Controller.class.getName())
riWSDL RIWSDL Private	the RIWSDL object which represents the wsdl file.	*Default:*

Operations

Method	Notes	Parameters
getMessageSubscriptionFrequency() long Private	Gets the message subscription frequency. @return the message subscription frequency	NTCIP2306Message [in] message the message
getRiWSDL() RIWSDL Public	Gets the ri wsdl. @return the wsdl object	

Method	Notes	Parameters
NTCIP2306Controller() Public	Constructor.	RIWSDL [in] theWSDL - reference to the WSDL fileCenterConfigurationController [in] ccc - reference to the CenterConfigurationController
performFTPGetEC() NTCIP2306ControllerResults Public	Perform an FTP Get EC Operation. @return - the results of the operation.	String [in] service - the related service from the WSDLString [in] port - the related port from the WSDLString [in] operation - the related operation from the WSDL
performFTPGetOC() NTCIP2306ControllerResults Public	Perform an FTP Get OC Operation. @return - the results of the operation. the results of the operation.	String [in] service - the related service from the WSDLString [in] port - the related port from the WSDLString [in] operation - the related operation from the WSDLString [in] responseMessage - the response message to be transmittedboolean [in] skipEncoding - send the response message without additional encoding
performHTTPGetEC() NTCIP2306ControllerResults Public	Perform an HTTP Get EC Operation. @return - the results of the operation.	String [in] service - the related service from the WSDLString [in] port - the related port from the WSDLString [in] operation - the related operation from the WSDL
performHTTPGetOC() NTCIP2306ControllerResults Public	Perform and HTTP Get OC Operation. @return - the results of the operation	String [in] service - the related service from the WSDLString [in] port - the related port from the WSDLString [in] operation - the related operation from the WSDLString [in] responseMessage - the response message to be sent to the SUTboolean [in] skipEncoding

Method	Notes	Parameters
		- flag indicating whether to encode the responseMessage
performHTTPPostEC() NTCIP2306ControllerResults Public	Perform the HTTP Post EC Operation. @return the results of the operation	String [in] service - the related service from the WSDLString [in] port - the related port from the WSDLString [in] operation - the related operation from the WSDLString [in] requestMessage - the request message to be sent to the SUTboolean [in] skipEncoding - flag indicating whether to encode the requestMessage
performHTTPPostOCRequest() NTCIP2306ControllerResults Public	Perform the HTTP Post OC Request Operation. @return - the results of the operation.	String [in] service - the related service from the WSDLString [in] port - the related port from the WSDLString [in] operation - the related operation from the WSDL
performHTTPPostOCResponse() NTCIP2306ControllerResults Public	Perform the HTTP Post OC Response Operation. @return - the results of the operation. the results of the operation	String [in] service - the related service from the WSDLString [in] port - the related port from the WSDLString [in] operation - the related operation from the WSDLString [in] responseMessage - the response message to send to the SUTboolean [in] skipEncoding - flag indication whether to encode the responseMessage
performSOAPPUBECRequest() NTCIP2306ControllerResults Public	Perform the Request portion of a SOAP Publication operation as an EC. @return - the results of the operation.	String [in] service - the related service from the WSDLString [in] port - the related port from the WSDLString [in] operation - the related operation from the WSDL
performSOAPPUBECResponse() NTCIP2306ControllerResults Public	Perform the response portion of a SOAP Publication operation as an EC. @return the results of the operation.	String [in] service - the related service from the WSDLString [in] port - the related port from the

Method	Notes	Parameters
		WSDLString [in] operation - the related operation from the WSDLString [in] responseMessage - the response message to be sent to the SUT.boolean [in] skipEncoding - flag indicating whether to encode the response Message.boolean [in] isErrorResponse - flag indicating whether the responseMessage reports an error.
performSOAPPUBOC() NTCIP2306ControllerResults Public	Perform a SOAP Publication operation as an OC. @return - the results of the operation.	String [in] service - the related service from the WSDLString [in] port - the related port from the WSDLString [in] operation - the related operation from the WSDLString [in] requestMessage - the request message to be sent to the SUT.boolean [in] skipEncoding - flag indicating whether to encode the requestMessage.
performSOAPRREC() NTCIP2306ControllerResults Public	Perform the SOAP Request Response EC Operation. @return - the results of the operation	String [in] service - the related service from the WSDLString [in] port - the related port from the WSDLString [in] operation - the related operation from the WSDLString [in] requestMessage - the request message to be sent to the SUTboolean [in] skipEncoding - flag indicating whether to encode the request message
performSOAPRROCRequest() NTCIP2306ControllerResults Public	Perform the Request portion of a SOAP Request Response OC operation. @return - the results of the operation.	String [in] service - the related service from the WSDLString [in] port - the related port from the WSDLString [in] operation - the related operation from the WSDL
performSOAPRROCRespons e() NTCIP2306ControllerResults Public	Perform the response portion of a SOAP Request Response OC operation. @return the results of the operation.	String [in] service - the related service from the WSDLString [in] port

Method	Notes	Parameters
		- the related port from the WSDLString [in] operation - the related operation from the WSDLString [in] responseMessage - the response message to be sent to the SUTboolean [in] skipEncoding - flag indicating whether the responseMessage should be encodedboolean [in] isErrorResponse - indicate whether the response message reports an error
performSOAPSUBEC() NTCIP2306ControllerResults Public	Perform a SOAP Subscription operation as an EC. @return - the results of the operation. the results of the operation.	String [in] service - the related service from the WSDLString [in] port - the related port from the WSDLString [in] operation - the related operation from the WSDLString [in] requestMessage - the request message to be sent to the SUT.boolean [in] skipEncoding - flag indicating whether additional encoding should be performed on the request message.
performSOAPSUBOCRequest() NTCIP2306ControllerResults Public	Perform the request portion of a SOAP subscription as an OC. @return - the results of the operation.	String [in] service - the related service from the WSDLString [in] port - the related port from the WSDLString [in] operation - the related operation from the WSDL
performSOAPSUBOCResponse() NTCIP2306ControllerResults Public	Perform the response portion of a SOAP Subscription operation as an OC. @return the results of the operation.	String [in] service - the related service from the WSDLString [in] port - the related port from the WSDLString [in] operation - the related operation from the WSDLString [in] responseMessage - the response message to send to the SUTboolean [in] skipEncoding - flag indicating whether to perform encoding on the responseMessageboolean [in]

Method	Notes	Parameters
		isErrorResponse - flag indicating whether the responseMessage reports an error.
processFTPGetECMsg() NTCIP2306ControllerResults Private	Process the message of an FTP Get operation as an EC. @return the results of the processing.	OperationResults [in] opResults - the results from the FTP Get operationInteger [in] responseTimeOut - the response timeout value setting.
processFTPGetOCOperation() NTCIP2306ControllerResults Private	Process the message of an FTP Get operation as an OC. @return the results of the processing.	OperationResults [in] opResults - the results from the FTP Get operationInteger [in] requestTimeOut the request time out
processHTTPGetECOperation() NTCIP2306ControllerResults Private	Process the message of an HTTP Get operation as an EC. @return the results of the processing.	OperationResults [in] opResults - the results from the HTTP Get operationInteger [in] responseTimeOut - the response timeout value setting.
processHTTPGetOCOperation() NTCIP2306ControllerResults Private	Process the message of an HTTP Get operation as an OC. @return the results of the processing.	OperationResults [in] opResults - the results from the HTTP Get operationInteger [in] requestTimeOut the request time out
processHTTPPostECOperation() NTCIP2306ControllerResults Private	Process the message of an HTTP Post operation as an EC. @return the results of the processing.	OperationResults [in] opResults - the results from the HTTP Post operationInteger [in] responseTimeOut - the response timeout value setting.
processHTTPPostOCRequestOperation() NTCIP2306ControllerResults Private	Process the request message of an HTTP Get operation as an OC. @return the results of the processing.	OperationSpecification [in] opSpec - the specification for the operation.NTCIP2306Message [in] requestMsg - the request message being processed.Integer [in] requestTimeOut - the request timeout value setting.
processHTTPPostOCResponseOperation() NTCIP2306ControllerResults Private	Process the results of an HTTP Post operation as an OC. @return the results of the processing.	OperationResults [in] opResults - the results from the HTTP Post operationInteger [in] responseTimeOut

Method	Notes	Parameters
		- the response timeout value setting.
processMessageResults() NTCIP2306ControllerResults Private	Incorporate the results of message diagnostics into the controller results. @return the updated controller results.	OperationSpecification [in] opSpec - the specification that defines the operationNTCIP2306ControllerResults [in] controllerResults - the controller results that are to be updatedNTCIP2306Message [in] targetMessage - the message that is to be evaluated.NTCIP2306Message.MESSAGETYPE [in] targetMessageType - indicates whether the target message is a Request, Response or Error message.
processOpResults() NTCIP2306ControllerResults Private	Process the operation results and include them in an NTCIP2306ControllerResults. @return the updated controller results.	OperationResults [in] opResults - the results from an NTCIP 2306 operationNTCIP2306ControllerResults [in] controllerResults - the original NTCIP2306ControllerResultNTCIP2306Message.MESSAGETYPE [in] targetMessageType - designates whether a Request, Response or Error message type is to be processed.
processSOAPPUBECRequestOperation() NTCIP2306ControllerResults Private	Process the request message of a SOAP Publication operation as an EC. @return the results of the processing.	OperationSpecification [in] opSpec - the specification for the operation.NTCIP2306Message [in] requestMsg - the request message being processed.Integer [in] requestTimeOut - the request timeout value setting.
processSOAPPUBECResponseOperation() NTCIP2306ControllerResults Private	Process the operation results of the response portion of a SOAP Publication operation as an EC. @return the results of the processing.	OperationResults [in] opResults - the results from the SOAP Publication operationInteger [in] responseTimeOut - the response timeout value setting.
processSOAPPUBOCOperation() NTCIP2306ControllerResults Private	Process the operation results of the response portion of a SOAP Publication operation as an OC. @return the results of the processing.	OperationResults [in] opResults - the results from the SOAP Publication operationInteger [in] responseTimeOut

Method	Notes	Parameters
		- the response timeout value setting.
processSOAPRRECOperation() NTCIP2306ControllerResults Private	Process the operation results of an SOAP Request Response operation as an EC. @return the results of the processing.	OperationResults [in] opResults - the results from the SOAP Request Response operationInteger [in] responseTimeOut - the response timeout value setting.
processSOAPRROCRequestOperation() NTCIP2306ControllerResults Private	Process the request message of a SOAP Request Response operation as an OC. @return the results of the processing.	OperationSpecification [in] opSpec - the specification for the operation.NTCIP2306Message [in] requestMsg - the request message being processed.Integer [in] requestTimeOut - the request timeout value setting.
processSOAPRROCResponseOperation() NTCIP2306ControllerResults Private	Process the operation results of the response portion of an SOAP Request Response operation as an OC. @return the results of the processing.	OperationResults [in] opResults - the results from the SOAP Request Response operationInteger [in] responseTimeOut - the response timeout value setting.
processSOAPSUBECOperation() NTCIP2306ControllerResults Private	Process the operation results of a SOAP Subscription operation as an EC. @return the results of the processing.	OperationResults [in] opResults - the results from the SOAP Subscription operationInteger [in] responseTimeOut - the response timeout value setting.
processSOAPSUBOCRequestOperation() NTCIP2306ControllerResults Private	Process the request message of an SOAP Subscription operation as an OC. @return the results of the processing.	OperationSpecification [in] opSpec - the specification for the operation.NTCIP2306Message [in] requestMsg - the request message being processed.Integer [in] requestTimeOut - the request timeout value setting.
processSOAPSUBOCResponseOperation() NTCIP2306ControllerResults Private	Process the operation results of the response portion of a SOAP Subscription operation as an OC. @return the results of the processing.	OperationResults [in] opResults - the results from the SOAP Subscription operationInteger [in] responseTimeOut - the response timeout value setting.
shutdown() void	Shutdown all of the NTCIP 2306 services	

Method	Notes	Parameters
Public	defined by the WSDL.	

5.8.1.2.5 Publication

Type: **Class** Runnable *Package:* subpub

The Class Publication manages the operation of a publication related to one-time, periodic or on-change subscriptions for both EC and OC modes. It also handles publication cancellation events.

Connections

Connector	Source	Target	Notes
Association	Public	Private opSpec	
Source -> Destination	Publication	OperationSpecification	
Association	Public	Private eqc	
Source -> Destination	Publication	QueueController	
Association	Public	Private iqc	
Source -> Destination	Publication	QueueController	
Association	Public	Private contents	
Source -> Destination	Publication	C2CMessagePublication	
Association update notifications Unspecified	Public Publication	Public EntityDataUpdateServic e	The EntityDataUpdateService notifies the publication object when information potentially related to a publication update has changed.
Association	Public	Private opId	
Source -> Destination	Publication	OperationIdentifier	
Association	Public	Private centerMode	
Source -> Destination	Publication	RICENTERMODE	
Association Source -> Destination	Public Publication	Package c2cObjectFactory ObjectFactory	
Association Source -> Destination	Public Publication	Private subscriptionType SUBSCRIPTIONTYPE	
Association	Public	Private state	
Source -> Destination	Publication	PUBLICATIONSTATE	
Association Publication Update Requests	Public Publication	Public RIEmulation	The Publication references RI Emulation in order to obtain publication messages for

Connector	Source	Target	Notes
Unspecified			distribution.
Association Source -> Destination	Public SubPubController	Public Publication	
Association Source -> Destination	Public Subscription	Private relatedPublication Publication	

Attributes

Attribute	Notes	Constraints and tags
c2cObjectFactory ObjectFactory Package	The c2c object factory.	*Default:* new ObjectFactory()
centerMode Center.RICENTERMODE Private	The center mode.	*Default:*
connectedToScript boolean Private	The connected to script.	*Default:* false
contents C2CMessagePublication Private	The contents.	*Default:*
endTime Date Private	The end time.	*Default:*
eqc QueueController Private Const	The eqc.	*Default:*
iqc QueueController Private Const	The iqc.	*Default:*
lastPublicationTimeInMillis long Private	The last publication time in millis.	*Default:* 0

Attribute	Notes	Constraints and tags
maxPublications long Private Const	The max publications.	*Default:* 4294967295L
millisSinceLastPublication Long Private	The millis since last publication.	*Default:* 0L
onChangeUpdate boolean Private	The on change update.	*Default:* false
opId OperationIdentifier Private	The op id.	*Default:*
opSpec OperationSpecification Private	The op spec.	*Default:*
publicationCount long Private	The publication count.	*Default:* 0
publicationError String Private	The publication error.	*Default:* ""
publicationErrorCount long Private	The publication error count.	*Default:* 0
publicationMessageIndex int Private	The publication message index.	*Default:* 0
publicationMessages ArrayList<NTCIP2306Message> Private Const	The publication messages.	*Default:* new ArrayList<NTCIP2306Message>()
publicationsTransmitted long Private	The publications transmitted.	*Default:* 0
returnAddress String	The return address.	*Default:*

Attribute	Notes	Constraints and tags
Private		
shutdown boolean Private	The shutdown.	*Default:* false
startTime Date Private	The start time.	*Default:*
state PUBLICATIONSTATE Private	The state.	*Default:* PUBLICATIONSTATE.INITIALIZING
subscriptionFrequency long Private	The subscription frequency.	*Default:*
subscriptionID String Private	The subscription id.	*Default:*
subscriptionName String Private	The subscription name.	*Default:*
subscriptionType Subscription.SUBSCRIPTIONTYPE Private	The subscription type.	*Default:*

Operations

Method	Notes	Parameters
addPublicationMessage() void Public	Adds the publication message. Pre-Conditions: N/A Post-Conditions: N/A	NTCIP2306Message [in] pubMessage the pub message
cancel() void Public	Cancel. Pre-Conditions: N/A Post-Conditions: N/A	
computeMillisSinceLastPublication() void Private	Compute millis since last publication. Pre-Conditions: N/A Post-Conditions: N/A	long [in] currentTimeInMillis the current time in millis
generateErrorResponseMsg()	Generate error response msg. Pre-	String [in] errorText

Method	Notes	Parameters
NTCIP2306Message Private	Conditions: N/A Post-Conditions: N/A @return the NTCIP2306 message	the error text
generatePublicationMsg() NTCIP2306Message Private	Generate publication msg. Pre-Conditions: N/A Post-Conditions: N/A @return the NTCIP2306 message	
generatePublicationResponseMsg() NTCIP2306Message Private	Generate publication response msg. Pre-Conditions: N/A Post-Conditions: N/A @return the NTCIP2306 message	OperationIdentifier [in] pubOpId the pub op idNTCIP2306Message [in] publicationRequest the publication request
generateReceiptMsg() NTCIP2306Message Private	Generate receipt msg. Pre-Conditions: N/A Post-Conditions: N/A @return the NTCIP2306 message	String [in] informationText the information text
getCenterMode() RICENTERMODE Private	Gets the center mode. @return the center mode	
getEndTime() Date Private	Gets the end time. @return the end time	
getMillisSinceLastPeriodicPublication() long Public	Gets the millis since last periodic publication. @return the millis since last periodic publication	
getOpId() OperationIdentifier Public	Gets the op id. @return the op id	
getPublicationCount() long	Gets the publication count.	

Method	Notes	Parameters
Public	@return the publication count	
getPublicationError() String Public	Gets the publication error. @return the publication error	
getPublicationErrorCount() long Public	Gets the publication error count. @return the publication error count	
getPublicationMessageIndex() int Private	Gets the publication message index. @return the publication message index	
getReturnAddress() String Private	Gets the return address. @return the return address	
getStartTime() Date Private	Gets the start time. @return the start time	
getState() PUBLICATIONSTATE Public	Gets the state. @return the state	
getSubscriptionFrequency() long Private	Gets the subscription frequency. @return the subscription frequency	
getSubscriptionID() String Public	Gets the subscription id. @return the subscription id	
getSubscriptionName() String	Gets the subscription name.	

Method	Notes	Parameters
Public	@return the subscription name	
getSubscriptionType() SUBSCRIPTIONTYPE Private	Gets the subscription type. @return the subscription type	
incrementLastPublished() void Private	Increment last published. Pre-Conditions: N/A Post-Conditions: N/A	
incrementPublicationCount() long Private	Increment publication count. Pre-Conditions: N/A Post-Conditions: N/A @return the long	
isConnectedToScript() boolean Public	Checks if is connected to script. Pre-Conditions: N/A Post-Conditions: N/A @return true, if is connected to script	
isOnChangeUpdate() boolean Private	Checks if is on change update. Pre-Conditions: N/A Post-Conditions: N/A @return true, if is on change update	
isSubscriptionExpired() boolean Private	Checks if is subscription expired. Pre-Conditions: N/A Post-Conditions: N/A @return true, if is subscription expired	
isWithinSubscriptionPeriod() boolean Private	Checks if is within subscription period. Pre-Conditions: N/A Post-Conditions: N/A @return true, if is within subscription period	
produceC2cMessagePublication() C2CMessagePublication Private	Produce c2c message publication. Pre-Conditions: N/A Post-Conditions: N/A @return the c2 c message publication	NTCIP2306Message [in] message the message

Method	Notes	Parameters
Static **produceC2cMessagePublicationXML()** byte Public	Produce c2c message publication xml. Pre-Conditions: N/A Post-Conditions: N/A @return the byte[]	C2CMessagePublication [in] message the message
Publication() Public	Instantiates a new publication. Pre-Conditions: N/A Post-Conditions: N/A	C2CMessageSubscription [in] relatedSubscription the related subscriptionQueueController [in] eqc the eqcQueueController [in] iqc the iqcOperationIdentifier [in] opId the op idOperationSpecification [in] opSpec the op specCenter.RICENTERMODE [in] centerMode the center modeboolean [in] connectedToScript the connected to script
publishMessageEC() void Private	Publish message ec. Pre-Conditions: N/A Post-Conditions: N/A	OperationIdentifier [in] pubOpId the pub op id
publishMessageOC() void Private	Publish message oc. Pre-Conditions: N/A Post-Conditions: N/A	OperationIdentifier [in] pubOpId the pub op idNTCIP2306Message [in] publishMessage the publish message
run() void Public	Loop while the publication is in the initializing state Loop while the publication is not in the completed state If the publication is not being updated Perform publications per the specifications of the subscription.	
setConnectedToScript() void Public	Sets the connected to script.	boolean [in] connectedToScript the new connected to script
setEndTime() void Private	Sets the end time.	Date [in] endTime the new end time
setOnChangeUpdate() void Public	Sets the on change update.	boolean [in] onChangeUpdate the new on change update

Method	Notes	Parameters
setReturnAddress() void Private	Sets the return address.	String [in] returnAddress the new return address
setStartTime() void Private	Sets the start time.	Date [in] startTime the new start time
setState() void Private	Sets the state.	PUBLICATIONSTATE [in] state the new state
setSubscriptionFrequency() void Private	Sets the subscription frequency.	long [in] subscriptionFrequency the new subscription frequency
setSubscriptionID() void Private	Sets the subscription id.	String [in] subscriptionID the new subscription id
setSubscriptionName() void Private	Sets the subscription name.	String [in] subscriptionName the new subscription name
setSubscriptionType() void Private	Sets the subscription type.	SUBSCRIPTIONTYPE [in] subscriptionType the new subscription type
shutdown() void Public	Shutdown. Pre-Conditions: N/A Post-Conditions: N/A	
update() void Public	Update. Pre-Conditions: N/A Post-Conditions: N/A	C2CMessageSubscription [in] message the message
verifyPublicationRequest() String Private	Verify publication request. Pre-Conditions: N/A Post-Conditions: N/A @return the string	OperationSpecification [in] opSpec the op specNTCIP2306Message [in] pubRequestMsg the pub request msg

5.8.1.2.6 SubPubController

Type: **Class** Runnable *Package:* subpub

The Class SubPubController handles the generation of subscriptions and publications for both owner center and external center modes of operation.

Connections

Connector	Source	Target	Notes
Association	Public	Public	
Source -> Destination	SubPubController	Publication	

Connector	Source	Target	Notes
Association	Public	Private iqc	
Source -> Destination	SubPubController	QueueController	
Association subscription requests Unspecified	Public SubPubController	Public RIEmulation	The SubPubController references RI Emulation in order to confirm subscription requests.
Association	Public	Private eqc	
Source -> Destination	SubPubController	QueueController	
Association	Public	Private centerMode	
Source -> Destination	SubPubController	RICENTERMODE	
Association Source -> Destination	Public SubPubController	Private opSpecCollection OperationSpecCollection	
Association	Public	Private subController	
Source -> Destination	CenterConfigurationController	SubPubController	

Attributes

Attribute	Notes	Constraints and tags
centerMode Center.RICENTERMODE Private	The center mode.	*Default:*
defaultSubPubMap HashMap<OperationIdentifier, OperationIdentifier> Private Const	The default sub pub map.	*Default:* new HashMap<OperationIdentifier, OperationIdentifier>()
definedPublications ArrayList<OperationIdentifier> Private Const	The defined publications.	*Default:* new ArrayList<OperationIdentifier>()
definedSubscriptions ArrayList<OperationIdentifier> Private Const	The defined subscriptions.	*Default:* new ArrayList<OperationIdentifier>()
eqc QueueController Private	The eqc.	*Default:*

Attribute	Notes	Constraints and tags
Const		
executor ExecutorService Private Const	The executor.	*Default:* Executors.newCachedThreadPool(new NamedThreadFactory("SubPubControl-" + getCenterMode()))
iqc QueueController Private Const	The iqc.	*Default:*
opSpecCollection OperationSpecCollection Private Const	The op spec collection.	*Default:*
publicationMessageMap HashMap<OperationIdentifier, ArrayList<NTCIP2306Message>> Private Const	The publication message map.	*Default:* new HashMap()
publications HashMap<OperationIdentifier, ArrayList<Publication>> Private Const	The publications.	*Default:* new HashMap<OperationIdentifier, ArrayList<Publication>>()
readySignal CountDownLatch Private Const	The ready signal.	*Default:*
scriptedPublications ArrayList<OperationIdentifier> Private Const	The scripted publications.	*Default:* new ArrayList<OperationIdentifier>()
scriptedSubscriptions ArrayList<OperationIdentifier> Private Const	The scripted subscriptions.	*Default:* new ArrayList<OperationIdentifier>()
shutdown boolean Private	The shutdown.	*Default:* false
subscriptions	The subscriptions.	*Default:* new

Attribute	Notes	Constraints and tags
HashMap<OperationIdentifier, ArrayList<Subscription>> Private Const		HashMap<OperationIdentifier, ArrayList<Subscription>>()

Operations

Method	Notes	Parameters
clearScriptControlForPublication() void Public	Clear script control for publication. Pre-Conditions: N/A Post-Conditions: N/A	OperationIdentifier [in] opId the op id
clearScriptControlForSubscription() void Public	Clear script control for subscription. Pre-Conditions: N/A Post-Conditions: N/A	OperationIdentifier [in] opId the op id
getCenterMode() RICENTERMODE Private	Gets the center mode. @return the center mode	
getDefaultSubPubMap() HashMap<OperationIdentifier, OperationIdentifier> Public	Gets the default sub pub map. @return the default sub pub map	
getMillisSinceLastPeriodicPublication() long Public	Gets the millis since last periodic publication. @return the millis since last periodic publication	OperationIdentifier [in] opId the op id
getPublicationCount() long Public	Gets the publication count. @return the publication count	OperationIdentifier [in] opId the op id
getPublications() HashMap<OperationIdentifier, ArrayList<Publication>> Private	Gets the publications. @return the publications	
getPublicationStatus() Publication.PUBLICATIONSTATE Public	Gets the publication status. @return the publication status	OperationIdentifier [in] opId the op id

Method	Notes	Parameters
getSubscriptions() HashMap<OperationIdentifier, ArrayList<Subscription>> Private	Gets the subscriptions. @return the subscriptions	
getSubscriptionStatus() Subscription.SUBSCRIPTION STATE Public	Gets the subscription status. @return the subscription status	OperationIdentifier [in] opId the op id
initialize() void Private	Initialize. Pre-Conditions: N/A Post-Conditions: N/A	
internalRequestUpdate() void Private	Internal request update. Pre-Conditions: N/A Post-Conditions: N/A	OperationIdentifier [in] operationId the operation idNTCIP2306Message [in] em the em
internalResponseUpdate() void Private	Internal response update. Pre-Conditions: N/A Post-Conditions: N/A	OperationIdentifier [in] operationId the operation idNTCIP2306Message [in] em the em
isShutdown() boolean Private	Checks if is shutdown. Pre-Conditions: N/A Post-Conditions: N/A @return true, if is shutdown	
loadPubMessage() void Public	Load pub message. Pre-Conditions: N/A Post-Conditions: N/A	OperationIdentifier [in] opId the op idArrayList<NTCIP2306Message> [in] messageList the message list
manageOperations() void Private	Manage operations. Pre-Conditions: N/A Post-Conditions: N/A	
processPublicationRequest() void Public	Process publication request. Pre-Conditions: N/A Post-Conditions: N/A	OperationSpecification [in] opSpec the op specNTCIP2306Message [in] pubRequestMsg the pub request msg
processPublicationResponse() void Public	Process publication response. Pre-Conditions: N/A Post-Conditions: N/A	OperationSpecification [in] opSpec the op specNTCIP2306Message [in] pubResponseMsg

Method	Notes	Parameters
		the pub response msg
processPublicationResults() void Public	Process publication results. Pre-Conditions: N/A Post-Conditions: N/A	OperationSpecification [in] opSpec the op specOperationResults [in] pubOperationResults the pub operation results
processSubscriptionRequest() void Public	If the messages are valid to schema and valid xml If both message parts are correct message types Create a c2cSubscription object from the subscription message If the subscription ID doesn't already exist send a receipt message back to the requester Else update the existing subscription with the c2cSubscriptionObject send a receipt message back to the requester Else Create an Invalid Response Message and send back to the source if a script is listening Store the operation result? Else Create an Invalid Response Message and send back to the source if a script is listening Store the operation result? End If.	OperationSpecification [in] opSpec the op specNTCIP2306Message [in] subRequestMsg the sub request msg
processSubscriptionResponse() void Public	If the messages are valid to schema and valid xml If both message parts are correct message types Create a c2cSubscription object from the subscription message If the subscription ID doesn't already exist send a receipt message back to the requester.	OperationSpecification [in] opSpec the op specNTCIP2306Message [in] subResponseMsg the sub response msg
processSubscriptionResults() void Public	Process subscription results. Pre-Conditions: N/A Post-Conditions: N/A	OperationSpecification [in] opSpec the op specOperationResults [in] subOperationResults the sub operation results
Static **produceC2cMessageReceiptXML()** byte Public	Produce c2c message receipt xml. Pre-Conditions: N/A Post-Conditions: N/A @return the byte[]	C2CMessageReceipt [in] message the message
produceC2cMessageSubscription() C2CMessageSubscription Private	Produce c2c message subscription. Pre-Conditions: N/A Post-Conditions: N/A @return the c2 c message subscription	NTCIP2306Message [in] message the message
run() void Public	Initialize the controller by registering sub and pub operations with the default listener If this Center is an owner center For each	

Method	Notes	Parameters
	sub or pub operation loop check for sub requests if there is a sub request operation from a handler process the subscription request check for pub responses if there is a pub response operation from a handler process the publication response check for sub operation results if there is a sub operation result from a handler process the sub operation result Else if this Center is an external center check for sub responses if there is a sub response operation from a handler process the subscription response check for pub requests if there is a pub request operation from a handler process the publication request check for pub operation results if there is a pub operation result from a handler process the pub operation result.	
setScriptControlForPublication() void Public	Sets the script control for publication.	OperationIdentifier [in] opId the new script control for publication
setScriptControlForSubscription() void Public	Sets the script control for subscription.	OperationIdentifier [in] opId the new script control for subscription
setShutdown() void Private	Sets the shutdown.	boolean [in] shutdown the new shutdown
shutdown() void Public	Shutdown. Pre-Conditions: N/A Post-Conditions: N/A	
SubPubController() Public	Instantiates a new sub pub controller. Pre-Conditions: N/A Post-Conditions: N/A	Center.RICENTERMODE [in] centerMode the center modeOperationSpecCollection [in] opSpec the op specQueueController [in] iqc the iqcQueueController [in] eqc the eqc
SubPubController() Public	Instantiates a new sub pub controller. Pre-Conditions: N/A Post-Conditions: N/A	Center.RICENTERMODE [in] centerMode the center modeOperationSpecCollection [in] opSpec the op specQueueController [in] iqc the iqcQueueController [in] eqc the eqcCountDownLatch [in]

Method	Notes	Parameters
		readySignal the ready signal
updateDefaultSubPubMapping() void Public	Update default sub pub mapping. Pre-Conditions: N/A Post-Conditions: N/A	HashMap<OperationIdentifier, OperationIdentifier> [in] subPubMap the sub pub map
verifySubscriptionRequest() String Private	Verify subscription request. Pre-Conditions: N/A Post-Conditions: N/A @return the string	OperationSpecification [in] opSpec the op specNTCIP2306Message [in] subRequestMsg the sub request msg

5.8.1.2.7 Subscription

Not directly traced to an SRS requirement. See source code.

5.8.1.2.8 NTCIP2306ApplicationLayerStandard

Type: **Class** *Package:* ntcip2306

This class is a NTCIP2306 implementation of the ApplicationLayerStandard interface.

Tagged Values

Ÿ annotations = @PluginImplementation.

Connections

Connector	Source	Target	Notes
Association Source -> Destination	Public NTCIP2306ApplicationLayerStandard	Private ccc2 CenterConfigurationController	
Realization Source -> Destination	Public NTCIP2306ApplicationLayerStandard	Public ApplicationLayerStandard	
Association Source -> Destination	Public NTCIP2306ApplicationLayerStandard	Private testConfigWSDL RIWSDL	
Association Source -> Destination	Public NTCIP2306ApplicationLayerStandard	Private dialogResults DialogResults	

Connector	Source	Target	Notes
Association Source -> Destination	Public NTCIP2306ApplicationLayerStandard	Private testSuiteWSDL RIWSDL	
Association Source -> Destination	Public NTCIP2306ApplicationLayerStandard	Private messageBodyGenerator MessageBodyGenerator	
Realization SRS40.2 to NTCIP2306ApplicationLayerStandard Source -> Destination	Public SRS40.2	Public NTCIP2306ApplicationLayerStandard	
Realization SRS41.5 to NTCIP2306ApplicationLayerStandard Source -> Destination	Public SRS41.5	Public NTCIP2306ApplicationLayerStandard	
Realization SRS40.4 to NTCIP2306ApplicationLayerStandard Source -> Destination	Public SRS40.4	Public NTCIP2306ApplicationLayerStandard	
Realization SRS41.3 to NTCIP2306ApplicationLayerStandard Source -> Destination	Public SRS41.3	Public NTCIP2306ApplicationLayerStandard	
Realization SRS40.1 to NTCIP2306ApplicationLayerStandard Source -> Destination	Public SRS40.1	Public NTCIP2306ApplicationLayerStandard	
Realization SRS41.2 to NTCIP2306ApplicationLayerStandard Source -> Destination	Public SRS41.2	Public NTCIP2306ApplicationLayerStandard	
Realization Source -> Destination	Public SRS40.11	Public NTCIP2306ApplicationLayerStandard	
Realization SRS41.6 to NTCIP2306ApplicationLayerStandard Source -> Destination	Public SRS41.6	Public NTCIP2306ApplicationLayerStandard	
Realization Source -> Destination	Public SRS40.12	Public NTCIP2306ApplicationLayerStandard	
Realization SRS40.9 to NTCIP2306ApplicationLa	Public SRS40.9	Public NTCIP2306Application	

Connector	Source	Target	Notes
yerStandard		LayerStandard	
Source -> Destination			
Realization SRS40.5 to NTCIP2306ApplicationLayerStandard	Public SRS40.5	Public NTCIP2306Application LayerStandard	
Source -> Destination			
Realization SRS41.1 to NTCIP2306ApplicationLayerStandard	Public SRS41.1	Public NTCIP2306Application LayerStandard	
Source -> Destination			
Realization SRS40.3 to NTCIP2306ApplicationLayerStandard	Public SRS40.3	Public NTCIP2306Application LayerStandard	
Source -> Destination			
Realization SRS40.8 to NTCIP2306ApplicationLayerStandard	Public SRS40.8	Public NTCIP2306Application LayerStandard	
Source -> Destination			
Realization SRS41.7 to NTCIP2306ApplicationLayerStandard	Public SRS41.7	Public NTCIP2306Application LayerStandard	
Source -> Destination			
Realization SRS41.4 to NTCIP2306ApplicationLayerStandard	Public SRS41.4	Public NTCIP2306Application LayerStandard	
Source -> Destination			
Realization SRS40.6 to NTCIP2306ApplicationLayerStandard	Public SRS40.6	Public NTCIP2306Application LayerStandard	
Source -> Destination			
Association	Public	Private appStandard	
Source -> Destination	TMDDv303Information LayerStandard	NTCIP2306Application LayerStandard	
Realization SRS40.7 to NTCIP2306ApplicationLayerStandard	Public SRS40.7	Public NTCIP2306Application LayerStandard	
Source -> Destination			
Realization SRS41 to NTCIP2306ApplicationLayerStandard	Public SRS41	Public NTCIP2306Application LayerStandard	
Source -> Destination			
Realization	Public	Public	

Connector	Source	Target	Notes
Source -> Destination	SRS40.10	NTCIP2306Application LayerStandard	

Attributes

Attribute	Notes	Constraints and tags
APPLICATION_STANDARD_NAME String Public Static		*Default:* "NTCIP2306v01"
applicationLayerStandard String Private		*Default:*
ccc2 CenterConfigurationController Private		*Default:*
centerMode String Public		*Default:*
dialogResults DialogResults Private		*Default:*
ExternalCenterMode String Public Static		*Default:* "EC"
informationLayerStandard String Private		*Default:*
log Logger Protected Static		*Default:* Logger.getLogger(NTCIP2306ApplicationLayerStandard.class.getName())
messageBodyGenerator MessageBodyGenerator Private		*Default:* null

Attribute	Notes	Constraints and tags
NTCIP2306FTPTransport String Public Static		*Default:* "FTP"
NTCIP2306GZIPEncoding String Public Static		*Default:* "GZipXML"
NTCIP2306HTTPTransport String Public Static		*Default:* "HTTP"
NTCIP2306SOAPEncoding String Public Static		*Default:* "SOAP_XML"
NTCIP2306Transport String Public Static		*Default:* "NTCIP2306v1"
NTCIP2306XMLEncoding String Public Static		*Default:* "XML"
OwnerCenterMode String Public Static		*Default:* "OC"
portName String Public		*Default:*
publicationDialog String Private		*Default:*
requestDialog String Private		*Default:*
serviceName String Public		*Default:*

Attribute	Notes	Constraints and tags
subscriptionDialog String Private		*Default:*
testCase String Private		*Default:*
testConfigWSDL RIWSDL Private		*Default:* null
testSuiteWSDL RIWSDL Private		*Default:* null
transportOperations Map<String, Map> Public	String is the portName and the Map is <operationName, OperationSpec>	*Default:* new HashMap<String, Map>()
useWSDL boolean Private		*Default:*

Operations

Method	Notes	Parameters
configureCenterServiceFromWSDL() void Private		String [in] wsdlURL String [in] testCase String [in] mode String [in] serviceName String [in] portName String [in] operationName boolean [in] isPublication
configureCenterServiceFromWSDL() void Public		String [in] wsdlURL String [in] serviceName String [in] portName String [in] operationName String [in] pathName

Method	Notes	Parameters
configureCenterServiceFromWSDL() void Public	public NTCIP2306ApplicationLayerStandard(String appStd, String infoStd, String requestDialog, String subscriptionDialog, String publicationDialog, String riMode, String testCaseIdentifier, boolean useWSDL) { this.centerMode = riMode; this.testCase = testCaseIdentifier; this.applicationLayerStandard = appStd; this.informationLayerStandard = infoStd; this.requestDialog = requestDialog; this.subscriptionDialog = subscriptionDialog; this.publicationDialog = publicationDialog; this.dialogResults = new DialogResults(); this.transportOperations = new HashMap<String, Map>(); // String is the portName and the Map is <operationName, OperationSpec> } public NTCIP2306ApplicationLayerStandard(String mode, String transportType, String encoding, String testCase) throws Exception { portName = transportType + "_Name"; this.transportOperations = new HashMap<String, Map>(); // String is the portName and the Map is <operationName, OperationSpec> this.dialogResults = new DialogResults(); } public NTCIP2306ApplicationLayerStandard(String mode, String serviceName, String portName, String operationName, boolean createFromWSDL, String wsdlURL, boolean isPublication, String transportType, String encoding, String testCase) throws Exception { this.dialogResults = new DialogResults(); this.transportOperations = new HashMap<String, Map>(); // String is the portName and the Map is <operationName, OperationSpec> if (createFromWSDL) { configureCenterServiceFromWSDL(wsdlURL, testCase, mode, serviceName, portName, operationName, isPublication); if (testConfigWSDL.isSchemaDocumentsExist()) { Map<String, URL> schemaURLMap = this.getTestConfigWSDL().getWsdlSchemas(); URL[] schemaURLs = new URL[schemaURLMap.size()]; Iterator urlIterator = schemaURLMap.values().iterator(); int ii = 0; while (urlIterator.hasNext()) { schemaURLs[ii] = (URL) urlIterator.next(); ii++; } try { this.messageBodyGenerator = new MessageBodyGenerator(schemaURLs); } catch (Exception ex) { // The Test Case	String [in] wsdlURL

Method	Notes	Parameters
	may not be trying to use automated message generation ex.printStackTrace(); } } } else { this.serviceName = serviceName; this.centerMode = mode; this.portName = portName; } }	
configureTestConfigurationWSDL() void Public		String [in] wsdlURL
configureTestSuiteWSDL() void Public		String [in] wsdlURL
createMessageBody() String Public		String [in] operation String [in] messageType
getDialogResults() DialogResults Public		
getErrorTypeForOperation() String Public		String [in] operation RIWSDL [in] targetWSDL
getInfoForOperation() ArrayList<OperationInfo> Public		String [in] operation
getInformationLayerAdapter() InformationLayerAdapter Public		
getInputTypeForOperation() String Public		String [in] operation RIWSDL [in] targetWSDL boolean [in] pubSub
getMessageBodyGenerator() MessageBodyGenerator Public		
getMessageQName() QName Private		byte[] [in] message
getName() String Public		
getOperationSpecification() OperationSpecification		String [in] operation

Method	Notes	Parameters
Public		RIWSDL [in] targetWSDL
getOutputTypeForOperation() String Public		String [in] operation RIWSDL [in] targetWSDL boolean [in] pubSub
getTestConfigWSDL() RIWSDL Public		
getTestSuiteWSDL() RIWSDL Public		
initializeStandard() void Public		URL [in] serviceSpecificationURL URL [in] baseStandardSpecificationURL String [in] centerMode String [in] informationLayerStandard String [in] testCaseName String [in] requestDialog String [in] subscriptionDialog String [in] publicationDialog
NTCIP2306ApplicationLayerStandard() Public	Creates a new Instance of NTCIP2306ApplicationLayerStandard	
publication_ec_receive() Message Public		String [in] dialog
publication_ec_reply() void Public		String [in] dialog Message [in] responseMessage
publication_oc() Message Public		String [in] dialog Message [in] publicationMessage
request_response_ec() Message Public		String [in] dialog Message [in] requestMessage

Method	Notes	Parameters
request_response_oc_receive(**)** Message Public		String [in] dialog
request_response_oc_reply() void Public		String [in] dialog Message [in] responseMessage
setMessageContentGenerator **()** void Public		MessageContentGenerator [in] msgGenerator
setSubPubMapper() void Public		SubPubMapper [in] subPubMapper
stopServices() void Public		
subcription_oc_receive() Message Public		String [in] dialog
subscription_ec() Message Public		String [in] dialog Message [in] subscriptionMessage
subscription_oc_reply() void Public		String [in] dialog Message [in] responseMessage

5.8.1.2.9 TMDDv303Controller

Type: **Class** *Package:* tmddv303c

The Class TMDDv303Controller wraps the InformationLayerAdapter provided by all Application Layer Standards, into an Information Layer Controller.

Connections

Connector	Source	Target	Notes
Association	Public	Private adapter	
Source -> Destination	TMDDv303Controller	NTCIP2306Information LayerAdapter	
Realization	Public	Public	
Source -> Destination	TMDDv303Controller	InformationLayerController	
Association	Public	Private adapter	

Connector	Source	Target	Notes
Source -> Destination	TMDDv303Controller	InformationLayerAdapter	
Realization	Public	Public	
Source -> Destination	SRS8.1	TMDDv303Controller	
Association	Public	Private informationLayerController	
Source -> Destination	TMDDv303InformationLayerStandard	TMDDv303Controller	
Realization	Public	Public	
Source -> Destination	SRS8	TMDDv303Controller	

Attributes

Attribute	Notes	Constraints and tags
adapter InformationLayerAdapter Private	The adapter.	*Default:*

Operations

Method	Notes	Parameters
performGetEC() InformationLayerOperationResults Public	Perform get ec. Pre-Conditions: N/A Post-Conditions: N/A @return the information layer operation results	String [in] dialog the dialog
performGetOC() InformationLayerOperationResults Public	Perform get oc. Pre-Conditions: N/A Post-Conditions: N/A @return the information layer operation results	String [in] dialog the dialogMessage [in] responseMessage the response message
performPublicationECReceive() InformationLayerOperationResults Public	Perform publication ec receive. Pre-Conditions: N/A Post-Conditions: N/A @return the information layer operation results	String [in] dialog the dialog
performPublicationECResponse() InformationLayerOperationRes	Perform publication ec response. Pre-Conditions: N/A Post-Conditions: N/A @return the information layer operation	String [in] dialog the dialogMessage [in]

Method	Notes	Parameters
ults Public	results	responseMessage the response messageboolean [in] isErrorResponse the is error response
performPublicationOC() InformationLayerOperationResults Public	Perform publication oc. Pre-Conditions: N/A Post-Conditions: N/A @return the information layer operation results	String [in] dialog the dialogMessage [in] requestMessage the request message
performRequestResponseEC() InformationLayerOperationResults Public	Perform request response ec. Pre-Conditions: N/A Post-Conditions: N/A @return the information layer operation results	String [in] dialog the dialogMessage [in] requestMessage the request message
performRequestResponseOCReceive() InformationLayerOperationResults Public	Perform request response oc receive. Pre-Conditions: N/A Post-Conditions: N/A @return the information layer operation results	String [in] dialog the dialog
performRequestResponseOCResponse() InformationLayerOperationResults Public	Perform request response oc response. Pre-Conditions: N/A Post-Conditions: N/A @return the information layer operation results	String [in] dialog the dialogMessage [in] responseMessage the response messageboolean [in] isErrorResponse the is error response
performSubscriptionEC() InformationLayerOperationResults Public	Perform subscription ec. Pre-Conditions: N/A Post-Conditions: N/A @return the information layer operation results	String [in] dialog the dialogMessage [in] requestMessage the request message
performSubscriptionOCReceive() InformationLayerOperationResults Public	Perform subscription oc receive. Pre-Conditions: N/A Post-Conditions: N/A @return the information layer operation results	String [in] dialog the dialog
performSubscriptionOCResp	Perform subscription oc response. Pre-	String [in] dialog

Method	Notes	Parameters
onse() InformationLayerOperationResults Public	Conditions: N/A Post-Conditions: N/A @return the information layer operation results	the dialogMessage [in] responseMessage the response messageboolean [in] isErrorResponse the is error response
processResults() InformationLayerOperationResults Private	Process results. Pre-Conditions: N/A Post-Conditions: N/A @return the information layer operation results	String [in] dialog the dialogApplicationLayerOperationResults [in] apResults the ap results
setDisableAppLayerEncoding() void Public	Sets the disable app layer encoding.	boolean [in] disableAppLayerEncoding the new disable app layer encoding
shutdown() void Public	Shutdown. Pre-Conditions: N/A Post-Conditions: N/A	
TMDDv303Controller() Public	Instantiates a new tMD dv303 controller. Pre-Conditions: N/A Post-Conditions: N/A	InformationLayerAdapter [in] controller the controller

5.8.1.2.10 TMDDv303InformationLayerStandard

Type: **Class** *Package:* tmddv303c

The Class TMDDv303InformationLayerStandard.

Tagged Values

Ÿ annotations = @PluginImplementation.

Connections

Connector	Source	Target	Notes
Realization Source -> Destination	Public TMDDv303InformationLayerStandard	Public InformationLayerStandard	
Association Source -> Destination	Public TMDDv303InformationLayerStandard	Private informationLayerController TMDDv303Controller	

Connector	Source	Target	Notes
Association	Public	Private controller	
Source -> Destination	TMDDv303InformationLayerStandard	InformationLayerController	
Association	Public	Private appStandard	
Source -> Destination	TMDDv303InformationLayerStandard	ApplicationLayerStandard	
Association	Public	Private appStandard	
Source -> Destination	TMDDv303InformationLayerStandard	NTCIP2306ApplicationLayerStandard	
Association	Public	Private centerMode	
Source -> Destination	TMDDv303InformationLayerStandard	CENTERMODE	
Association	Public	Private subPubMapping	
Source -> Destination	TMDDv303InformationLayerStandard	TMDDv303SubPubMapping	
Realization SRS8 to TMDDv303InformationLayerStandard	Public	Public	
Source -> Destination	SRS8	TMDDv303InformationLayerStandard	
Realization SRS8.1 to TMDDv303InformationLayerStandard	Public	Public	
Source -> Destination	SRS8.1	TMDDv303InformationLayerStandard	
Realization	Public	Public	
Source -> Destination	SRS95	TMDDv303InformationLayerStandard	

Attributes

Attribute	Notes	Constraints and tags
appStandard ApplicationLayerStandard Private	The app standard.	*Default:*
centerMode CenterDefinitions.CENTERMODE Private	The center mode.	*Default:*
controller InformationLayerController Private	The controller used by this standard.	*Default:*

Attribute	Notes	Constraints and tags
nameSpaceMap HashMap<String, String> Private	The name space map.	*Default:* new HashMap()
subPubMapping TMDDv303SubPubMapping Private	The subscription publication mapping for this standard.	*Default:*
testCaseName String Private	The test case name.	*Default:*

Operations

Method	Notes	Parameters
getInformationLayerController() InformationLayerController Public	Gets the information layer controller. @return the information layer controller	
getName() String Public	Gets the name. @return the name	
initializeStandard() void Public	Initialize standard. Pre-Conditions: N/A Post-Conditions: N/A	String [in] testCaseName the test case nameString [in] centerMode the center modeApplicationLayerStandard [in] appStandard the app standard
stopServices() void Public	Stop services. Pre-Conditions: N/A Post-Conditions: N/A	
TMDDv303InformationLayerStandard() Public	Creates a new Instance of TMDDv303InformationLayerStandard This method is required to support proper ServiceLoader operation.	

5.8.1.2.11 TMDDv303OperationResults

Type: **Class** *Package:* tmddv303c

The Class TMDDv303OperationResults represents the result of a TMDD dialog operation.

Connections

Connector	Source	Target	Notes
Realization Source -> Destination	Public TMDDv303OperationResults	Public InformationLayerOperationResults	
Association Source -> Destination	Public TMDDv303OperationResults	Private appLayerResults ApplicationLayerOperationResults	
Realization SRS92 to TMDDv303OperationResults Source -> Destination	Public SRS92	Public TMDDv303OperationResults	

Attributes

Attribute	Notes	Constraints and tags
appLayerResults ApplicationLayerOperationResults Private	The app layer results.	*Default:*
dialog String Private	The dialog.	*Default:*
dialogName String Private	The dialog name.	*Default:* ""
dialogType String Private	The dialog type.	*Default:* ""
errorMessageName String Private	The error message name.	*Default:* ""
operationErrorsList	The operation errors list.	*Default:* new ArrayList()

Attribute	Notes	Constraints and tags
ArrayList<String> Private		
requestMessageErrorsList ArrayList<String> Private	The request message errors list.	*Default:* new ArrayList()
requestMessageName String Private	The request message name.	*Default:* ""
responseMessageErrorsList ArrayList<String> Private	The response message errors list.	*Default:* new ArrayList()
responseMessageName String Private	The response message name.	*Default:* ""

Operations

Method	Notes	Parameters
getDialogResults() DialogResults Public	Gets the dialog results. @return the dialog results	Message.MESSAGETYPE [in] messageType the message type
getMessageResults() MessageResults Public	Gets the message results. @return the message results	Message.MESSAGETYPE [in] messageType the message type
getMillisSinceLastPeriodicPublication() long Public	Gets the millis since last periodic publication. @return the millis since last periodic publication	
getPublicationCount() long Public	Gets the publication count. @return the publication count	

Method	Notes	Parameters
getRequestMessage() Message Public	Gets the request message. @return the request message	
getResponseMessage() Message Public	Gets the response message. @return the response message	
getSubscriptionPeriodicFrequency() long Public	Gets the subscription periodic frequency. @return the subscription periodic frequency	
getTMDDExtensionErrors() ArrayList<String> Public	Gets the tMDD extension errors. @return the tMDD extension errors	
getTMDDOperationErrors() ArrayList<String> Public	Gets the tMDD operation errors. @return the tMDD operation errors	
getTotalOperationDelay() long Public	Gets the total operation delay. @return the total operation delay	
isPublicationComplete() boolean Public	Checks if is publication complete. Pre-Conditions: N/A Post-Conditions: N/A @return true, if is publication complete	
isSubscriptionActive() boolean Public	Checks if is subscription active. Pre-Conditions: N/A Post-Conditions: N/A @return true, if is subscription active	
isValidTMDDExtension() boolean Public	Checks if is valid tmdd extension. Pre-Conditions: N/A Post-Conditions: N/A @return true, if is valid tmdd extension	

Method	Notes	Parameters
isValidTMDDOperation() boolean Public	Checks if is valid tmdd operation. Pre-Conditions: N/A Post-Conditions: N/A @return true, if is valid tmdd operation	
processNTCIP2306DialogResults() DialogResults Private	Process ntci p2306 dialog results. Pre-Conditions: N/A Post-Conditions: N/A @return the dialog results	NTCIP2306ApplicationLayerOperationResults [in] ntcip2306Results the ntcip2306 resultsMessage.MESSAGETYPE [in] messageType the message typeDialogResults [in] dialogResults the dialog results
TMDDv303OperationResults() Public	Instantiates a new tMD dv303 operation results. Pre-Conditions: N/A Post-Conditions: N/A	String [in] dialog the dialogApplicationLayerOperationResults [in] appLayerResults the app layer results
updateDialogParameters() void Private	Update dialog parameters. Pre-Conditions: N/A Post-Conditions: N/A	

5.8.1.2.12 TMDDv303SubPubMapping

Type: **Class** *Package:* tmddv303c

This class maintains a map between the subscriptions and related publication dialogs defined within TMDD v3.03c.

Connections

Connector	Source	Target	Notes
Realization Source -> Destination	Public TMDDv303SubPubMapping	Public SubPubMapper	
Realization SRS8 to TMDDv303SubPubMapping Source -> Destination	Public SRS8	Public TMDDv303SubPubMapping	
Association Source -> Destination	Public TMDDv303Information	Private subPubMapping TMDDv303SubPubMap	

Connector	Source	Target	Notes
	LayerStandard	ping	
Realization SRS8.1 to TMDDv303SubPubMapping Source -> Destination	Public SRS8.1	Public TMDDv303SubPubMapping	

Attributes

Attribute	Notes	Constraints and tags
pathToDatabaseCopy String Private	The path to database copy.	*Default:* ""

Operations

Method	Notes	Parameters
getRelatedPublicationName() String Public	Given an subscription dialog name and the message associated with the description return the appropriate publication dialog. @return the related publication name	String [in] subscriptionOperationName the subscription operation nameMessage [in] inputMsg the input msg
getRelatedSubscriptionName() String Public	Given an publication dialog name return the appropriate subscription dialog as defined by the standard. @return the related subscription name	String [in] publicationOperationName the publication operation name
isRequirementFulfilledNTCIP2306() boolean Public	This procedure verifies that the content of the inputMessage satisfies each of the conditions defined within the matchList. @return true, if is requirement fulfilled ntci p2306	ArrayList<MatchCandidate> [in] matchList the match listMessage [in] inputMessage the input message
lookupPublication() String Public	Given an publication dialog name return the appropriate subscription dialog as defined by the standard. @return the string	String [in] pubOperation the pub operation
lookupSubscription() String Public	Given an subscription dialog name and the message associated with the description	String [in] subOperation the sub operationMessage [in]

Method	Notes	Parameters
	return the appropriate publication dialog. @return the string	inputMessage the input message
Static **main()** void Public	The main method. Pre-Conditions: N/A Post-Conditions: N/A	String[] [in] args the arguments
setPathToDatabaseCopy() void Public	Sets the path to database copy.	String [in] pathToDatabaseCopy the new path to database copy

5.8.1.3 Context Diagram

N/A

5.8.1.4 Variability Guide

N/A

5.8.1.5 Rationale

N/A

5.8.2 Client Services

The Client Services functionality has been implemented as part of the NTCIP 2306 standard functionality.

5.8.3 XML/SOAP Support

5.8.3.1 NTCIP 2306

The Client Services functionality has been implemented as part of the NTCIP 2306 standard functionality.

5.8.3.2 WSDL

The Client Services functionality has been implemented as part of the NTCIP 2306 standard functionality.

5.8.3.3 SOAP

The Client Services functionality has been implemented as part of the NTCIP 2306 standard functionality.

5.8.3.4 XML

The Client Services functionality has been implemented as part of the NTCIP 2306 standard functionality.

5.8.3.5 HTTP

5.8.3.5.1 Primary Presentation

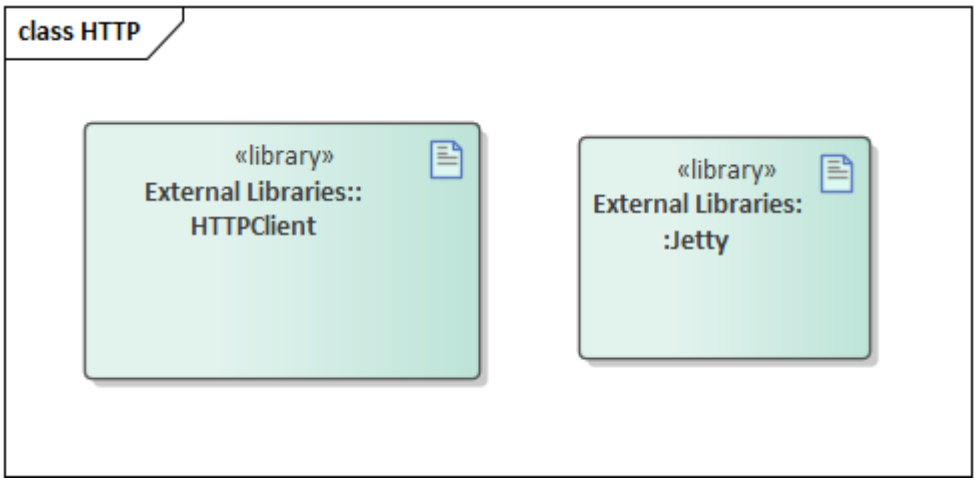

Figure 5-33 HTTP Modules

5.8.3.5.2 Element Catalog

5.8.3.5.2.1 HTTPClient

Type: **Class** *Package:* Logical View

http://hc.apache.org/httpcomponents-client-ga/tutorial/html/index.html

Preface

The Hyper-Text Transfer Protocol (HTTP) is perhaps the most significant protocol used on the Internet today. Web services, network-enabled appliances and the growth of network computing continue to expand the role of the HTTP protocol beyond user-driven web browsers, while increasing the number of applications that require HTTP support.

Although the java.net package provides basic functionality for accessing resources via HTTP, it doesn't provide the full flexibility or functionality needed by many applications. HttpClient seeks to fill this void by providing an efficient, up-to-date, and feature-rich package implementing the client side of the most recent HTTP standards and recommendations.

Designed for extension while providing robust support for the base HTTP protocol, HttpClient may be of interest to anyone building HTTP-aware client applications such as web browsers, web service clients, or systems that leverage or extend the HTTP protocol for distributed communication.

1. HttpClient scope

- Client-side HTTP transport library based on HttpCore
- Based on classic (blocking) I/O
- Content agnostic

2. What HttpClient is NOT

- HttpClient is NOT a browser. It is a client side HTTP transport library. HttpClient's purpose is to transmit and receive HTTP messages. HttpClient will not attempt to process content, execute javascript embedded in HTML pages, try to guess content type, if not explicitly set, or reformat request / redirect location URIs, or other functionality unrelated to the HTTP transport.

Connections

Connector	Source	Target	Notes
Realization Source -> Destination	Public SRS8.2.2	Public HTTPClient	
Realization SRS8.2.2 to HTTPClient Source -> Destination	Public SRS8.2.2	Public HTTPClient	

5.8.3.5.2.2 Jetty

Type: **Class** *Package:* Logical View

http://www.eclipse.org/jetty/

Jetty

Jetty provides a Web server and javax.servlet container, plus support for SPDY, WebSocket, OSGi, JMX, JNDI, JAAS and many other integrations. These components are open source and available for commercial use and distribution.

Jetty is used in a wide variety of projects and products, both in development and production. Jetty can be easily embedded in devices, tools, frameworks, application servers, and clusters. See the Jetty Powered page for more uses of Jetty.

The current recommended version for use is Jetty 9 which can be obtained here: Jetty Downloads. Also available are the latest maintenance releases of Jetty 8 and Jetty 7.

The Jetty project is hosted entirely at the Eclipse Foundation and has been for a number of years. Prior releases of Jetty have existed in part or completely under the Jetty project at the Codehaus. See the About page for more information about the history of Jetty.

Features

Full-featured and standards-based

Open source and commercially usable

Flexible and extensible Small footprint

Embeddable Asynchronous Enterprise scalable

Dual licensed under Apache and Eclipse

Connections

Connector	Source	Target	Notes
Realization Source -> Destination	Public SRS8.2.2	Public Jetty	
Realization SRS8.2.2 to Jetty Source -> Destination	Public SRS8.2.2	Public Jetty	

5.8.3.5.3 Context Diagram

N/A

5.8.3.5.4 Variability Guide

N/A

5.8.3.5.5 Rationale

N/A

5.8.4 DATEX-ASN Support (Deprecated)

5.8.4.1 NTCIP 2304

This function has not been implemented since DATEX-ASN will not be supported.

5.8.4.2 ASN.1

This function has not been implemented since DATEX-ASN will not be supported.

5.8.4.3 UDP/IP

This function has not been implemented since DATEX-ASN will not be supported.

5.8.4.4 TCP/IP

This function has not been implemented since DATEX-ASN will not be supported.

5.9 FTPClient

Type: **Class** *Package:* Logical View

Apache Commons Net

Apache Commons Net™ library implements the client side of many basic Internet protocols. The purpose of the library is to provide fundamental protocol access, not higher-level abstractions. Therefore, some of the design violates object-oriented design principles. Our philosophy is to make the global functionality of a protocol accessible (e.g., TFTP send file and receive file) when possible, but also provide access to the fundamental protocols where applicable so that the programmer may construct his own custom implementations (e.g, the TFTP packet classes and the TFTP packet send and receive methods are exposed).

Features

Supported protocols include:

FTP/FTPS

FTP over HTTP (experimental)

NNTP

SMTP(S)

POP3(S)

IMAP(S)

Telnet

TFTP

Finger

Whois

rexec/rcmd/rlogin

Time (rdate) and Daytime

Echo

Discard

NTP/SNTP

Background

Apache Jakarta Commons Net started as a commercial Java library called NetComponents, originally developed by ORO, Inc. in the early days of Java. After its 1.3.8 release in 1998, the source code was donated to the Apache Software Foundation and made available under the Apache License. Since then, many programmers have contributed to the continued development of Jakarta Commons Net. The current version numbering scheme bears no relation to the old. In other words, Jakarta Commons Net 1.0 succeeded and supplanted NetComponents 1.3.8. Apache Jakarta Commons is now an independent project and is called Apache Commons.

Connections

Connector	Source	Target	Notes
Realization Source -> Destination	Public SRS8.2.2	Public FTPClient	
Realization SRS8.2.2 to FTPClient Source -> Destination	Public SRS8.2.2	Public FTPClient	
Generalization Source -> Destination	Public RIFTPClient	Public FTPClient	

5.10 FTPServer

Type: **Class** *Package:* Logical View

http://mina.apache.org/ftpserver-project/index.html

Overview

The Apache FtpServer is a 100% pure Java FTP server. It's designed to be a complete and portable FTP server engine solution based on currently available open protocols. FtpServer can be run standalone as a Windows service or Unix/Linux daemon, or embedded into a Java application. We also provide support for integration within Spring applications and provide our releases as OSGi bundles.

The default network support is based on Apache MINA, a high performance asynchronous IO library. Using MINA, FtpServer can scale to a large number of concurrent users.

It is also an FTP application platform. We have developed a Java API to let you write Java code to process FTP event notifications that we call the Ftplet API. Apache FtpServer provides an implementation of an FTP server to support this API.

To get started, have a look at one of our tutorials:

- Embedding FtpServer in 5 minutes
- Running FtpServer stand-alone in 5 minutes

You can also have a look at the documentation for how to configure FtpServer to suite your needs.

Connections

Connector	Source	Target	Notes
Realization SRS8.2.2 to FTPServer	Public SRS8.2.2	Public FTPServer	

Connector	Source	Target	Notes
Source -> Destination			
Realization Source -> Destination	Public SRS8.2.2	Public FTPServer	

5.11 Java

Type: **Class** *Package:* Logical View

Java is the foundation for virtually every type of networked application and is the global standard for developing and delivering embedded and mobile applications, games, Web-based content, and enterprise software. With more than 9 million developers worldwide, Java enables you to efficiently develop, deploy and use exciting applications and services.

From laptops to datacenters, game consoles to scientific supercomputers, cell phones to the Internet, Java is everywhere!

- 97% of Enterprise Desktops Run Java
- 89% of Desktops (or Computers) in the U.S. Run Java
- 9 Million Java Developers Worldwide
- #1 Choice for Developers
- #1 Development Platform
- 3 Billion Mobile Phones Run Java
- 100% of Blu-ray Disc Players Ship with Java
- 5 Billion Java Cards in Use
- 125 million TV devices run Java
- 5 of the Top 5 Original Equipment Manufacturers Ship Java ME

Why Software Developers Choose Java

Java has been tested, refined, extended, and proven by a dedicated community of Java developers, architects and enthusiasts. Java is designed to enable development of portable, high-performance applications for the widest range of computing platforms possible. By making applications available across heterogeneous environments, businesses can provide more services and boost end-user productivity, communication, and collaboration—and dramatically reduce the cost of ownership of both enterprise and consumer applications. Java has become invaluable to developers by enabling them to:

- Write software on one platform and run it on virtually any other platform
- Create programs that can run within a web browser and access available web services
- Develop server-side applications for online forums, stores, polls, HTML forms processing, and more
- Combine applications or services using the Java language to create highly customized applications or services
- Write powerful and efficient applications for mobile phones, remote processors, microcontrollers, wireless modules, sensors, gateways, consumer products, and practically any other electronic device

Accessibility

Since 1973 when Section 508 of the Federal Rehabilitation Act was established, US Federal agencies are required to purchase electronic and information technology products and services that meet the standards of Section 508.

</br><td align="left" valign="bottom" width="70297411" bgcolor="#000000" style="border-left:solid #1ea63c 5617340px; border-right:solid #08931f 2639041px; border-top:solid #086e64 84260100px; border-bottom:solid #feffff 4686494px; writing-mode: tb-rl;">For you, the Java technology developer, accessibility means building the services and support into an application that enable people with disabilities to use the software. Java accessibility technologies are implemented in the Java Foundation Classes (JFC) and they comprise two separate packages: Java Accessibility API and Java Accessibility Utilities.Accessibility API and Java Accessibility Utilities.

How to Download the Accessibility Utilities

If you're a general application developer, then you don't need to download the accessibility utilities. You can use the accessibility support that's already built into the JFC.

The Java Accessibility Utilities package provides support so assistive technologies can locate and query user interface objects inside a Java application running in a Java Virtual Machine. It also supports installation of "event listeners" into these objects, and provides example tools that highlight how to enable assistive technologies to interact with the Accessibility API support built into Swing components.

If you're unsure of which release to download, please see the compatibility table.

Note: You don't need to download the accessibility utilities to make your JFC programs support accessibility.

Connections

Connector	Source	Target	Notes
Realization SRS79 to Java Source -> Destination	Public SRS79	Public Java	
Realization SRS74 to Java Source -> Destination	Public SRS74	Public Java	
Realization Source -> Destination	Public SRS81	Public Java	
Realization SRS78.1 to Java Source -> Destination	Public SRS78.1	Public Java	

Connector	Source	Target	Notes
Realization SRS37.2 to Java Source -> Destination	Public SRS37.2	Public Java	
Realization Source -> Destination	Public SRS89	Public Java	
Realization Source -> Destination	Public SRS78.1	Public Java	
Realization Source -> Destination	Public SRS75	Public Java	
Realization Source -> Destination	Public SRS5	Public Java	
Realization SRS7.1 to Java Source -> Destination	Public SRS7.1	Public Java	
Realization Source -> Destination	Public SRS4.4	Public Java	
Realization SRS78 to Java Source -> Destination	Public SRS78	Public Java	
Realization SRS7 to Java Source -> Destination	Public SRS7	Public Java	
Realization Source -> Destination	Public SRS4.5	Public Java	
Realization Source -> Destination	Public SRS4.2	Public Java	
Realization Source -> Destination	Public SRS84.1	Public Java	
Realization Source -> Destination	Public SRS82	Public Java	

Connector	Source	Target	Notes
Realization SRS89 to Java Source -> Destination	Public SRS89	Public Java	
Realization SRS76 to Java Source -> Destination	Public SRS76	Public Java	
Realization SRS84 to Java Source -> Destination	Public SRS84	Public Java	
Realization Source -> Destination	Public SRS76	Public Java	
Realization Source -> Destination	Public SRS4.1	Public Java	
Realization SRS77 to Java Source -> Destination	Public SRS77	Public Java	
Realization Source -> Destination	Public SRS84	Public Java	
Realization Source -> Destination	Public SRS4	Public Java	
Realization Source -> Destination	Public SRS78	Public Java	
Realization SRS80 to Java Source -> Destination	Public SRS80	Public Java	
Realization Source -> Destination	Public SRS4.3	Public Java	
Realization Source -> Destination	Public SRS80	Public Java	
Realization	Public	Public	

Connector	Source	Target	Notes
Source -> Destination	SRS4.6	Java	
Realization Source -> Destination	Public SRS6	Public Java	
Realization Source -> Destination	Public SRS37.2	Public Java	
Realization Source -> Destination	Public SRS74	Public Java	
Realization SRS75 to Java Source -> Destination	Public SRS75	Public Java	
Realization SRS82 to Java Source -> Destination	Public SRS82	Public Java	
Realization Source -> Destination	Public SRS7.1	Public Java	
Realization Source -> Destination	Public SRS7	Public Java	
Realization SRS81 to Java Source -> Destination	Public SRS81	Public Java	
Realization Source -> Destination	Public SRS79	Public Java	
Realization SRS84.1 to Java Source -> Destination	Public SRS84.1	Public Java	
Realization SRS7.2 to Java Source -> Destination	Public SRS7.2	Public Java	
Realization Source -> Destination	Public SRS7.2	Public Java	

Connector	Source	Target	Notes
Realization Source -> Destination	Public SRS77	Public Java	

6 FIGURES

6.1 ReportGeneration

Report Generation Component View - *(Component diagram)*

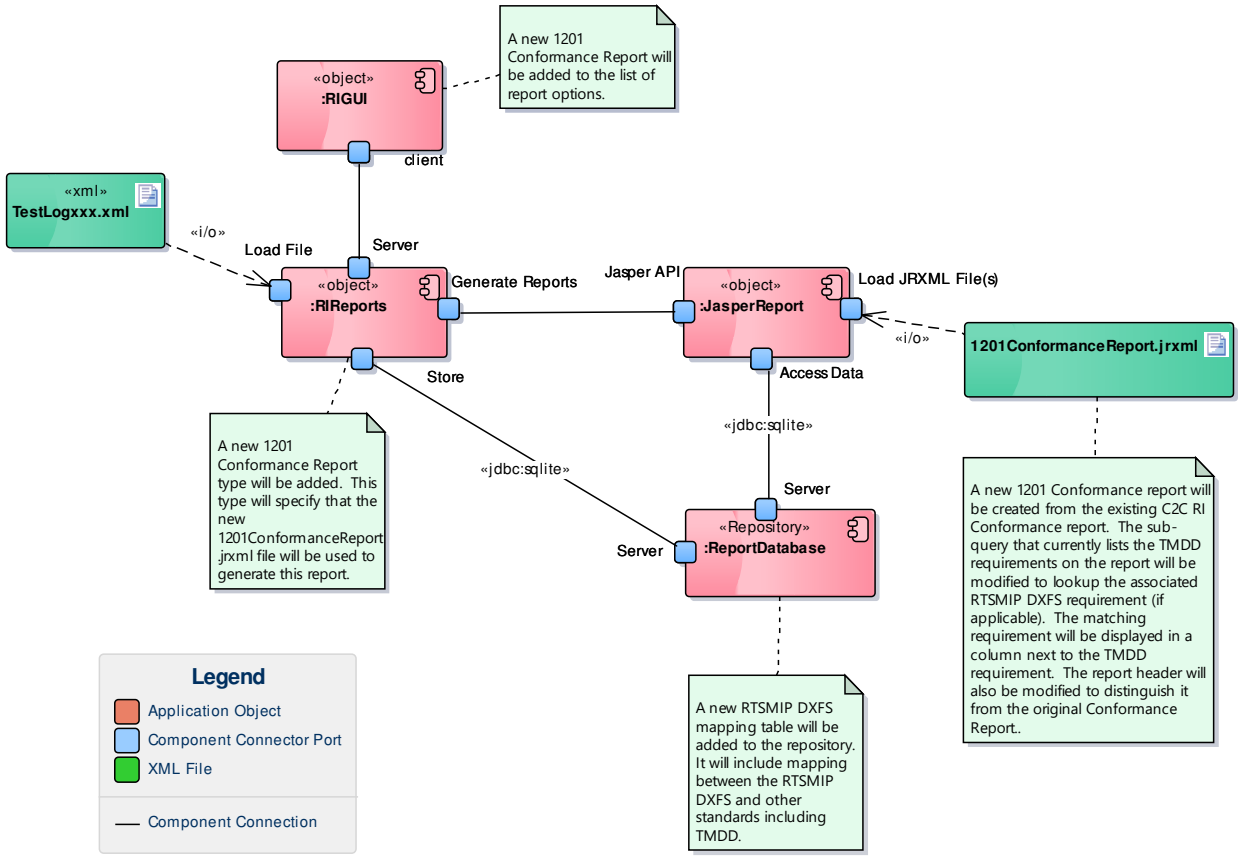

Figure 6-1: Report Generation Components

This component diagram illustrates the significant C2C RI components that are involved in determining and reporting on 1201 Conformance.

6.1.1 :JasperReport

Type: **Component**
Package: ReportGeneration

The C2C RI utilizes the JasperReport engine to generate its reports. The reports are defined in *.jrxml files and are processed by the JasperReport Engine.

6.1.1.1 :RIGUI

Type: **Component**
Package: ReportGeneration

The RI GUI provides the user interface for all user control of the C2C RI application including:
Generating Reports
Creating Test Configurations
Running Tests
Perform File Maintenance
Setting System Options

Connections

Connector	Source	Target	Notes
NoteLink Source -> Destination	Public <anonymous>	Public :RIGUI	

6.1.1.2 :RIReports

Type: **Component**
Package: ReportGeneration

An instance of the RIReports Class. It provides access to all report creation methods.

Connections

Connector	Source	Target	Notes
NoteLink Source -> Destination	Public <anonymous>	Public :RIReports	

6.1.1.3 :ReportDatabase

Type: **Component**
Package: ReportGeneration

A database designed to hold C2C RI Test report information. An instance is created each time reports are generated. It is populated with data from the configuration file and log file that were used for a test.

Connections

Connector	Source	Target	Notes
NoteLink Source -> Destination	Public <anonymous>	Public :ReportDatabase	

6.1.1.4 :TestConfiguration

Type: **Component**
Package: ReportGeneration

6.1.1.5 1201ConformanceReport.jrxml

Type: **Artifact**
Package: ReportGeneration

The 1201ConformanceReport.jrxml file defines the fields, format and data processing for the 1201 Conformance Report in JasperReport syntax.

Connections

Connector	Source	Target	Notes
Dependency Source -> Destination	Public 1201ConformanceReport.jrxml	Public Load JRXML File(s)	
NoteLink Source -> Destination	Public <anonymous>	Public 1201ConformanceReport.jrxml	

6.1.1.6 TestLogxxx.xml

Type: **XmlDocument**
Package: ReportGeneration

The test log generated from a prior C2C RI Test.

Connections

Connector	Source	Target	Notes
Dependency Source -> Destination	Public TestLogxxx.xml	Public Load File	

6.2 Entity Emulation Overview

Type: Package *Package:* TMDD Entity Emulation

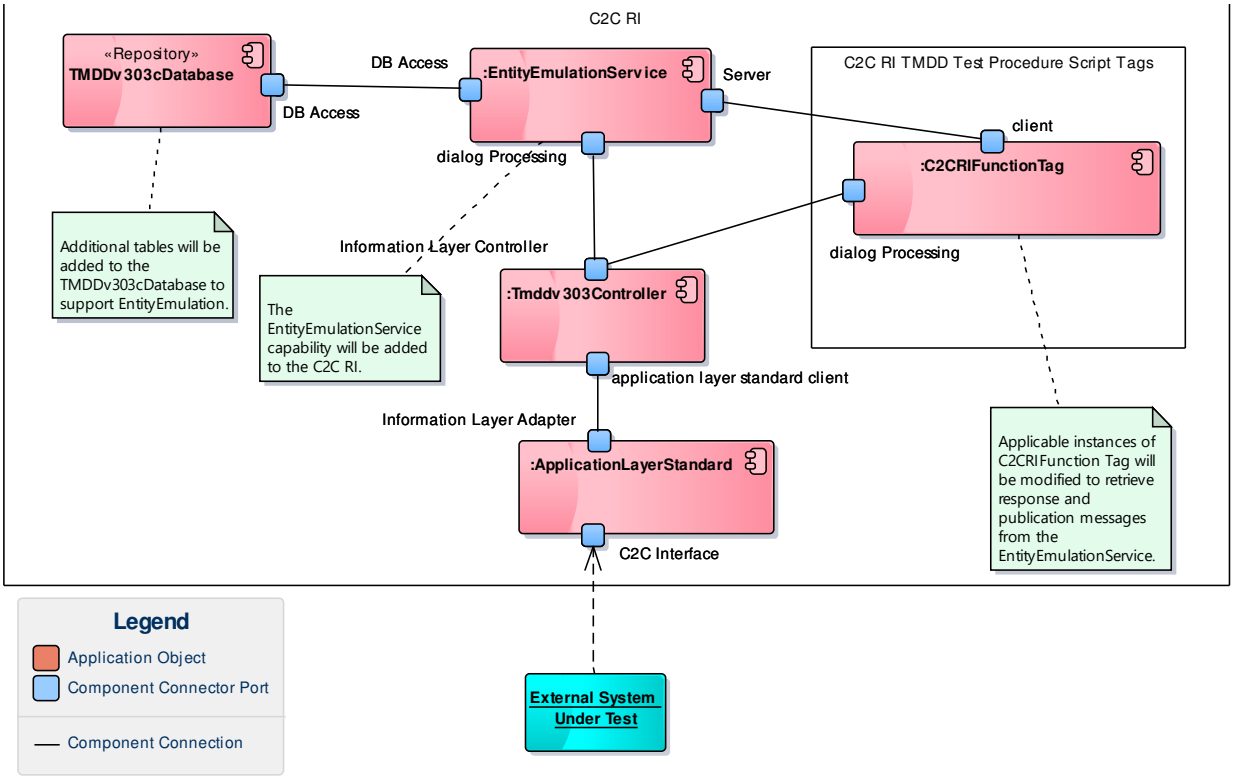

Figure 6-2: TMDD Entity Emulation Overview Component Diagram

This diagram provides an overview of the components involved in satisfying the C2C RI entity emulation requirements.

Refer to the following TMDD Test Procedure documentation to see where each TMDD related dialog requirement is verified.

TPS-1-dlCenterActiveVerificationRequest-EC.xml

TPS-1-dlCenterActiveVerificationSubscription-EC.xmlTPS-5-dlEventIndexRequest-EC

TPS-5-dlEventIndexSubscription-EC

TPS-6-dlFullEventUpdateRequest-EC

TPS-6-dlFullEventUpdateSubscription-EC

TPS-7-dlFullEventUpdateRequest-EC

TPS-7-dlFullEventUpdateSubscription-EC

TPS-8-dlFullEventUpdateRequest-EC

TPS-8-dlFullEventUpdateSubscription-EC

TPS-9-dlActionLogRequest-EC

TPS-9-dlActionLogSubscription-EC

TPS-10-dlNodeInventoryRequest-EC.xml

TPS-10-dlTrafficNetworkInformationSubscription-EC.xml

TPS-11-dlLinkInventoryRequest-EC.xml

TPS-11-dlTrafficNetworkInformationSubscription-EC.xml

TPS-13-dlNodeStatusRequest-EC.xml

TPS-13-dlTrafficNetworkInformationSubscription-EC.xml

TPS-14-dlLinkStatusRequest-EC.xml

TPS-14-dlTrafficNetworkInformationSubscription-EC.xml

TPS-16-dlLinkStatusRequest-EC.xml

TPS-16-dlTrafficNetworkInformationSubscription-EC.xml

TPS-18-dlDetectorInventoryRequest-EC

TPS-18-dlDeviceInformationSubscription-EC

TPS-19-dlDeviceInformationSubscription-EC

TPS-20-dlDetectorStatusRequest-EC

TPS-20-dlDeviceInformationSubscription-EC

TPS-21-dlArchivedDataProcessingDocumentationMetadataRequest-EC

TPS-22-dlDetectorInventoryRequest-EC

TPS-22-dlDeviceInformationSubscription-EC

TPS-23-dlDetectorDataRequest-EC

TPS-23-dlDetectorDataSubscription-EC

TPS-24-dlDetectorMaintenanceHistoryRequest-EC

TPS-25-dlCCTVInventoryRequest-EC

TPS-25-dlDeviceInformationSubscription-EC

TPS-26-dlDeviceInformationSubscription-EC

TPS-27-dlCCTVStatusRequest-EC

TPS-27-dlDeviceInformationSubscription-EC

TPS-28-dlCCTVControlRequest-EC

TPS-29-dlDeviceControlStatusRequest-EC

TPS-30-dlDeviceCancelControlRequest-EC

TPS-31-dlDeviceInformationSubscription-EC

TPS-31-dlVideoSwitchInventoryRequest-EC

TPS-32-dlDeviceInformationSubscription-EC

TPS-33-dlDeviceInformationSubscription-EC

TPS-33-dlVideoSwitchStatusRequest-EC

TPS-34-dlVideoSwitchControlRequest-EC

TPS-35-dlDeviceControlStatusRequest-EC

TPS-36-dlDeviceCancelControlRequest-EC

TPS-37-dlDeviceInformationSubscription-EC

TPS-37-dlDMSInventoryRequest-EC

TPS-38-dlDeviceInformationSubscription-EC

TPS-39-dlDeviceInformationSubscription-EC

TPS-39-dlDMSStatusRequest-EC

TPS-40-dlDMSControlRequest-EC

TPS-41-dlDeviceControlStatusRequest-EC

TPS-42-dlDMSPriorityQueueRequest-EC

TPS-43-dlDeviceCancelControlRequest-EC

TPS-44-dlDMSFontTableRequest-EC

TPS-44-dlDMSMessageAppearanceRequest-EC

TPS-45-dlDMSMessageInventoryRequest-EC

TPS-45-dlDMSMessageInventorySubscription-EC

TPS-46-dlDMSFontTableRequest-EC

TPS-47-dlDeviceInformationSubscription-EC

TPS-47-dlESSInventoryRequest-EC

TPS-48-dlDeviceInformationSubscription-EC

TPS-49-dlDeviceInformationSubscription-EC

TPS-49-dlESSStatusRequest-EC

TPS-50-dlDeviceInformationSubscription-EC

TPS-50-dlESSObservationReportRequest-EC

TPS-51-dlESSObservationMetadataRequest-EC

TPS-52-dlDeviceInformationSubscription-EC

TPS-52-dlESSObservationReportRequest-EC

TPS-53-dlESSObservationMetadataRequest-EC

TPS-54-dlDeviceInformationSubscription-EC

TPS-54-dlGateInventoryRequest-EC

TPS-55-dlDeviceInformationSubscription-EC

TPS-56-dlDeviceInformationSubscription-EC

TPS-56-dlGateStatusRequest-EC

TPS-57-dlGateControlRequest-EC

TPS-58-dlDeviceControlStatusRequest-EC

TPS-59-dlDeviceCancelControlRequest-EC

TPS-60-dlDeviceInformationSubscription-EC

TPS-60-dlGateControlScheduleRequest-EC

TPS-61-dlDeviceInformationSubscription-EC

TPS-61-dlHARInventoryRequest-EC

TPS-62-dlDeviceInformationSubscription-EC

TPS-63-dlDeviceInformationSubscription-EC

TPS-63-dlHARStatusRequest-EC

TPS-64-dlHARControlRequest-EC

TPS-65-dlDeviceControlStatusRequest-EC

TPS-66-dlHARPriorityQueueRequest-EC

TPS-67-dlDeviceCancelControlRequest-EC

TPS-68-dlDeviceInformationSubscription-EC

TPS-68-dlHARControlScheduleRequest-EC

TPS-69-dlDeviceInformationSubscription-EC

TPS-69-dlHARMessageInventoryRequest-EC

TPS-70-dlDeviceInformationSubscription-EC

TPS-70-dlLCSInventoryRequest-EC

TPS-71-dlDeviceInformationSubscription-EC

TPS-72-dlDeviceInformationSubscription-EC

TPS-72-dlLCSStatusRequest-EC

TPS-73-dlLCSControlRequest-EC

TPS-74-dlDeviceControlStatusRequest-EC

TPS-75-dlDeviceCancelControlRequest-EC

TPS-76-dlDeviceInformationSubscription-EC

TPS-76-dlLCSControlScheduleRequest-EC

TPS-77-dlDeviceInformationSubscription-EC

TPS-77-dlRampMeterInventoryRequest-EC

TPS-78-dlDeviceInformationSubscription-EC

TPS-79-dlDeviceInformationSubscription-EC

TPS-79-dlRampMeterStatusRequest-EC

TPS-80-dlRampMeterControlRequest-EC

TPS-81-dlDeviceControlStatusRequest-EC

TPS-82-dlDeviceCancelControlRequest-EC

TPS-83-dlRampMeterPriorityQueueRequest-EC

TPS-84-dlDeviceInformationSubscription-EC

TPS-84-dlRampMeterControlScheduleRequest-EC

TPS-85-dlRampMeterPlanInventoryRequest-EC

TPS-85-dlRampMeterPlanInventorySubscription-EC

TPS-86-dlDeviceInformationSubscription-EC

TPS-86-dlIntersectionSignalInventoryRequest-EC

TPS-87-dlDeviceInformationSubscription-EC

TPS-88-dlDeviceInformationSubscription-EC

TPS-88-dlIntersectionSignalStatusRequest-EC

TPS-89-dlIntersectionSignalControlRequest-EC

TPS-90-dlDeviceControlStatusRequest-EC

TPS-91-dlIntersectionSignalPriorityQueueRequest-EC

TPS-92-dlDeviceCancelControlRequest-EC

TPS-93-dlIntersectionSignalTimingPatternInventoryRequest-EC

TPS-93-dlIntersectionSignalTimingPatternInventorySubscription-EC

TPS-94-dlDeviceInformationSubscription-EC

TPS-94-dlIntersectionSignalControlScheduleRequest-EC

TPS-95-dlDeviceInformationSubscription-EC

TPS-95-dlIntersectionSignalInventoryRequest-EC

TPS-96-dlDeviceInformationSubscription-EC

TPS-96-dlIntersectionSignalInventoryRequest-EC

TPS-97-dlDeviceInformationSubscription-EC

TPS-97-dlIntersectionSignalStatusRequest-EC

TPS-98-dlDeviceInformationSubscription-EC

TPS-98-dlSectionStatusRequest-EC

TPS-99-dlSectionControlRequest-EC

TPS-100-dlSectionControlStatusRequest-EC

TPS-101-dlSectionPriorityQueueRequest-EC

TPS-102-dlDeviceCancelControlRequest-EC

TPS-103-dlDeviceInformationSubscription-EC

TPS-103-dlSectionControlScheduleRequest-EC

TPS-110-dlLinkInventoryRequest-EC.xml

TPS-110-dlLinkStatusRequest-EC.xml

TPS-110-dlNodeInventoryRequest-EC.xml

TPS-110-dlTrafficNetworkInformationSubscription-1-EC.xml

TPS-110-dlTrafficNetworkInformationSubscription-2-EC.xml

TPS-110-dlTrafficNetworkInformationSubscription-3-EC.xml

TPS-111-dlFullEventUpdateRequest-EC

TPS-111-dlFullEventUpdateSubscription-EC

6.2.1 External System Under Test

Type: **Object** *Package:* Entity Emulation Overview

The system that is undergoing a TMDD conformance test and acting as an External Center.

Connections

Connector	Source	Target	Notes
Dependency	Public	Public	
Source -> Destination	External System Under Test	C2C Interface	
Sequence Request from	Public	Public	

Connector	Source	Target	Notes
the EC Source -> Destination	External System Under Test	ApplicationLayerStandard	
Sequence Response from the OC Source -> Destination	Public ApplicationLayerStandard	Public External System Under Test	

6.2.1.1 :ApplicationLayerStandard

Type: **Component** *Package:* Entity Emulation Overview

The :ApplicationLayerStandard represents the selected Application Layer Standard for a test run. It provides the direct interface to the system under test.

6.2.1.2 :C2CRIFunctionTag

Type: **Component** *Package:* Entity Emulation Overview

The :C2CRIFunctionTag represents the various tags that are used in the TMDDv303c test procedure scripts. The various OC related tags will request response and publication messages from the :EntityEmulationService.

Connections

Connector	Source	Target	Notes
NoteLink	Public	Public	
Source -> Destination	<anonymous>	:C2CRIFunctionTag	

6.2.1.3 :EntityEmulationService

Type: **Component** *Package:* Entity Emulation Overview

The :EntityEmulationService component represents an instance of the EntityEmulationServiceClass. This class represents the entities and devices that are being modeled by the C2C RI.

Connections

Connector	Source	Target	Notes
NoteLink	Public	Public	
Source -> Destination	<anonymous>	:EntityEmulationService	

6.2.1.4 :Tmddv303Controller

Type: **Component** *Package:* Entity Emulation Overview

The :Tmddv303Controller provides an interface to the selected :ApplicationLayerStandard. All TMDD dialog actions are translated to into the applicable ApplicationLayerStandard operations through this component.

6.2.1.5 TMDDv303cDatabase

Type: **Component** *Package:* Entity Emulation Overview

The TMDDv303c database contains data from the TMDD standard including the definition of needs, requirements, dialogs, messages, frames and elements. It also includes some information that allows the TMDD test suite to map subscriptions to publications. Finally, it includes a schema for maintaining entity emulation data.

Connections

Connector	Source	Target	Notes
NoteLink	Public	Public	
Source -> Destination	<anonymous>	TMDDv303cDatabase	

6.2.2 Entity Emulation Dialog Processing

Type: Package *Package:* Entity Emulation Overview

The following table defines how entity emulation handlers will process the incoming dialog/message combinations that it manages. It includes a description of how filtering is performed.

Dialog Name	Dialog Processing Description
dlActionLogRequest	This method processes a eventRequestMsg message received as part of a dlActionLogRequest dialog request from the External Center. The processing includes the following steps: verify message authentication, verify the message is consistent with the project NRTM, verify that required values are set (if applicable), generate filters for any filter data provided, send a request (with filters) to the entity data type and receive the response. If a verification step failed or no valid data could be retrieved, then return an error response message. Verify that the message information received for the entity data type conforms to the project specification. If it does not, then return an error response message. Otherwise create and return the actionLogMsg message received from the entity data source.
dlActionLogSubscription	This method processes a eventRequestMsg message received as part of a dlActionLogSubscription dialog Subscription request from the External Center. The processing includes the following steps: verify message authentication, verify the message is consistent with the project NRTM, verify that required values are set (if applicable), generate filters for any filter data provided, send a request (with filters) to the entity data type and receive the response. If a verification step failed or no valid data could be retrieved, then return an error response message. Verify that the message information received for the entity data type conforms to the project specification. If it

Dialog Name	Dialog Processing Description
	does not, then return an error response message. Otherwise create and return a c2cMessageReceipt message indicating that the subscription request was valid.
dlActionLogUpdate	This method processes a dlActionLogUpdate dialog update request. The message expected is the original subscription message that was used to set up the publication. The processing includes the following steps: verify message authentication, verify the message is consistent with the project NRTM, verify that required values are set (if applicable), generate filters for any filter data provided, send a request (with filters) to the entity data type and receive the response. If a verification step failed or no valid data could be retrieved, then return an error response message. Verify that the message information received for the entity data type conforms to the project specification. If it does not, then return an Exception. Otherwise create and return a actionLogMsg message.
dlArchivedDataProcessingDocumentationMetadataRequest	This method processes a archivedDataProcessingDocumentationMetadataRequestMsg message received as part of a dlArchivedDataProcessingDocumentationMetadataRequest dialog request from the External Center. The processing includes the following steps: verify message authentication, verify the message is consistent with the project NRTM, verify that required values are set (if applicable), generate filters for any filter data provided, send a request (with filters) to the entity data type and receive the response. If a verification step failed or no valid data could be retrieved, then return an error response message. Verify that the message information received for the entity data type conforms to the project specification. If it does not, then return an error response message. Otherwise create and return the archivedDataProcessingDocumentationMetadataMsg message received from the entity data source.
dlCCTVControlRequest	This method processes a dlCCTVControlRequest dialog control request. The processing includes the following steps: verify message authentication, verify the message is consistent with the project NRTM, verify that required values are set (if applicable), create a command and send it to the command queue. Check the command queue to see if this request is available for processing. If yes, then process the command and send a deviceControlResponseMsg message indicating the status of the message processing. If no, then send a deviceControlResponseMsg message indicating that the message has been queued. Process the currently available command from the queue. Once completed, send a request to the command queue to update the status of the command. If a verification step failed or no valid data could be retrieved, then return an error response message.
dlCenterActiveVerificationRequest	This method processes a centerActiveVerificationRequestMsg message received as part of a dlCenterActiveVerificationRequest dialog request from the External Center. The processing includes the following steps: verify message authentication, verify the message is consistent with the project NRTM, verify that required values are set (if applicable), send a request to the entity data type and receive the response. If a verification step failed or no valid data could be retrieved, then return an error response message. Verify that the message information received for the entity data type conforms to the project specification. If it does not, then return an error response message. Otherwise create and return the centerActiveVerificationMsg message received from the entity data source.

Dialog Name	Dialog Processing Description
dlCenterActiveVerificationUpdate	This method processes a dlCenterActiveVerificationUpdate dialog update request. The message expected is the original subscription message that was used to set up the publication. The processing includes the following steps: verify message authentication, verify the message is consistent with the project NRTM, verify that required values are set (if applicable), send a request to the entity data type and receive the response. If a verification step failed or no valid data could be retrieved, then return an error response message. Verify that the message information received for the entity data type conforms to the project specification. If it does not, then return an Exception. Otherwise create and return a CenterActiveVerificationMsg message.
dlCCTVInventoryRequest	This method processes a deviceInformationRequestMsg message received as part of a dlCCTVInventoryRequest dialog request from the External Center. The processing includes the following steps: verify message authentication, verify the message is consistent with the project NRTM, verify that required values are set (if applicable), generate filters for any filter data provided, send a request (with filters) to the entity data type and receive the response. If a verification step failed or no valid data could be retrieved, then return an error response message. Verify that the message information received for the entity data type conforms to the project specification. If it does not, then return an error response message. Otherwise create and return the cCTVInventoryMsg message received from the entity data source.
dlCCTVInventoryUpdate	This method processes a dlCCTVInventoryUpdate dialog update request. The message expected is the original subscription message that was used to set up the publication. The processing includes the following steps: verify message authentication, verify the message is consistent with the project NRTM, verify that required values are set (if applicable), generate filters for any filter data provided, send a request (with filters) to the entity data type and receive the response. If a verification step failed or no valid data could be retrieved, then return an error response message. Verify that the message information received for the entity data type conforms to the project specification. If it does not, then return an Exception. Otherwise create and return a cCTVInventoryMsg message.
dlCCTVStatusRequest	This method processes a deviceInformationRequestMsg message received as part of a dlCCTVStatusRequest dialog request from the External Center. The processing includes the following steps: verify message authentication, verify the message is consistent with the project NRTM, verify that required values are set (if applicable), generate filters for any filter data provided, send a request (with filters) to the entity data type and receive the response. If a verification step failed or no valid data could be retrieved, then return an error response message. Verify that the message information received for the entity data type conforms to the project specification. If it does not, then return an error response message. Otherwise create and return the cCTVStatusMsg message received from the entity data source.
dlCCTVStatusUpdate	This method processes a dlCCTVStatusUpdate dialog update request. The message expected is the original subscription message that was used to set up the publication. The processing includes the following steps: verify message authentication, verify the message is consistent with the project NRTM, verify that required values are set (if applicable), generate filters for any filter data provided, send a request (with filters) to the entity data type and receive the response. If a verification step failed or no valid data could be retrieved, then

Dialog Name	Dialog Processing Description
	return an error response message. Verify that the message information received for the entity data type conforms to the project specification. If it does not, then return an Exception. Otherwise create and return a cCTVStatusMsg message.
dlDetectorDataRequest	This method processes a detectorDataRequestMsg message received as part of a dlDetectorDataRequest dialog request from the External Center. The processing includes the following steps: verify message authentication, verify the message is consistent with the project NRTM, verify that required values are set (if applicable), generate filters for any filter data provided, send a request (with filters) to the entity data type and receive the response. If a verification step failed or no valid data could be retrieved, then return an error response message. Verify that the message information received for the entity data type conforms to the project specification. If it does not, then return an error response message. Otherwise create and return the detectorDataMsg message received from the entity data source.
dlDetectorDataSubscription	This method processes a deviceInformationRequestMsg message received as part of a dlDetectorDataSubscription dialog Subscription request from the External Center. The processing includes the following steps: verify message authentication, verify the message is consistent with the project NRTM, verify that required values are set (if applicable), generate filters for any filter data provided, send a request (with filters) to the entity data type and receive the response. If a verification step failed or no valid data could be retrieved, then return an error response message. Verify that the message information received for the entity data type conforms to the project specification. If it does not, then return an error response message. Otherwise create and return a c2cMessageReceipt message indicating that the subscription request was valid.
dlDetectorDataUpdate	This method processes a dlDetectorDataUpdate dialog update request. The message expected is the original subscription message that was used to set up the publication. The processing includes the following steps: verify message authentication, verify the message is consistent with the project NRTM, verify that required values are set (if applicable), generate filters for any filter data provided, send a request (with filters) to the entity data type and receive the response. If a verification step failed or no valid data could be retrieved, then return an error response message. Verify that the message information received for the entity data type conforms to the project specification. If it does not, then return an Exception. Otherwise create and return a detectorDataMsg message.
dlDetectorInventoryRequest	This method processes a deviceInformationRequestMsg message received as part of a dlDetectorInventoryRequest dialog request from the External Center. The processing includes the following steps: verify message authentication, verify the message is consistent with the project NRTM, verify that required values are set (if applicable), generate filters for any filter data provided, send a request (with filters) to the entity data type and receive the response. If a verification step failed or no valid data could be retrieved, then return an error response message. Verify that the message information received for the entity data type conforms to the project specification. If it does not, then return an error response message. Otherwise create and return the detectorInventoryMsg message received from the entity data source.
dlDetectorInventoryUpdate	This method processes a dlDetectorInventoryUpdate dialog update request. The message expected is the original subscription message that was used to set

Dialog Name	Dialog Processing Description
	up the publication. The processing includes the following steps: verify message authentication, verify the message is consistent with the project NRTM, verify that required values are set (if applicable), generate filters for any filter data provided, send a request (with filters) to the entity data type and receive the response. If a verification step failed or no valid data could be retrieved, then return an error response message. Verify that the message information received for the entity data type conforms to the project specification. If it does not, then return an Exception. Otherwise create and return a detectorInventoryMsg message.
dlDetectorMaintenanceHistoryRequest	This method processes a detectorMaintenanceHistoryRequestMsg message received as part of a dlDetectorMaintenanceHistoryRequest dialog request from the External Center. The processing includes the following steps: verify message authentication, verify the message is consistent with the project NRTM, verify that required values are set (if applicable), generate filters for any filter data provided, send a request (with filters) to the entity data type and receive the response. If a verification step failed or no valid data could be retrieved, then return an error response message. Verify that the message information received for the entity data type conforms to the project specification. If it does not, then return an error response message. Otherwise create and return the detectorMaintenanceHistoryMsg message received from the entity data source.
dlDetectorStatusRequest	This method processes a deviceInformationRequestMsg message received as part of a dlDetectorStatusRequest dialog request from the External Center. The processing includes the following steps: verify message authentication, verify the message is consistent with the project NRTM, verify that required values are set (if applicable), generate filters for any filter data provided, send a request (with filters) to the entity data type and receive the response. If a verification step failed or no valid data could be retrieved, then return an error response message. Verify that the message information received for the entity data type conforms to the project specification. If it does not, then return an error response message. Otherwise create and return the detectorStatusMsg message received from the entity data source.
dlDetectorStatusUpdate	This method processes a dlDetectorStatusUpdate dialog update request. The message expected is the original subscription message that was used to set up the publication. The processing includes the following steps: verify message authentication, verify the message is consistent with the project NRTM, verify that required values are set (if applicable), generate filters for any filter data provided, send a request (with filters) to the entity data type and receive the response. If a verification step failed or no valid data could be retrieved, then return an error response message. Verify that the message information received for the entity data type conforms to the project specification. If it does not, then return an Exception. Otherwise create and return a detectorStatusMsg message.
dlDeviceCancelControlRequest	This method processes a dlDeviceCancelControlRequest dialog cancel control request. The processing includes the following steps on the deviceCancelControlRequestMsg message received: verify message authentication, verify the message is consistent with the project NRTM, verify that required values are set (if applicable). Check the command queue to see if this request is available for cancellation. If yes, then cancel the command and send a deviceControlResponseMsg message indicating the request has been cancelled. If the request is already complete, then send a RR message

Dialog Name	Dialog Processing Description
	indicating that the message has been completed. If the request could not be found on the command queue, then return an error response message.
dlDeviceControlStatusRequest	This method processes a dlDeviceControlStatusRequest dialog control request. The processing includes the following steps: verify message authentication, verify the message is consistent with the project NRTM, verify that required values are set (if applicable), create a command and send it to the command queue. Check the command queue to see if this request is available for processing. If yes, then process the command and send a deviceControlResponseMsg message indicating the status of the message processing. If no, then send a deviceControlResponseMsg message indicating that the message has been queued. Process the currently available command from the queue. Once completed, send a request to the command queue to update the status of the command. If a verification step failed or no valid data could be retrieved, then return an error response message.
dlDeviceInformationSubscription	This method processes a deviceInformationRequestMsg message received as part of a dlDeviceInformationSubscription dialog Subscription request from the External Center. The processing includes the following steps: verify message authentication, verify the message is consistent with the project NRTM, verify that required values are set (if applicable), generate filters for any filter data provided, send a request (with filters) to the entity data type and receive the response. If a verification step failed or no valid data could be retrieved, then return an error response message. Verify that the message information received for the entity data type conforms to the project specification. If it does not, then return an error response message. Otherwise create and return a c2cMessageReceipt message indicating that the subscription request was valid.
dlDMSControlRequest	This method processes a dlDMSControlRequest dialog control request. The processing includes the following steps: verify message authentication, verify the message is consistent with the project NRTM, verify that required values are set (if applicable), create a command and send it to the command queue. Check the command queue to see if this request is available for processing. If yes, then process the command and send a deviceControlResponseMsg message indicating the status of the message processing. If no, then send a deviceControlResponseMsg message indicating that the message has been queued. Process the currently available command from the queue. Once completed, send a request to the command queue to update the status of the command. If a verification step failed or no valid data could be retrieved, then return an error response message.
dlDMSFontTableRequest	This method processes a dMSFontTableRequestMsg message received as part of a dlDMSFontTableRequest dialog request from the External Center. The processing includes the following steps: verify message authentication, verify the message is consistent with the project NRTM, verify that required values are set (if applicable), generate filters for any filter data provided, send a request (with filters) to the entity data type and receive the response. If a verification step failed or no valid data could be retrieved, then return an error response message. Verify that the message information received for the entity data type conforms to the project specification. If it does not, then return an error response message. Otherwise create and return the dMSFontTableMsg message received from the entity data source.

Dialog Name	Dialog Processing Description
dlDMSInventoryRequest	This method processes a deviceInformationRequestMsg message received as part of a dlDMSInventoryRequest dialog request from the External Center. The processing includes the following steps: verify message authentication, verify the message is consistent with the project NRTM, verify that required values are set (if applicable), generate filters for any filter data provided, send a request (with filters) to the entity data type and receive the response. If a verification step failed or no valid data could be retrieved, then return an error response message. Verify that the message information received for the entity data type conforms to the project specification. If it does not, then return an error response message. Otherwise create and return the dMSInventoryMsg message received from the entity data source.
dlDMSInventoryUpdate	This method processes a dlDMSInventoryUpdate dialog update request. The message expected is the original subscription message that was used to set up the publication. The processing includes the following steps: verify message authentication, verify the message is consistent with the project NRTM, verify that required values are set (if applicable), generate filters for any filter data provided, send a request (with filters) to the entity data type and receive the response. If a verification step failed or no valid data could be retrieved, then return an error response message. Verify that the message information received for the entity data type conforms to the project specification. If it does not, then return an Exception. Otherwise create and return a dMSInventoryMsg message.
dlDMSMessageAppearanceRequest	This method processes a dMSMessageAppearanceRequestMsg message received as part of a dlDMSMessageAppearanceRequest dialog request from the External Center. The processing includes the following steps: verify message authentication, verify the message is consistent with the project NRTM, verify that required values are set (if applicable), generate filters for any filter data provided, send a request (with filters) to the entity data type and receive the response. If a verification step failed or no valid data could be retrieved, then return an error response message. Verify that the message information received for the entity data type conforms to the project specification. If it does not, then return an error response message. Otherwise create and return the dMSMessageAppearanceMsg message received from the entity data source.
dlDMSMessageInventoryRequest	This method processes a dMSMessageInventoryRequestMsg message received as part of a dlDMSMessageInventoryRequest dialog request from the External Center. The processing includes the following steps: verify message authentication, verify the message is consistent with the project NRTM, verify that required values are set (if applicable), generate filters for any filter data provided, send a request (with filters) to the entity data type and receive the response. If a verification step failed or no valid data could be retrieved, then return an error response message. Verify that the message information received for the entity data type conforms to the project specification. If it does not, then return an error response message. Otherwise create and return the dMSMessageInventoryMsg message received from the entity data source.
dlDMSMessageInventorySubscription	This method processes a dMSMessageInventoryRequestMsg message received as part of a dlDMSMessageInventorySubscription dialog Subscription request from the External Center. The processing includes the following steps: verify message authentication, verify the message is consistent with the project NRTM, verify that required values are set (if applicable), generate filters for

Dialog Name	Dialog Processing Description
	any filter data provided, send a request (with filters) to the entity data type and receive the response. If a verification step failed or no valid data could be retrieved, then return an error response message. Verify that the message information received for the entity data type conforms to the project specification. If it does not, then return an error response message. Otherwise create and return a c2cMessageReceipt message indicating that the subscription request was valid.
dlDMSMessageInventoryUpdate	This method processes a dlDMSMessageInventoryUpdate dialog update request. The message expected is the original subscription message that was used to set up the publication. The processing includes the following steps: verify message authentication, verify the message is consistent with the project NRTM, verify that required values are set (if applicable), generate filters for any filter data provided, send a request (with filters) to the entity data type and receive the response. If a verification step failed or no valid data could be retrieved, then return an error response message. Verify that the message information received for the entity data type conforms to the project specification. If it does not, then return an Exception. Otherwise create and return a dMSMessageInventoryMsg message.
dlDMSPriorityQueueRequest	This method processes a dlDMSPriorityQueueRequest dialog control request. The processing includes the following steps: verify message authentication, verify the message is consistent with the project NRTM, verify that required values are set (if applicable), create a command and send it to the command queue. Check the command queue to see if this request is available for processing. If yes, then process the command and send a dMSPriorityQueueMsg message indicating the status of the message processing. If no, then send a dMSPriorityQueueMsg message indicating that the message has been queued. Process the currently available command from the queue. Once completed, send a request to the command queue to update the status of the command. If a verification step failed or no valid data could be retrieved, then return an error response message.
dlDMSStatusRequest	This method processes a deviceInformationRequestMsg message received as part of a dlDMSStatusRequest dialog request from the External Center. The processing includes the following steps: verify message authentication, verify the message is consistent with the project NRTM, verify that required values are set (if applicable), generate filters for any filter data provided, send a request (with filters) to the entity data type and receive the response. If a verification step failed or no valid data could be retrieved, then return an error response message. Verify that the message information received for the entity data type conforms to the project specification. If it does not, then return an error response message. Otherwise create and return the dMSStatusMsg message received from the entity data source.
dlDMSStatusUpdate	This method processes a dlDMSStatusUpdate dialog update request. The message expected is the original subscription message that was used to set up the publication. The processing includes the following steps: verify message authentication, verify the message is consistent with the project NRTM, verify that required values are set (if applicable), generate filters for any filter data provided, send a request (with filters) to the entity data type and receive the response. If a verification step failed or no valid data could be retrieved, then return an error response message. Verify that the message information received for the entity data type conforms to the project specification. If it does not, then return an Exception. Otherwise create and return a

Dialog Name	Dialog Processing Description
	dMSStatusMsg message.
dlESSInventoryRequest	This method processes a deviceInformationRequestMsg message received as part of a dlESSInventoryRequest dialog request from the External Center. The processing includes the following steps: verify message authentication, verify the message is consistent with the project NRTM, verify that required values are set (if applicable), generate filters for any filter data provided, send a request (with filters) to the entity data type and receive the response. If a verification step failed or no valid data could be retrieved, then return an error response message. Verify that the message information received for the entity data type conforms to the project specification. If it does not, then return an error response message. Otherwise create and return the eSSInventoryMsg message received from the entity data source.
dlESSInventoryUpdate	This method processes a dlESSInventoryUpdate dialog update request. The message expected is the original subscription message that was used to set up the publication. The processing includes the following steps: verify message authentication, verify the message is consistent with the project NRTM, verify that required values are set (if applicable), generate filters for any filter data provided, send a request (with filters) to the entity data type and receive the response. If a verification step failed or no valid data could be retrieved, then return an error response message. Verify that the message information received for the entity data type conforms to the project specification. If it does not, then return an Exception. Otherwise create and return a eSSInventoryMsg message.
dlESSObservationMetadataRequest	This method processes a deviceInformationRequestMsg message received as part of a dlESSObservationMetadataRequest dialog request from the External Center. The processing includes the following steps: verify message authentication, verify the message is consistent with the project NRTM, verify that required values are set (if applicable), generate filters for any filter data provided, send a request (with filters) to the entity data type and receive the response. If a verification step failed or no valid data could be retrieved, then return an error response message. Verify that the message information received for the entity data type conforms to the project specification. If it does not, then return an error response message. Otherwise create and return the eSSObservationMetadataMsg message received from the entity data source.
dlESSObservationReportRequest	This method processes a deviceInformationRequestMsg message received as part of a dlESSObservationReportRequest dialog request from the External Center. The processing includes the following steps: verify message authentication, verify the message is consistent with the project NRTM, verify that required values are set (if applicable), generate filters for any filter data provided, send a request (with filters) to the entity data type and receive the response. If a verification step failed or no valid data could be retrieved, then return an error response message. Verify that the message information received for the entity data type conforms to the project specification. If it does not, then return an error response message. Otherwise create and return the eSSObservationReportMsg message received from the entity data source.
dlESSObservationReportUpdate	This method processes a dlESSObservationReportUpdate dialog update request. The message expected is the original subscription message that was used to set up the publication. The processing includes the following steps: verify message authentication, verify the message is consistent with the project

Dialog Name	Dialog Processing Description
	NRTM, verify that required values are set (if applicable), generate filters for any filter data provided, send a request (with filters) to the entity data type and receive the response. If a verification step failed or no valid data could be retrieved, then return an error response message. Verify that the message information received for the entity data type conforms to the project specification. If it does not, then return an Exception. Otherwise create and return a eSSObservationReportMsg message.
dlESSStatusRequest	This method processes a deviceInformationRequestMsg message received as part of a dlESSStatusRequest dialog request from the External Center. The processing includes the following steps: verify message authentication, verify the message is consistent with the project NRTM, verify that required values are set (if applicable), generate filters for any filter data provided, send a request (with filters) to the entity data type and receive the response. If a verification step failed or no valid data could be retrieved, then return an error response message. Verify that the message information received for the entity data type conforms to the project specification. If it does not, then return an error response message. Otherwise create and return the eSSStatusMsg message received from the entity data source.
dlESSStatusUpdate	This method processes a dlESSStatusUpdate dialog update request. The message expected is the original subscription message that was used to set up the publication. The processing includes the following steps: verify message authentication, verify the message is consistent with the project NRTM, verify that required values are set (if applicable), generate filters for any filter data provided, send a request (with filters) to the entity data type and receive the response. If a verification step failed or no valid data could be retrieved, then return an error response message. Verify that the message information received for the entity data type conforms to the project specification. If it does not, then return an Exception. Otherwise create and return a eSSStatusMsg message.
dlEventIndexRequest	This method processes a eventRequestMsg message received as part of a dlEventIndexRequest dialog request from the External Center. The processing includes the following steps: verify message authentication, verify the message is consistent with the project NRTM, verify that required values are set (if applicable), generate filters for any filter data provided, send a request (with filters) to the entity data type and receive the response. If a verification step failed or no valid data could be retrieved, then return an error response message. Verify that the message information received for the entity data type conforms to the project specification. If it does not, then return an error response message. Otherwise create and return the eventIndexMsg message received from the entity data source.
dlEventIndexSubscription	This method processes a eventRequestMsg message received as part of a dlEventIndexSubscription dialog Subscription request from the External Center. The processing includes the following steps: verify message authentication, verify the message is consistent with the project NRTM, verify that required values are set (if applicable), generate filters for any filter data provided, send a request (with filters) to the entity data type and receive the response. If a verification step failed or no valid data could be retrieved, then return an error response message. Verify that the message information received for the entity data type conforms to the project specification. If it does not, then return an error response message. Otherwise create and return a

Dialog Name	Dialog Processing Description
	c2cMessageReceipt message indicating that the subscription request was valid.
dlEventIndexUpdate	This method processes a dlEventIndexUpdate dialog update request. The message expected is the original subscription message that was used to set up the publication. The processing includes the following steps: verify message authentication, verify the message is consistent with the project NRTM, verify that required values are set (if applicable), generate filters for any filter data provided, send a request (with filters) to the entity data type and receive the response. If a verification step failed or no valid data could be retrieved, then return an error response message. Verify that the message information received for the entity data type conforms to the project specification. If it does not, then return an Exception. Otherwise create and return a eventIndexMsg message.
dlFullEventUpdateRequest	This method processes a eventRequestMsg message received as part of a dlFullEventUpdateRequest dialog request from the External Center. The processing includes the following steps: verify message authentication, verify the message is consistent with the project NRTM, verify that required values are set (if applicable), generate filters for any filter data provided, send a request (with filters) to the entity data type and receive the response. If a verification step failed or no valid data could be retrieved, then return an error response message. Verify that the message information received for the entity data type conforms to the project specification. If it does not, then return an error response message. Otherwise create and return the fEUMsg message received from the entity data source.
dlFullEventUpdateSubscription	This method processes a eventRequestMsg message received as part of a dlFullEventUpdateSubscription dialog Subscription request from the External Center. The processing includes the following steps: verify message authentication, verify the message is consistent with the project NRTM, verify that required values are set (if applicable), generate filters for any filter data provided, send a request (with filters) to the entity data type and receive the response. If a verification step failed or no valid data could be retrieved, then return an error response message. Verify that the message information received for the entity data type conforms to the project specification. If it does not, then return an error response message. Otherwise create and return a c2cMessageReceipt message indicating that the subscription request was valid.
dlFullEventUpdateUpdate	This method processes a dlFullEventUpdateUpdate dialog update request. The message expected is the original subscription message that was used to set up the publication. The processing includes the following steps: verify message authentication, verify the message is consistent with the project NRTM, verify that required values are set (if applicable), generate filters for any filter data provided, send a request (with filters) to the entity data type and receive the response. If a verification step failed or no valid data could be retrieved, then return an error response message. Verify that the message information received for the entity data type conforms to the project specification. If it does not, then return an Exception. Otherwise create and return a fEUMsg message.
dlGateControlRequest	This method processes a dlGateControlRequest dialog control request. The processing includes the following steps: verify message authentication, verify the message is consistent with the project NRTM, verify that required values are set (if applicable), create a command and send it to the command queue. Check the command queue to see if this request is available for processing. If

Dialog Name	Dialog Processing Description
	yes, then process the command and send a deviceControlResponseMsg message indicating the status of the message processing. If no, then send a deviceControlResponseMsg message indicating that the message has been queued. Process the currently available command from the queue. Once completed, send a request to the command queue to update the status of the command. If a verification step failed or no valid data could be retrieved, then return an error response message.
dlGateControlScheduleRequest	This method processes a dlGateControlScheduleRequest dialog control request. The processing includes the following steps: verify message authentication, verify the message is consistent with the project NRTM, verify that required values are set (if applicable), create a command and send it to the command queue. Check the command queue to see if this request is available for processing. If yes, then process the command and send a gateControlScheduleMsg message indicating the status of the message processing. If no, then send a gateControlScheduleMsg message indicating that the message has been queued. Process the currently available command from the queue. Once completed, send a request to the command queue to update the status of the command. If a verification step failed or no valid data could be retrieved, then return an error response message.
dlGateInventoryRequest	This method processes a deviceInformationRequestMsg message received as part of a dlGateInventoryRequest dialog request from the External Center. The processing includes the following steps: verify message authentication, verify the message is consistent with the project NRTM, verify that required values are set (if applicable), generate filters for any filter data provided, send a request (with filters) to the entity data type and receive the response. If a verification step failed or no valid data could be retrieved, then return an error response message. Verify that the message information received for the entity data type conforms to the project specification. If it does not, then return an error response message. Otherwise create and return the gateInventoryMsg message received from the entity data source.
dlGateInventoryUpdate	This method processes a dlGateInventoryUpdate dialog update request. The message expected is the original subscription message that was used to set up the publication. The processing includes the following steps: verify message authentication, verify the message is consistent with the project NRTM, verify that required values are set (if applicable), generate filters for any filter data provided, send a request (with filters) to the entity data type and receive the response. If a verification step failed or no valid data could be retrieved, then return an error response message. Verify that the message information received for the entity data type conforms to the project specification. If it does not, then return an Exception. Otherwise create and return a gateInventoryMsg message.
dlGateStatusRequest	This method processes a deviceInformationRequestMsg message received as part of a dlGateStatusRequest dialog request from the External Center. The processing includes the following steps: verify message authentication, verify the message is consistent with the project NRTM, verify that required values are set (if applicable), generate filters for any filter data provided, send a request (with filters) to the entity data type and receive the response. If a verification step failed or no valid data could be retrieved, then return an error response message. Verify that the message information received for the entity data type conforms to the project specification. If it does not, then return an error response message. Otherwise create and return the gateStatusMsg

Dialog Name	Dialog Processing Description
	message received from the entity data source.
dlGateStatusUpdate	This method processes a dlGateStatusUpdate dialog update request. The message expected is the original subscription message that was used to set up the publication. The processing includes the following steps: verify message authentication, verify the message is consistent with the project NRTM, verify that required values are set (if applicable), generate filters for any filter data provided, send a request (with filters) to the entity data type and receive the response. If a verification step failed or no valid data could be retrieved, then return an error response message. Verify that the message information received for the entity data type conforms to the project specification. If it does not, then return an Exception. Otherwise create and return a gateStatusMsg message.
dlHARControlRequest	This method processes a dlHARControlRequest dialog control request. The processing includes the following steps: verify message authentication, verify the message is consistent with the project NRTM, verify that required values are set (if applicable), create a command and send it to the command queue. Check the command queue to see if this request is available for processing. If yes, then process the command and send a deviceControlResponseMsg message indicating the status of the message processing. If no, then send a deviceControlResponseMsg message indicating that the message has been queued. Process the currently available command from the queue. Once completed, send a request to the command queue to update the status of the command. If a verification step failed or no valid data could be retrieved, then return an error response message.
dlHARControlScheduleRequest	This method processes a dlHARControlScheduleRequest dialog control request. The processing includes the following steps: verify message authentication, verify the message is consistent with the project NRTM, verify that required values are set (if applicable), create a command and send it to the command queue. Check the command queue to see if this request is available for processing. If yes, then process the command and send a hARControlScheduleMsg message indicating the status of the message processing. If no, then send a hARControlScheduleMsg message indicating that the message has been queued. Process the currently available command from the queue. Once completed, send a request to the command queue to update the status of the command. If a verification step failed or no valid data could be retrieved, then return an error response message.
dlHARInventoryRequest	This method processes a deviceInformationRequestMsg message received as part of a dlHARInventoryRequest dialog request from the External Center. The processing includes the following steps: verify message authentication, verify the message is consistent with the project NRTM, verify that required values are set (if applicable), generate filters for any filter data provided, send a request (with filters) to the entity data type and receive the response. If a verification step failed or no valid data could be retrieved, then return an error response message. Verify that the message information received for the entity data type conforms to the project specification. If it does not, then return an error response message. Otherwise create and return the hARInventoryMsg message received from the entity data source.
dlHARInventoryUpdate	This method processes a dlHARInventoryUpdate dialog update request. The message expected is the original subscription message that was used to set up the publication. The processing includes the following steps: verify message

Dialog Name	Dialog Processing Description
	authentication, verify the message is consistent with the project NRTM, verify that required values are set (if applicable), generate filters for any filter data provided, send a request (with filters) to the entity data type and receive the response. If a verification step failed or no valid data could be retrieved, then return an error response message. Verify that the message information received for the entity data type conforms to the project specification. If it does not, then return an Exception. Otherwise create and return a hARInventoryMsg message.
dlHARMessageInventoryRequest	This method processes a deviceInformationRequestMsg message received as part of a dlHARMessageInventoryRequest dialog request from the External Center. The processing includes the following steps: verify message authentication, verify the message is consistent with the project NRTM, verify that required values are set (if applicable), generate filters for any filter data provided, send a request (with filters) to the entity data type and receive the response. If a verification step failed or no valid data could be retrieved, then return an error response message. Verify that the message information received for the entity data type conforms to the project specification. If it does not, then return an error response message. Otherwise create and return the hARMessageInventoryMsg message received from the entity data source.
dlHARMessageInventoryUpdate	This method processes a dlHARMessageInventoryUpdate dialog update request. The message expected is the original subscription message that was used to set up the publication. The processing includes the following steps: verify message authentication, verify the message is consistent with the project NRTM, verify that required values are set (if applicable), generate filters for any filter data provided, send a request (with filters) to the entity data type and receive the response. If a verification step failed or no valid data could be retrieved, then return an error response message. Verify that the message information received for the entity data type conforms to the project specification. If it does not, then return an Exception. Otherwise create and return a hARMessageInventoryMsg message.
dlHARPriorityQueueRequest	This method processes a dlHARPriorityQueueRequest dialog control request. The processing includes the following steps: verify message authentication, verify the message is consistent with the project NRTM, verify that required values are set (if applicable), create a command and send it to the command queue. Check the command queue to see if this request is available for processing. If yes, then process the command and send a hARPriorityQueueMsg message indicating the status of the message processing. If no, then send a hARPriorityQueueMsg message indicating that the message has been queued. Process the currently available command from the queue. Once completed, send a request to the command queue to update the status of the command. If a verification step failed or no valid data could be retrieved, then return an error response message.
dlHARStatusRequest	This method processes a deviceInformationRequestMsg message received as part of a dlHARStatusRequest dialog request from the External Center. The processing includes the following steps: verify message authentication, verify the message is consistent with the project NRTM, verify that required values are set (if applicable), generate filters for any filter data provided, send a request (with filters) to the entity data type and receive the response. If a verification step failed or no valid data could be retrieved, then return an error response message. Verify that the message information received for the entity data type conforms to the project specification. If it does not, then return an

Dialog Name	Dialog Processing Description
	error response message. Otherwise create and return the hARStatusMsg message received from the entity data source.
dlHARStatusUpdate	This method processes a dlHARStatusUpdate dialog update request. The message expected is the original subscription message that was used to set up the publication. The processing includes the following steps: verify message authentication, verify the message is consistent with the project NRTM, verify that required values are set (if applicable), generate filters for any filter data provided, send a request (with filters) to the entity data type and receive the response. If a verification step failed or no valid data could be retrieved, then return an error response message. Verify that the message information received for the entity data type conforms to the project specification. If it does not, then return an Exception. Otherwise create and return a hARStatusMsg message.
dlIntersectionSignalControlRequest	This method processes a dlIntersectionSignalControlRequest dialog control request. The processing includes the following steps: verify message authentication, verify the message is consistent with the project NRTM, verify that required values are set (if applicable), create a command and send it to the command queue. Check the command queue to see if this request is available for processing. If yes, then process the command and send a intersectionSignalControlResponseMsg message indicating the status of the message processing. If no, then send a intersectionSignalControlResponseMsg message indicating that the message has been queued. Process the currently available command from the queue. Once completed, send a request to the command queue to update the status of the command. If a verification step failed or no valid data could be retrieved, then return an error response message.
dlIntersectionSignalControlScheduleRequest	This method processes a deviceInformationRequestMsg message received as part of a dlIntersectionSignalControlScheduleRequest dialog request from the External Center. The processing includes the following steps: verify message authentication, verify the message is consistent with the project NRTM, verify that required values are set (if applicable), generate filters for any filter data provided, send a request (with filters) to the entity data type and receive the response. If a verification step failed or no valid data could be retrieved, then return an error response message. Verify that the message information received for the entity data type conforms to the project specification. If it does not, then return an error response message. Otherwise create and return the intersectionSignalControlScheduleMsg message received from the entity data source.
dlIntersectionSignalControlScheduleUpdate	This method processes a dlIntersectionSignalControlScheduleUpdate dialog update request. The message expected is the original subscription message that was used to set up the publication. The processing includes the following steps: verify message authentication, verify the message is consistent with the project NRTM, verify that required values are set (if applicable), generate filters for any filter data provided, send a request (with filters) to the entity data type and receive the response. If a verification step failed or no valid data could be retrieved, then return an error response message. Verify that the message information received for the entity data type conforms to the project specification. If it does not, then return an Exception. Otherwise create and return a intersectionSignalControlScheduleMsg message.
dlIntersectionSignalInventoryReq	This method processes a deviceInformationRequestMsg message received as

Dialog Name	Dialog Processing Description
uest	part of a dlIntersectionSignalInventoryRequest dialog request from the External Center. The processing includes the following steps: verify message authentication, verify the message is consistent with the project NRTM, verify that required values are set (if applicable), generate filters for any filter data provided, send a request (with filters) to the entity data type and receive the response. If a verification step failed or no valid data could be retrieved, then return an error response message. Verify that the message information received for the entity data type conforms to the project specification. If it does not, then return an error response message. Otherwise create and return the intersectionSignalInventoryMsg message received from the entity data source.
dlIntersectionSignalInventoryUpdate	This method processes a dlIntersectionSignalInventoryUpdate dialog update request. The message expected is the original subscription message that was used to set up the publication. The processing includes the following steps: verify message authentication, verify the message is consistent with the project NRTM, verify that required values are set (if applicable), generate filters for any filter data provided, send a request (with filters) to the entity data type and receive the response. If a verification step failed or no valid data could be retrieved, then return an error response message. Verify that the message information received for the entity data type conforms to the project specification. If it does not, then return an Exception. Otherwise create and return a intersectionSignalInventoryMsg message.
dlIntersectionSignalPriorityQueueRequest	This method processes a dlIntersectionSignalPriorityQueueRequest dialog control request. The processing includes the following steps: verify message authentication, verify the message is consistent with the project NRTM, verify that required values are set (if applicable), create a command and send it to the command queue. Check the command queue to see if this request is available for processing. If yes, then process the command and send a intersectionSignalPriorityQueueMsg message indicating the status of the message processing. If no, then send a intersectionSignalPriorityQueueMsg message indicating that the message has been queued. Process the currently available command from the queue. Once completed, send a request to the command queue to update the status of the command. If a verification step failed or no valid data could be retrieved, then return an error response message.
dlIntersectionSignalStatusRequest	This method processes a deviceInformationRequestMsg message received as part of a dlIntersectionSignalStatusRequest dialog request from the External Center. The processing includes the following steps: verify message authentication, verify the message is consistent with the project NRTM, verify that required values are set (if applicable), generate filters for any filter data provided, send a request (with filters) to the entity data type and receive the response. If a verification step failed or no valid data could be retrieved, then return an error response message. Verify that the message information received for the entity data type conforms to the project specification. If it does not, then return an error response message. Otherwise create and return the intersectionSignalStatusMsg message received from the entity data source.
dlIntersectionSignalStatusUpdate	This method processes a dlIntersectionSignalStatusUpdate dialog update request. The message expected is the original subscription message that was used to set up the publication. The processing includes the following steps: verify message authentication, verify the message is consistent with the project NRTM, verify that required values are set (if applicable), generate filters for

Dialog Name	Dialog Processing Description
	any filter data provided, send a request (with filters) to the entity data type and receive the response. If a verification step failed or no valid data could be retrieved, then return an error response message. Verify that the message information received for the entity data type conforms to the project specification. If it does not, then return an Exception. Otherwise create and return a intersectionSignalStatusMsg message.
dlIntersectionSignalTimingPatternInventoryRequest	This method processes a intersectionSignalTimingPatternInventoryRequestMsg message received as part of a dlIntersectionSignalTimingPatternInventoryRequest dialog request from the External Center. The processing includes the following steps: verify message authentication, verify the message is consistent with the project NRTM, verify that required values are set (if applicable), generate filters for any filter data provided, send a request (with filters) to the entity data type and receive the response. If a verification step failed or no valid data could be retrieved, then return an error response message. Verify that the message information received for the entity data type conforms to the project specification. If it does not, then return an error response message. Otherwise create and return the intersectionSignalTimingPatternInventoryMsg message received from the entity data source.
dlIntersectionSignalTimingPatternInventorySubscription	This method processes a intersectionSignalTimingPatternInventoryRequestMsg message received as part of a dlIntersectionSignalTimingPatternInventorySubscription dialog Subscription request from the External Center. The processing includes the following steps: verify message authentication, verify the message is consistent with the project NRTM, verify that required values are set (if applicable), generate filters for any filter data provided, send a request (with filters) to the entity data type and receive the response. If a verification step failed or no valid data could be retrieved, then return an error response message. Verify that the message information received for the entity data type conforms to the project specification. If it does not, then return an error response message. Otherwise create and return a c2cMessageReceipt message indicating that the subscription request was valid.
dlIntersectionSignalTimingPatternInventoryUpdate	This method processes a dlIntersectionSignalTimingPatternInventoryUpdate dialog update request. The message expected is the original subscription message that was used to set up the publication. The processing includes the following steps: verify message authentication, verify the message is consistent with the project NRTM, verify that required values are set (if applicable), generate filters for any filter data provided, send a request (with filters) to the entity data type and receive the response. If a verification step failed or no valid data could be retrieved, then return an error response message. Verify that the message information received for the entity data type conforms to the project specification. If it does not, then return an Exception. Otherwise create and return a intersectionSignalTimingPatternInventoryMsg message.
dlLCSControlRequest	This method processes a dlLCSControlRequest dialog control request. The processing includes the following steps: verify message authentication, verify the message is consistent with the project NRTM, verify that required values are set (if applicable), create a command and send it to the command queue. Check the command queue to see if this request is available for processing. If yes, then process the command and send a deviceControlResponseMsg message indicating the status of the message processing. If no, then send a

Dialog Name	Dialog Processing Description
	deviceControlResponseMsg message indicating that the message has been queued. Process the currently available command from the queue. Once completed, send a request to the command queue to update the status of the command. If a verification step failed or no valid data could be retrieved, then return an error response message.
dlLCSControlScheduleRequest	This method processes a deviceInformationRequestMsg message received as part of a dlLCSControlScheduleRequest dialog request from the External Center. The processing includes the following steps: verify message authentication, verify the message is consistent with the project NRTM, verify that required values are set (if applicable), generate filters for any filter data provided, send a request (with filters) to the entity data type and receive the response. If a verification step failed or no valid data could be retrieved, then return an error response message. Verify that the message information received for the entity data type conforms to the project specification. If it does not, then return an error response message. Otherwise create and return the lCSControlScheduleMsg message received from the entity data source.
dlLCSControlScheduleUpdate	This method processes a dlLCSControlScheduleUpdate dialog update request. The message expected is the original subscription message that was used to set up the publication. The processing includes the following steps: verify message authentication, verify the message is consistent with the project NRTM, verify that required values are set (if applicable), generate filters for any filter data provided, send a request (with filters) to the entity data type and receive the response. If a verification step failed or no valid data could be retrieved, then return an error response message. Verify that the message information received for the entity data type conforms to the project specification. If it does not, then return an Exception. Otherwise create and return a lCSControlScheduleMsg message.
dlLCSInventoryRequest	This method processes a deviceInformationRequestMsg message received as part of a dlLCSInventoryRequest dialog request from the External Center. The processing includes the following steps: verify message authentication, verify the message is consistent with the project NRTM, verify that required values are set (if applicable), generate filters for any filter data provided, send a request (with filters) to the entity data type and receive the response. If a verification step failed or no valid data could be retrieved, then return an error response message. Verify that the message information received for the entity data type conforms to the project specification. If it does not, then return an error response message. Otherwise create and return the lCSInventoryMsg message received from the entity data source.
dlLCSInventoryUpdate	This method processes a dlLCSInventoryUpdate dialog update request. The message expected is the original subscription message that was used to set up the publication. The processing includes the following steps: verify message authentication, verify the message is consistent with the project NRTM, verify that required values are set (if applicable), generate filters for any filter data provided, send a request (with filters) to the entity data type and receive the response. If a verification step failed or no valid data could be retrieved, then return an error response message. Verify that the message information received for the entity data type conforms to the project specification. If it does not, then return an Exception. Otherwise create and return a lCSInventoryMsg message.

Dialog Name	Dialog Processing Description
dlLCSStatusRequest	This method processes a deviceInformationRequestMsg message received as part of a dlLCSStatusRequest dialog request from the External Center. The processing includes the following steps: verify message authentication, verify the message is consistent with the project NRTM, verify that required values are set (if applicable), generate filters for any filter data provided, send a request (with filters) to the entity data type and receive the response. If a verification step failed or no valid data could be retrieved, then return an error response message. Verify that the message information received for the entity data type conforms to the project specification. If it does not, then return an error response message. Otherwise create and return the lCSStatusMsg message received from the entity data source.
dlLCSStatusUpdate	This method processes a dlLCSStatusUpdate dialog update request. The message expected is the original subscription message that was used to set up the publication. The processing includes the following steps: verify message authentication, verify the message is consistent with the project NRTM, verify that required values are set (if applicable), generate filters for any filter data provided, send a request (with filters) to the entity data type and receive the response. If a verification step failed or no valid data could be retrieved, then return an error response message. Verify that the message information received for the entity data type conforms to the project specification. If it does not, then return an Exception. Otherwise create and return a lCSStatusMsg message.
dlLinkInventoryRequest	This method processes a trafficNetworkInformationRequestMsg message received as part of a dlLinkInventoryRequest dialog request from the External Center. The processing includes the following steps: verify message authentication, verify the message is consistent with the project NRTM, verify that required values are set (if applicable), generate filters for any filter data provided, send a request (with filters) to the entity data type and receive the response. If a verification step failed or no valid data could be retrieved, then return an error response message. Verify that the message information received for the entity data type conforms to the project specification. If it does not, then return an error response message. Otherwise create and return the LinkInventoryMsg message received from the entity data source.
dlLinkInventoryUpdate	This method processes a dlLinkInventoryUpdate dialog update request. The message expected is the original subscription message that was used to set up the publication. The processing includes the following steps: verify message authentication, verify the message is consistent with the project NRTM, verify that required values are set (if applicable), generate filters for any filter data provided, send a request (with filters) to the entity data type and receive the response. If a verification step failed or no valid data could be retrieved, then return an error response message. Verify that the message information received for the entity data type conforms to the project specification. If it does not, then return an Exception. Otherwise create and return a LinkInventoryMsg message.
dlLinkStatusRequest	This method processes a trafficNetworkInformationRequestMsg message received as part of a dlLinkStatusRequest dialog request from the External Center. The processing includes the following steps: verify message authentication, verify the message is consistent with the project NRTM, verify that required values are set (if applicable), generate filters for any filter data provided, send a request (with filters) to the entity data type and receive the

Dialog Name	Dialog Processing Description
	response. If a verification step failed or no valid data could be retrieved, then return an error response message. Verify that the message information received for the entity data type conforms to the project specification. If it does not, then return an error response message. Otherwise create and return the LinkStatusMsg message received from the entity data source.
dlLinkStatusUpdate	This method processes a dlLinkStatusUpdate dialog update request. The message expected is the original subscription message that was used to set up the publication. The processing includes the following steps: verify message authentication, verify the message is consistent with the project NRTM, verify that required values are set (if applicable), generate filters for any filter data provided, send a request (with filters) to the entity data type and receive the response. If a verification step failed or no valid data could be retrieved, then return an error response message. Verify that the message information received for the entity data type conforms to the project specification. If it does not, then return an Exception. Otherwise create and return a LinkStatusMsg message.
dlNodeInventoryRequest	This method processes a trafficNetworkInformationRequestMsg message received as part of a dlNodeInventoryRequest dialog request from the External Center. The processing includes the following steps: verify message authentication, verify the message is consistent with the project NRTM, verify that required values are set (if applicable), generate filters for any filter data provided, send a request (with filters) to the entity data type and receive the response. If a verification step failed or no valid data could be retrieved, then return an error response message. Verify that the message information received for the entity data type conforms to the project specification. If it does not, then return an error response message. Otherwise create and return the NodeInventoryMsg message received from the entity data source.
dlNodeInventoryUpdate	This method processes a dlNodeInventoryUpdate dialog update request. The message expected is the original subscription message that was used to set up the publication. The processing includes the following steps: verify message authentication, verify the message is consistent with the project NRTM, verify that required values are set (if applicable), generate filters for any filter data provided, send a request (with filters) to the entity data type and receive the response. If a verification step failed or no valid data could be retrieved, then return an error response message. Verify that the message information received for the entity data type conforms to the project specification. If it does not, then return an Exception. Otherwise create and return a NodeInventoryMsg message.
dlNodeStatusRequest	This method processes a trafficNetworkInformationRequestMsg message received as part of a dlNodeStatusRequest dialog request from the External Center. The processing includes the following steps: verify message authentication, verify the message is consistent with the project NRTM, verify that required values are set (if applicable), generate filters for any filter data provided, send a request (with filters) to the entity data type and receive the response. If a verification step failed or no valid data could be retrieved, then return an error response message. Verify that the message information received for the entity data type conforms to the project specification. If it does not, then return an error response message. Otherwise create and return the NodeStatusMsg message received from the entity data source.

Dialog Name	Dialog Processing Description
dlNodeStatusUpdate	This method processes a dlNodeStatusUpdate dialog update request. The message expected is the original subscription message that was used to set up the publication. The processing includes the following steps: verify message authentication, verify the message is consistent with the project NRTM, verify that required values are set (if applicable), generate filters for any filter data provided, send a request (with filters) to the entity data type and receive the response. If a verification step failed or no valid data could be retrieved, then return an error response message. Verify that the message information received for the entity data type conforms to the project specification. If it does not, then return an Exception. Otherwise create and return a NodeStatusMsg message.
dlRampMeterControlRequest	This method processes a dlRampMeterControlRequest dialog control request. The processing includes the following steps: verify message authentication, verify the message is consistent with the project NRTM, verify that required values are set (if applicable), create a command and send it to the command queue. Check the command queue to see if this request is available for processing. If yes, then process the command and send a deviceControlResponseMsg message indicating the status of the message processing. If no, then send a deviceControlResponseMsg message indicating that the message has been queued. Process the currently available command from the queue. Once completed, send a request to the command queue to update the status of the command. If a verification step failed or no valid data could be retrieved, then return an error response message.
dlRampMeterControlScheduleRequest	This method processes a deviceInformationRequestMsg message received as part of a dlRampMeterControlScheduleRequest dialog request from the External Center. The processing includes the following steps: verify message authentication, verify the message is consistent with the project NRTM, verify that required values are set (if applicable), generate filters for any filter data provided, send a request (with filters) to the entity data type and receive the response. If a verification step failed or no valid data could be retrieved, then return an error response message. Verify that the message information received for the entity data type conforms to the project specification. If it does not, then return an error response message. Otherwise create and return the rampMeterControlScheduleMsg message received from the entity data source.
dlRampMeterControlScheduleUpdate	This method processes a dlRampMeterControlScheduleUpdate dialog update request. The message expected is the original subscription message that was used to set up the publication. The processing includes the following steps: verify message authentication, verify the message is consistent with the project NRTM, verify that required values are set (if applicable), generate filters for any filter data provided, send a request (with filters) to the entity data type and receive the response. If a verification step failed or no valid data could be retrieved, then return an error response message. Verify that the message information received for the entity data type conforms to the project specification. If it does not, then return an Exception. Otherwise create and return a rampMeterControlScheduleMsg message.
dlRampMeterInventoryRequest	This method processes a deviceInformationRequestMsg message received as part of a dlRampMeterInventoryRequest dialog request from the External Center. The processing includes the following steps: verify message authentication, verify the message is consistent with the project NRTM, verify

Dialog Name	Dialog Processing Description
	that required values are set (if applicable), generate filters for any filter data provided, send a request (with filters) to the entity data type and receive the response. If a verification step failed or no valid data could be retrieved, then return an error response message. Verify that the message information received for the entity data type conforms to the project specification. If it does not, then return an error response message. Otherwise create and return the rampMeterInventoryMsg message received from the entity data source.
dlRampMeterInventoryUpdate	This method processes a dlRampMeterInventoryUpdate dialog update request. The message expected is the original subscription message that was used to set up the publication. The processing includes the following steps: verify message authentication, verify the message is consistent with the project NRTM, verify that required values are set (if applicable), generate filters for any filter data provided, send a request (with filters) to the entity data type and receive the response. If a verification step failed or no valid data could be retrieved, then return an error response message. Verify that the message information received for the entity data type conforms to the project specification. If it does not, then return an Exception. Otherwise create and return a rampMeterInventoryMsg message.
dlRampMeterPlanInventoryRequest	This method processes a deviceInformationRequestMsg message received as part of a dlRampMeterPlanInventoryRequest dialog request from the External Center. The processing includes the following steps: verify message authentication, verify the message is consistent with the project NRTM, verify that required values are set (if applicable), generate filters for any filter data provided, send a request (with filters) to the entity data type and receive the response. If a verification step failed or no valid data could be retrieved, then return an error response message. Verify that the message information received for the entity data type conforms to the project specification. If it does not, then return an error response message. Otherwise create and return the rampMeterPlanInventoryMsg message received from the entity data source.
dlRampMeterPlanInventorySubscription	This method processes a deviceInformationRequestMsg message received as part of a dlRampMeterPlanInventorySubscription dialog Subscription request from the External Center. The processing includes the following steps: verify message authentication, verify the message is consistent with the project NRTM, verify that required values are set (if applicable), generate filters for any filter data provided, send a request (with filters) to the entity data type and receive the response. If a verification step failed or no valid data could be retrieved, then return an error response message. Verify that the message information received for the entity data type conforms to the project specification. If it does not, then return an error response message. Otherwise create and return a c2cMessageReceipt message indicating that the subscription request was valid.
dlRampMeterPlanInventoryUpdate	This method processes a dlRampMeterPlanInventoryUpdate dialog update request. The message expected is the original subscription message that was used to set up the publication. The processing includes the following steps: verify message authentication, verify the message is consistent with the project NRTM, verify that required values are set (if applicable), generate filters for any filter data provided, send a request (with filters) to the entity data type and receive the response. If a verification step failed or no valid data could be retrieved, then return an error response message. Verify that the message information received for the entity data type conforms to the project

Dialog Name	Dialog Processing Description
	specification. If it does not, then return an Exception. Otherwise create and return a rampMeterPlanInventoryMsg message.
dlRampMeterPriorityQueueRequest	This method processes a dlRampMeterPriorityQueueRequest dialog control request. The processing includes the following steps: verify message authentication, verify the message is consistent with the project NRTM, verify that required values are set (if applicable), create a command and send it to the command queue. Check the command queue to see if this request is available for processing. If yes, then process the command and send a rampMeterPriorityQueueMsg message indicating the status of the message processing. If no, then send a rampMeterPriorityQueueMsg message indicating that the message has been queued. Process the currently available command from the queue. Once completed, send a request to the command queue to update the status of the command. If a verification step failed or no valid data could be retrieved, then return an error response message.
dlRampMeterStatusRequest	This method processes a deviceInformationRequestMsg message received as part of a dlRampMeterStatusRequest dialog request from the External Center. The processing includes the following steps: verify message authentication, verify the message is consistent with the project NRTM, verify that required values are set (if applicable), generate filters for any filter data provided, send a request (with filters) to the entity data type and receive the response. If a verification step failed or no valid data could be retrieved, then return an error response message. Verify that the message information received for the entity data type conforms to the project specification. If it does not, then return an error response message. Otherwise create and return the rampMeterStatusMsg message received from the entity data source.
dlRampMeterStatusUpdate	This method processes a dlRampMeterStatusUpdate dialog update request. The message expected is the original subscription message that was used to set up the publication. The processing includes the following steps: verify message authentication, verify the message is consistent with the project NRTM, verify that required values are set (if applicable), generate filters for any filter data provided, send a request (with filters) to the entity data type and receive the response. If a verification step failed or no valid data could be retrieved, then return an error response message. Verify that the message information received for the entity data type conforms to the project specification. If it does not, then return an Exception. Otherwise create and return a rampMeterStatusMsg message.
dlSectionControlRequest	This method processes a dlSectionControlRequest dialog control request. The processing includes the following steps: verify message authentication, verify the message is consistent with the project NRTM, verify that required values are set (if applicable), create a command and send it to the command queue. Check the command queue to see if this request is available for processing. If yes, then process the command and send a sectionControlResponseMsg message indicating the status of the message processing. If no, then send a sectionControlResponseMsg message indicating that the message has been queued. Process the currently available command from the queue. Once completed, send a request to the command queue to update the status of the command. If a verification step failed or no valid data could be retrieved, then return an error response message.
dlSectionControlScheduleRequest	This method processes a deviceInformationRequestMsg message received as part of a dlSectionControlScheduleRequest dialog request from the External

Dialog Name	Dialog Processing Description
	Center. The processing includes the following steps: verify message authentication, verify the message is consistent with the project NRTM, verify that required values are set (if applicable), generate filters for any filter data provided, send a request (with filters) to the entity data type and receive the response. If a verification step failed or no valid data could be retrieved, then return an error response message. Verify that the message information received for the entity data type conforms to the project specification. If it does not, then return an error response message. Otherwise create and return the sectionControlScheduleMsg message received from the entity data source.
dlSectionControlScheduleUpdate	This method processes a dlSectionControlScheduleUpdate dialog update request. The message expected is the original subscription message that was used to set up the publication. The processing includes the following steps: verify message authentication, verify the message is consistent with the project NRTM, verify that required values are set (if applicable), generate filters for any filter data provided, send a request (with filters) to the entity data type and receive the response. If a verification step failed or no valid data could be retrieved, then return an error response message. Verify that the message information received for the entity data type conforms to the project specification. If it does not, then return an Exception. Otherwise create and return a sectionControlScheduleMsg message.
dlSectionControlStatusRequest	This method processes a sectionControlStatusRequestMsg message received as part of a dlSectionControlStatusRequest dialog request from the External Center. The processing includes the following steps: verify message authentication, verify the message is consistent with the project NRTM, verify that required values are set (if applicable), generate filters for any filter data provided, send a request (with filters) to the entity data type and receive the response. If a verification step failed or no valid data could be retrieved, then return an error response message. Verify that the message information received for the entity data type conforms to the project specification. If it does not, then return an error response message. Otherwise create and return the sectionControlResponseMsg message received from the entity data source.
dlSectionPriorityQueueRequest	This method processes a dlSectionPriorityQueueRequest dialog control request. The processing includes the following steps: verify message authentication, verify the message is consistent with the project NRTM, verify that required values are set (if applicable), create a command and send it to the command queue. Check the command queue to see if this request is available for processing. If yes, then process the command and send a sectionPriorityQueueMsg message indicating the status of the message processing. If no, then send a sectionPriorityQueueMsg message indicating that the message has been queued. Process the currently available command from the queue. Once completed, send a request to the command queue to update the status of the command. If a verification step failed or no valid data could be retrieved, then return an error response message.
dlSectionStatusRequest	This method processes a deviceInformationRequestMsg message received as part of a dlSectionStatusRequest dialog request from the External Center. The processing includes the following steps: verify message authentication, verify the message is consistent with the project NRTM, verify that required values are set (if applicable), generate filters for any filter data provided, send a request (with filters) to the entity data type and receive the response. If a verification step failed or no valid data could be retrieved, then return an error response message. Verify that the message information received for the entity

Dialog Name	Dialog Processing Description
	data type conforms to the project specification. If it does not, then return an error response message. Otherwise create and return the sectionStatusMsg message received from the entity data source.
dlSectionStatusUpdate	This method processes a dlSectionStatusUpdate dialog update request. The message expected is the original subscription message that was used to set up the publication. The processing includes the following steps: verify message authentication, verify the message is consistent with the project NRTM, verify that required values are set (if applicable), generate filters for any filter data provided, send a request (with filters) to the entity data type and receive the response. If a verification step failed or no valid data could be retrieved, then return an error response message. Verify that the message information received for the entity data type conforms to the project specification. If it does not, then return an Exception. Otherwise create and return a sectionStatusMsg message.
dlVideoSwitchControlRequest	This method processes a dlVideoSwitchControlRequest dialog control request. The processing includes the following steps: verify message authentication, verify the message is consistent with the project NRTM, verify that required values are set (if applicable), create a command and send it to the command queue. Check the command queue to see if this request is available for processing. If yes, then process the command and send a deviceControlResponseMsg message indicating the status of the message processing. If no, then send a deviceControlResponseMsg message indicating that the message has been queued. Process the currently available command from the queue. Once completed, send a request to the command queue to update the status of the command. If a verification step failed or no valid data could be retrieved, then return an error response message.
dlVideoSwitchInventoryRequest	This method processes a deviceInformationRequestMsg message received as part of a dlVideoSwitchInventoryRequest dialog request from the External Center. The processing includes the following steps: verify message authentication, verify the message is consistent with the project NRTM, verify that required values are set (if applicable), generate filters for any filter data provided, send a request (with filters) to the entity data type and receive the response. If a verification step failed or no valid data could be retrieved, then return an error response message. Verify that the message information received for the entity data type conforms to the project specification. If it does not, then return an error response message. Otherwise create and return the videoSwitchInventoryMsg message received from the entity data source.
dlVideoSwitchInventoryUpdate	This method processes a dlVideoSwitchInventoryUpdate dialog update request. The message expected is the original subscription message that was used to set up the publication. The processing includes the following steps: verify message authentication, verify the message is consistent with the project NRTM, verify that required values are set (if applicable), generate filters for any filter data provided, send a request (with filters) to the entity data type and receive the response. If a verification step failed or no valid data could be retrieved, then return an error response message. Verify that the message information received for the entity data type conforms to the project specification. If it does not, then return an Exception. Otherwise create and return a videoSwitchInventoryMsg message.
dlVideoSwitchStatusRequest	This method processes a deviceInformationRequestMsg message received as part of a dlVideoSwitchStatusRequest dialog request from the External Center.

Dialog Name	Dialog Processing Description
	The processing includes the following steps: verify message authentication, verify the message is consistent with the project NRTM, verify that required values are set (if applicable), generate filters for any filter data provided, send a request (with filters) to the entity data type and receive the response. If a verification step failed or no valid data could be retrieved, then return an error response message. Verify that the message information received for the entity data type conforms to the project specification. If it does not, then return an error response message. Otherwise create and return the videoSwitchStatusMsg message received from the entity data source.

6.2.3 Processing of Commands from an EC

Type: Package *Package:* Entity Emulation Overview

The following table provides details on how entity control commands are processed within the Entity Emulation capability when a valid Control Request message is received from an EC.

dialog	Message	Path	elementName	Status Data Updated	Inventory Data Referenced	Command Processing
dlLCSControlRequest	lCSControlRequestMsg	/tmdd303:lCSControlRequestMsg/lcs-request-command	lcs-request-command	/tmdd303:lCSStatusMsg/lcs-status-item[*]/lane-current-state		Change the status value based on the command provided.
dlRampMeterControlRequest	rampMeterControlRequestMsg	/tmdd303:rampMeterControlRequestMsg/metered-lane-list[*]/metered-lane/meter-command-parameters/meter-requested-plan	meter-requested-plan	/tmdd303:rampMeterStatusMsg/ramp-meter-status-item[*]/metered-status-list[*]/metered-lane/meter-implemented-plan /tmdd303:rampMeterStatusMsg/ramp-meter-status-item[*]/metered-status-list[*]/metered-lane/meter-requested-plan		Change the status values based on the parameter provided.
dlRampMeterControlRequest	rampMeterControlRequestMsg	/tmdd303:rampMeterControlRequestMsg/metered-lane-list[*]/metered-lane/meter-command-parameters/meter-requested-rate	meter-requested-rate	/tmdd303:rampMeterStatusMsg/ramp-meter-status-item[*]/metered-status-list[*]/metered-lane/meter-implemented-rate /tmdd303:rampMeterStatusMsg/ramp-meter-status-item[*]/metered-status-list[*]/metered-		Change the status values based on the parameter provided.

dialog	Message	Path	elementName	Status Data Updated	Inventory Data Referenced	Command Processing
				lane/meter-requested-rate		
dlRampMeterControlRequest	rampMeterControlRequestMsg	/tmdd303:rampMeterControlRequestMsg/metered-lane-list[*]/metered-lane/meter-request-command	meter-request-command	/tmdd303:rampMeterStatusMsg/ramp-meter-status-item[*]/metered-status-list[*]/metered-lane/meter-requested-action /tmdd303:rampMeterStatusMsg/ramp-meter-status-item[*]/metered-status-list[*]/metered-lane/meter-requested-plan		Change the status values based on the parameter provided.
dlIntersectionSignalControlRequest	intersectionSignalControlRequestMsg	/tmdd303:intersectionSignalControlRequestMsg/intersection-command-parameters/offset-adjustment	offset-adjustment	/tmdd303:intersectionSignalStatusMsg/intersection-signal-status-item[*]/offset-time-planned /tmdd303:intersectionSignalStatusMsg/intersection-signal-status-item[*]/offset-time-current		Change the status values based on the parameter provided. The command should be set to offset adjustment.
dlIntersectionSignalControlRequest	intersectionSignalControlRequestMsg	/tmdd303:intersectionSignalControlRequestMsg/intersection-command-parameters/request-timing-mode	request-timing-mode	/tmdd303:intersectionSignalStatusMsg/intersection-signal-status-item[*]/current-signal-timing-mode		Change the status value based on the command provided. The command should be set for timing mode.
dlIntersectionSignalControlRequest	intersectionSignalControlRequestMsg	/tmdd303:intersectionSignalControlRequestMsg/intersection-command-parameters/timing-pattern-id	timing-pattern-id	/tmdd303:intersectionSignalStatusMsg/intersection-signal-status-item[*]/timing-pattern-id-current		Change the status value as long as the command is set to change the pattern id. The pattern ID should probably exist in the pattern inventory.

dialog	Message	Path	elementName	Status Data Updated	Inventory Data Referenced	Command Processing
dlSectionControlRequest	sectionControlRequestMsg	/tmdd303:sectionControlRequestMsg/section-command-parameters/request-control-mode	request-control-mode	/tmdd303:sectionStatusMsg/section-status-item[*]/section-control-mode		Change the status value based on the value provided.
dlSectionControlRequest	sectionControlRequestMsg	/tmdd303:sectionControlRequestMsg/section-command-parameters/timing-pattern-id	timing-pattern-id	/tmdd303:sectionStatusMsg/section-status-item[*]/timing-pattern-id		Change the status value based on the value provided. The pattern id should probably exist in the pattern inventory. Ensure pattern id was specified for the command type.
dlCCTVControlRequest	cCTVControlRequestMsg	/tmdd303:cCTVControlRequestMsg/cctv-command-parameters/cctv-position-iris-lens	cctv-position-iris-lens	/tmdd303:cCTVStatusMsg/cctv-status-item[*]/cctv-position-iris-lens	/tmdd303:cCTVInventoryMsg/cctv-inventory-item[*]/cctv-camera-iris-limit /tmdd303:cCTVInventoryMsg/cctv-inventory-item[*]/cctv-requests-supported-list[*]/cctv-requests-supported	Change the status value as long as the requested value is below the limit expressed in the inventory data, and the command is valid per the inventory settings.
dlCCTVControlRequest	cCTVControlRequestMsg	/tmdd303:cCTVControlRequestMsg/cctv-command-parameters/cctv-position-pan	cctv-position-pan	/tmdd303:cCTVStatusMsg/cctv-status-item[*]/cctv-position-pan	/tmdd303:cCTVInventoryMsg/cctv-inventory-item[*]/cctv-camera-pan-left-limit /tmdd303:cCTVInventoryMsg/cctv-inventory-item[*]/cctv-camera-pan-right-limit /tmdd303:cCTVInventoryMsg/cctv-inventory-item[*]/cctv-requests-supported-list[*]/cctv-requests-supported	Change the status value as long as the requested value is between the limits expressed in the inventory data, and the command is valid per the inventory defined capabilities.

dialog	Message	Path	elementName	Status Data Updated	Inventory Data Referenced	Command Processing
dlCCTVControlRequest	cCTVControlRequestMsg	/tmdd303:cCTVControlRequestMsg/cctv-command-parameters/cctv-position-preset	cctv-position-preset	/tmdd303:cCTVStatusMsg/cctv-status-item[*]/cctv-position-preset	/tmdd303:cCTVInventoryMsg/cctv-inventory-item[*]/cctv-requests-supported-list[*]/cctv-requests-supported	Change the status value as long as the command is valid based on the inventory settings.
dlCCTVControlRequest	cCTVControlRequestMsg	/tmdd303:cCTVControlRequestMsg/cctv-command-parameters/cctv-position-tilt	cctv-position-tilt	/tmdd303:cCTVStatusMsg/cctv-status-item[*]/cctv-position-tilt	/tmdd303:cCTVInventoryMsg/cctv-inventory-item[*]/cctv-camera-tilt-up-limit /tmdd303:cCTVInventoryMsg/cctv-inventory-item[*]/cctv-camera-tilt-down-limit /tmdd303:cCTVInventoryMsg/cctv-inventory-item[*]/cctv-requests-supported-list[*]/cctv-requests-supported	Change the status value as long as the requested value is between the limits expressed in the inventory data, and the command is valid per the inventory defined capabilities.
dlCCTVControlRequest	cCTVControlRequestMsg	/tmdd303:cCTVControlRequestMsg/cctv-command-parameters/cctv-lock	cctv-lock		/tmdd303:cCTVInventoryMsg/cctv-inventory-item[*]/cctv-requests-supported-list[*]/cctv-requests-supported	The command is valid if specified in the inventory setting. Save the lock settings separately since it is not part of the defined inventory or status data, but is expected in control responses.
dlCCTVControlRequest	cCTVControlRequestMsg	/tmdd303:cCTVControlRequestMsg/cctv-command-parameters/cctv-text	cctv-text		/tmdd303:cCTVInventoryMsg/cctv-inventory-item[*]/cctv-requests-supported-list[*]/cctv-requests-supported	Valid as long as the capability is specified in the inventory data. It is not stored since it is not provided in any known messages.
dlCCTVControlRequest	cCTVControlRequestMsg	/tmdd303:cCTVControlRequestMsg/cctv-	cctv-environment	/tmdd303:cCTVStatusMsg/cctv-status-	/tmdd303:cCTVInventoryMsg/cctv-inventory-	Change the status value as long as the value is

dialog	Message	Path	elementName	Status Data Updated	Inventory Data Referenced	Command Processing
		command-parameters/cctv-environment		item[*]/cctv-environmental-status	item[*]/cctv-camera-environmental /tmdd303:cCTVInventoryMsg/cctv-inventory-item[*]/cctv-requests-supported-list[*]/cctv-requests-supported	within the limit specified in the inventory and this capability is enabled.
dlCCTVControlRequest	cCTVControlRequestMsg	/tmdd303:cCTVControlRequestMsg/cctv-command-parameters/cctv-position-zoom-lens	cctv-position-zoom-lens	/tmdd303:cCTVStatusMsg/cctv-status-item[*]/cctv-position-zoom-lens	/tmdd303:cCTVInventoryMsg/cctv-inventory-item[*]/cctv-camera-zoom-limit /tmdd303:cCTVInventoryMsg/cctv-inventory-item[*]/cctv-requests-supported-list[*]/cctv-requests-supported	Change the status value as long as it is within the limit defined in the inventory and the capability is present in the inventory setting.
dlCCTVControlRequest	cCTVControlRequestMsg	/tmdd303:cCTVControlRequestMsg/cctv-request-command	cctv-request-command		/tmdd303:cCTVInventoryMsg/cctv-inventory-item[*]/cctv-requests-supported-list[*]/cctv-requests-supported	Confirm the command is valid per the inventory before processing the parameters.
dlCCTVControlRequest	cCTVControlRequestMsg	/tmdd303:cCTVControlRequestMsg/cctv-command-parameters/cctv-position-focus-lens	cctv-position-focus-lens	/tmdd303:cCTVStatusMsg/cctv-status-item[*]/cctv-position-focus-lens	/tmdd303:cCTVInventoryMsg/cctv-inventory-item[*]/cctv-camera-focus-limit /tmdd303:cCTVInventoryMsg/cctv-inventory-item[*]/cctv-requests-supported-list[*]/cctv-requests-supported	Change the status value if it is within the limit specified in the inventory message. The capability must be included in the inventory.
dlVideoSwitchControlRequest	videoSwitchControlRequestMsg	/tmdd303:videoSwitchControlRequestMsg/channel-titling-text	channel-titling-text	/tmdd303:videoSwitchStatusMsg/video-switch-status-item[*]/switched-channel-list[*]/switched-channel/channel-titling-		Change the status value.

dialog	Message	Path	elementName	Status Data Updated	Inventory Data Referenced	Command Processing
				text		
dlVideoSwitchControlRequest	videoSwitchControlRequestMsg	/tmdd303:videoSwitchControlRequestMsg/output-channel-id	output-channel-id	/tmdd303:videoSwitchStatusMsg/video-switch-status-item[*]/switched-channel-list[*]/switched-channel/output-channel-id		Change the status value.
dlVideoSwitchControlRequest	videoSwitchControlRequestMsg	/tmdd303:videoSwitchControlRequestMsg/input-channel-id	input-channel-id	/tmdd303:videoSwitchStatusMsg/video-switch-status-item[*]/switched-channel-list[*]/switched-channel/input-channel-id		Change the status value.
dlVideoSwitchControlRequest	videoSwitchControlRequestMsg	/tmdd303:videoSwitchControlRequestMsg/set-output-channel-lock	set-output-channel-lock			Save the lock settings separately since it is not part of the defined inventory or status data, but is expected in control responses.
dlDMSControlRequest	dMSControlRequestMsg	/tmdd303:dMSControlRequestMsg/dms-beacon-control	dms-beacon-control	/tmdd303:dMSStatusMsg/dms-status-item[*]/message-beacon	/tmdd303:dMSInventoryMsg/dms-inventory-item[*]/dms-beacon-type	Change the status value as long as the inventory value is not set to none.
dlDMSControlRequest	dMSControlRequestMsg	/tmdd303:dMSControlRequestMsg/dms-command-parameters/message-number	message-number	/tmdd303:dMSStatusMsg/dms-status-item[*]/message-number		Change the status value as long as the correct dms-command-value is set to (1) put up a custom message or (2) put up a library messsage. The library message should be in the message inventory.
dlDMSControlRequest	dMSControlRequestMsg	/tmdd303:dMSControlRequestMsg/dms-command-	dms-message	/tmdd303:dMSStatusMsg/dms-status-item[*]/current-message		Change the status value as long as the correct dms-command-value is

dialog	Message	Path	elementName	Status Data Updated	Inventory Data Referenced	Command Processing
		parameters/dms-message				set to (1) put up a custom message.
dlGateControlRequest	gateControlRequestMsg	/tmdd303:gateControlRequestMsg/gate-request-command	gate-request-command	/tmdd303:gateStatusMsg/gate-status-item[*]/gate-status		Update the status based on the command value.
dlHARControlRequest	hARControlRequestMsg	/tmdd303:hARControlRequestMsg/har-command-parameters/har-message-number	har-message-number	/tmdd303:hARStatusMsg/har-status-item[*]/message-number		Change the status value if the har-request-command was set to (1) put up custom message or (2) put up library message. The message number should probably be present in the message inventory.
dlHARControlRequest	hARControlRequestMsg	/tmdd303:hARControlRequestMsg/har-command-parameters/har-message	har-message	/tmdd303:hARStatusMsg/har-status-item[*]/har-current-message		Change the status value if the har-request-command was set to (1) put up custom message.
dlSectionControlRequest	sectionControlRequestMsg	/tmdd303:sectionControlRequestMsg/section-command-parameters/section-offset-adjustment	section-offset-adjustment			Ensure that offet adjustment was selected in the command.

6.2.4 SubscriptionPublication

Type: Package *Package:* Entity Emulation Overview

Because the handling of subscriptions and publication dialogs is defined at the application layer (NTCIP 2306, NTCIP 2304), the C2C RI allocates the majority of subscription and publication processing to the classes/objects which are defined in the application layer standard. The Entity Emulation capability will support the publication of message data by 1) providing the getSubResponse method of the RIEmulation class in order to give accurate subscription request feedback 2) providing the getPubResponse method of the RIEmulation class in order to provide entity data for publication and 3) providing an entity data update notification function that will provide notifications to publishers when entity data related to there dialog has changed.

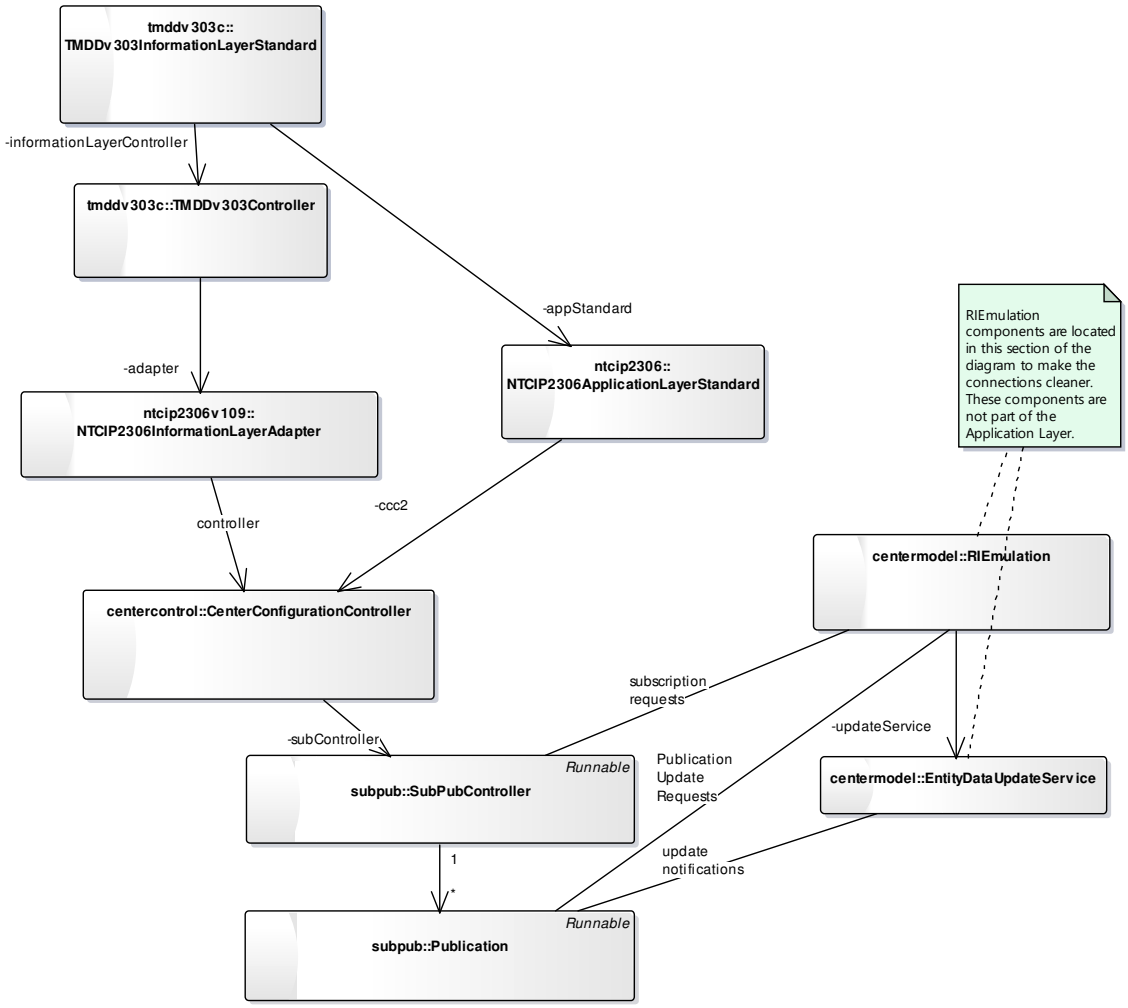

Figure 6-3: Objects Involved in Subscription Publication

This diagram presents a structure diagram of the primary set of run-time objects that are involved in the management of subscription and publication messages when the RI is operating as an OC. In this diagram, the RI is using NTCIP2306 as the application layer standard and TMDDv303c or newer as the information layer standard. Additional information about each object can be found in the documentation on each related class.

6.3 Application Layer Control

6.3.1 Application Layer Control Sequence Diagram - *(Interaction diagram)*

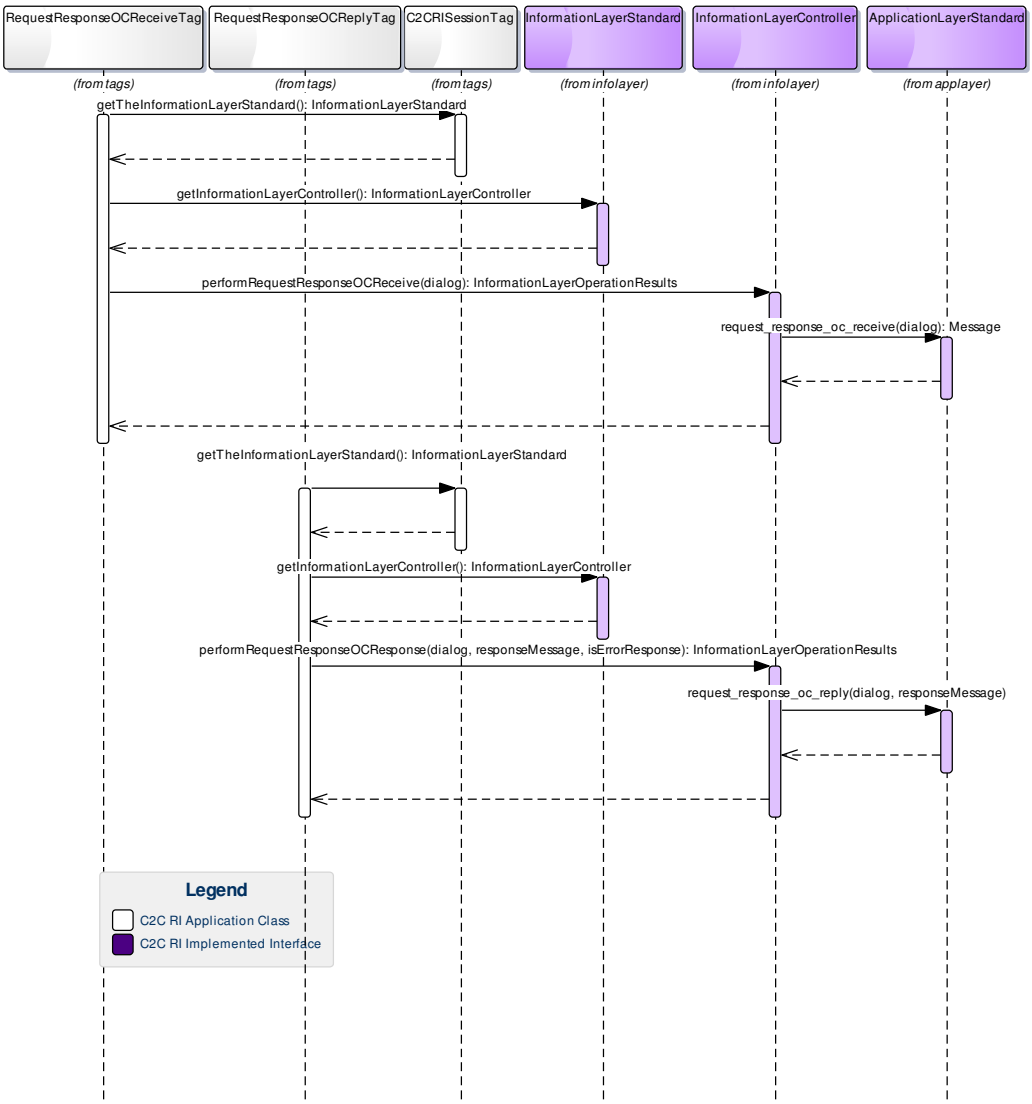

Figure 6-4: Application Layer Control Sequence Diagram

This sequence diagram illustrates how the execution of a script tag from the Information Layer Standard level results in an action being sent to the selection ApplicationLayerStandard object.

In this particular example, the owner center receives a message from the external center (RequestResponseOCReceiveTag) then subsequently sends a response (RequestResponseOCReplyTag). Note that any script tags, particularly those that prompt for user action, will affect the response time from the Owner Center (the RI) to the External Center (the SUT).

6.3.2 Application Layer Control with Entity Emulation Sequence Diagram - *(Interaction diagram)*

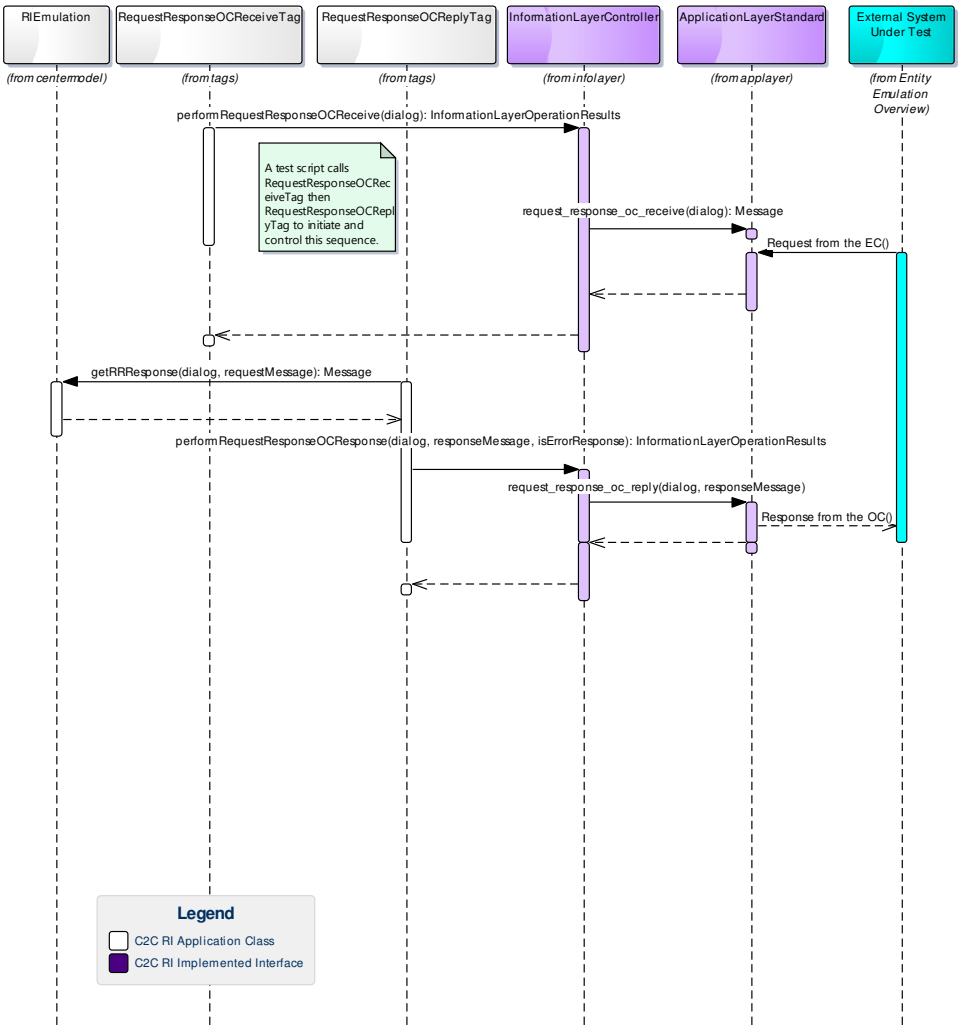

Figure 6-5: Application Layer Control with Entity Emulation Sequence Diagram

This sequence diagram illustrates how an RIEmulation object will be used by the OC Tags to request entity emulation information.

In this particular example, the owner center receives a message from the external center (RequestResponseOCReceiveTag).

The RequestResponseOCReplyTag sends the message received from the owner center to RIEmulation. RIEmulation processes the message, updates the model (when appropriate), and then returns a Message for response.
This response message is then sent to the EC through the ApplicationLayerStandard object.

Note that any script tags, particularly those that prompt for user action, will affect the response time from the Owner Center (the RI) to the External Center (the SUT).

6.3.3 Information Layer and Application Layer Standard Creation Sequence Diagram - *(Interaction diagram)*

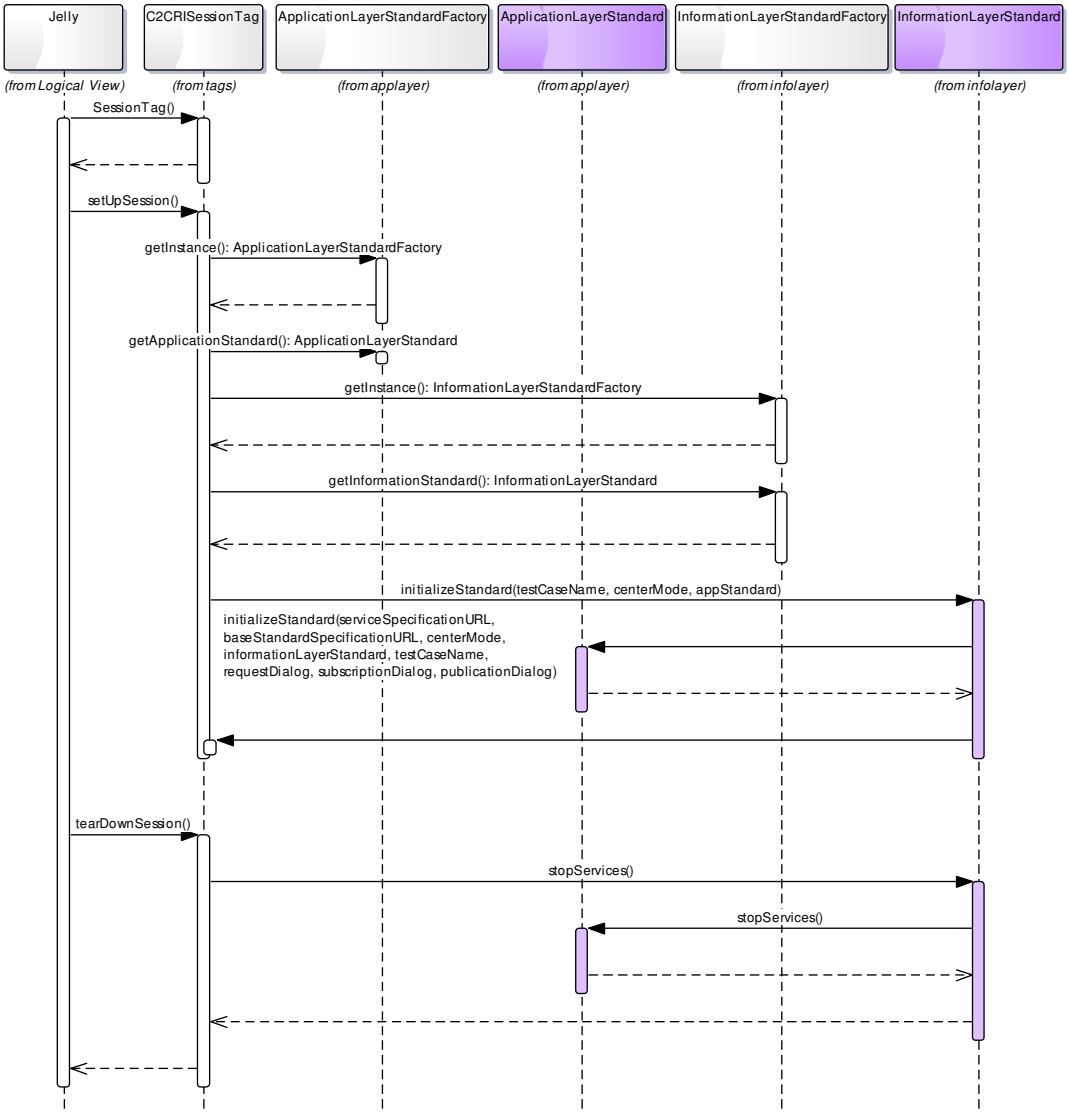

Figure 6-6: Information Layer and Application Layer Standard Creation Sequence Diagram

This sequence diagram illustrates how application layer standard and information layer standards are created, initialized and stopped based on the execution of a test script.

A C2CRISessionTag object is created based on the parameters included within each test script. The parameters include a specification of the Application and Information Layer standards that will be utilized during the test. Factories are used to create objects representing the two types of standard. The application layer standard is created first. This standard is passed to the information layer standard factory (not shown) and is associated with the newly created InformationLayerStandard object. The InformationLayerStandard is then initialized, and it, in turn, initializes the ApplicationLayerStandard object. For NTCIP2306, the initialization includes reading the specified WSDL file and creating the appropriate HTTP and FTP Servers and Clients to support the services and operations specified.

At the conclusion of the test script, Jelly calls the C2CRISessionTag's tearDownSession method. The C2CRISessionTag then calls the stopServices method of the InformationLayerStandard object which calls the stopServices method of the ApplicationLayerStandard.

6.3.4 Test Execution Sequence Diagram - *(Interaction diagram)*

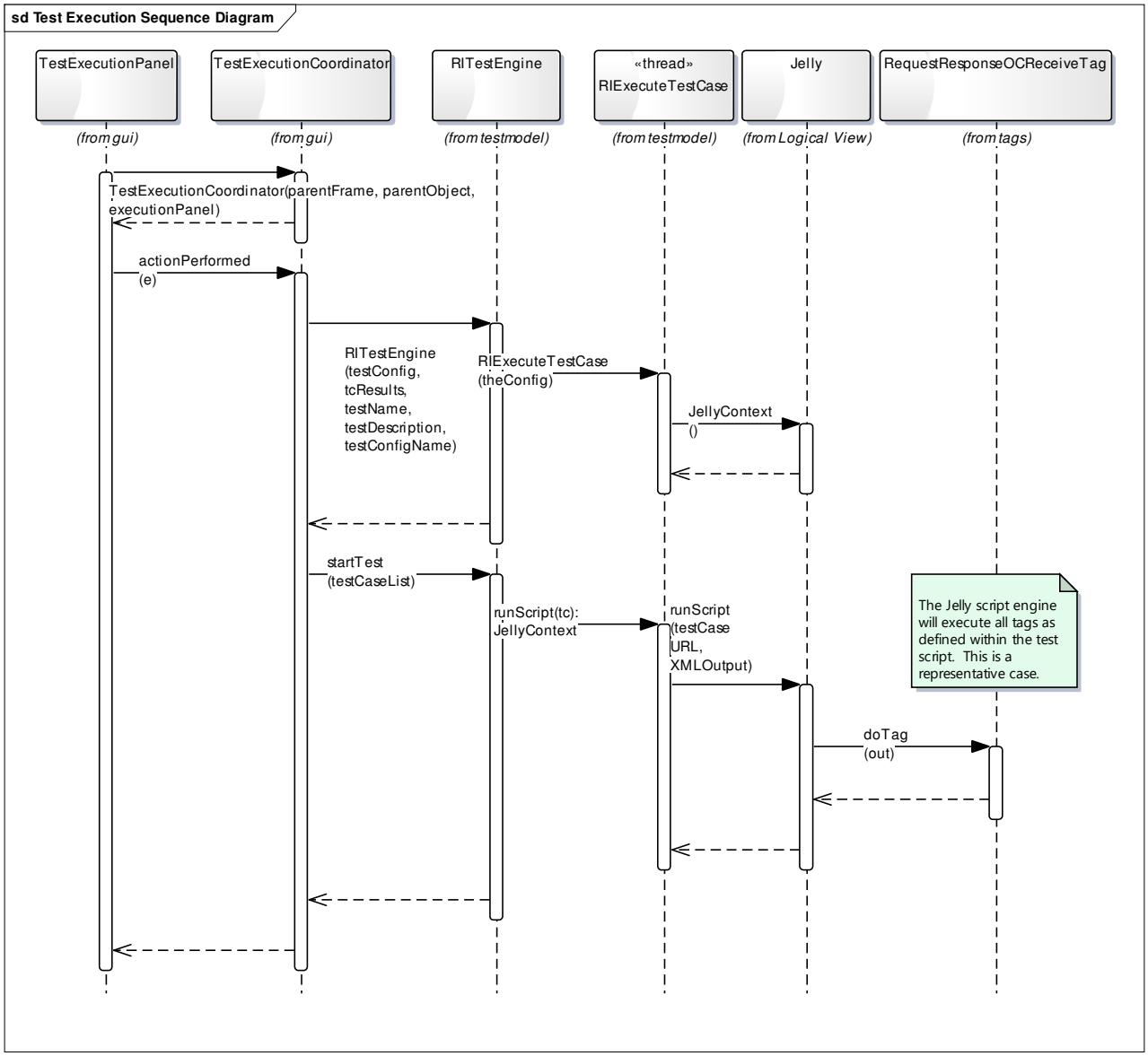

Figure 6-7: Test Execution Sequence Diagram

This sequence diagram illustrates the flow of control between the objects that are involved in executing a test in the C2C RI. This example only shows the execution of a single tag (RequestResponseOCReceiveTag), but in Jelly will execute all tags that are specified in the Test Script provided to it.

6.3.5 Other Figures

C2C RI Version: Version 1.0 → → 01/28/2015 11:10 AM
Created by: Tester

Section 1201 Conformance/Compliance Test Report

SUT in Owner Center Mode

Test Suites:
Application: NTCIP 2306 v1.69
Information: TMDD v3.03

Configuration File
Name: C:\C2CRI\RITest.ricfg
Checksum: 6988268F50ADCB323E1AF8DAB9BABDBD

Log File Name:	C:\C2CRI\TestLog.2015-01-28_11-03-29.xml
Log File Description:	Sample Test
Log File Creation Date:	2015-01-28_11-03-29

Information Layer Standard Conformance Report

UN-ID	**User Need**			**Results**
2.3.1.1	Verify Connection Active			Passed
		Requirement ID	**Related DXSF Requirement**	**Results**
		3.3.1.1.1	3.4.1.1.1	Passed
		3.3.1.2	3.4.1.1.2	Not Tested
2.3.1.2	**Need to Support Requesdts**			Not Tested
		Requirement ID	**Related DXSF Requirements**	**Results**
		3.3.1.2	3.4.2	Failed
		Time	**Test Case ID**	**Error Description**
		01:28:15 11:05:50.658	TCS-1-dlCenterActiveVerificationRequest-OC-InValid-1	Errors NOT expected but found! No Response Received

Application Layer Standard Conformance Report

UN-ID	**User Need**			**Results**
1	SOAP over HTTP			Passed
		Requirement ID	**Related DXSF Requirement**	**Results**
		1.a	Not Applicable	Passed
		1.c	Not Applicable	Not Tested

Source: C:\C2CRI\TestLog.2015-01-28_11-03-29.xml → Page 2 of 9

Figure 6-8: Draft 1201 Conformance Report

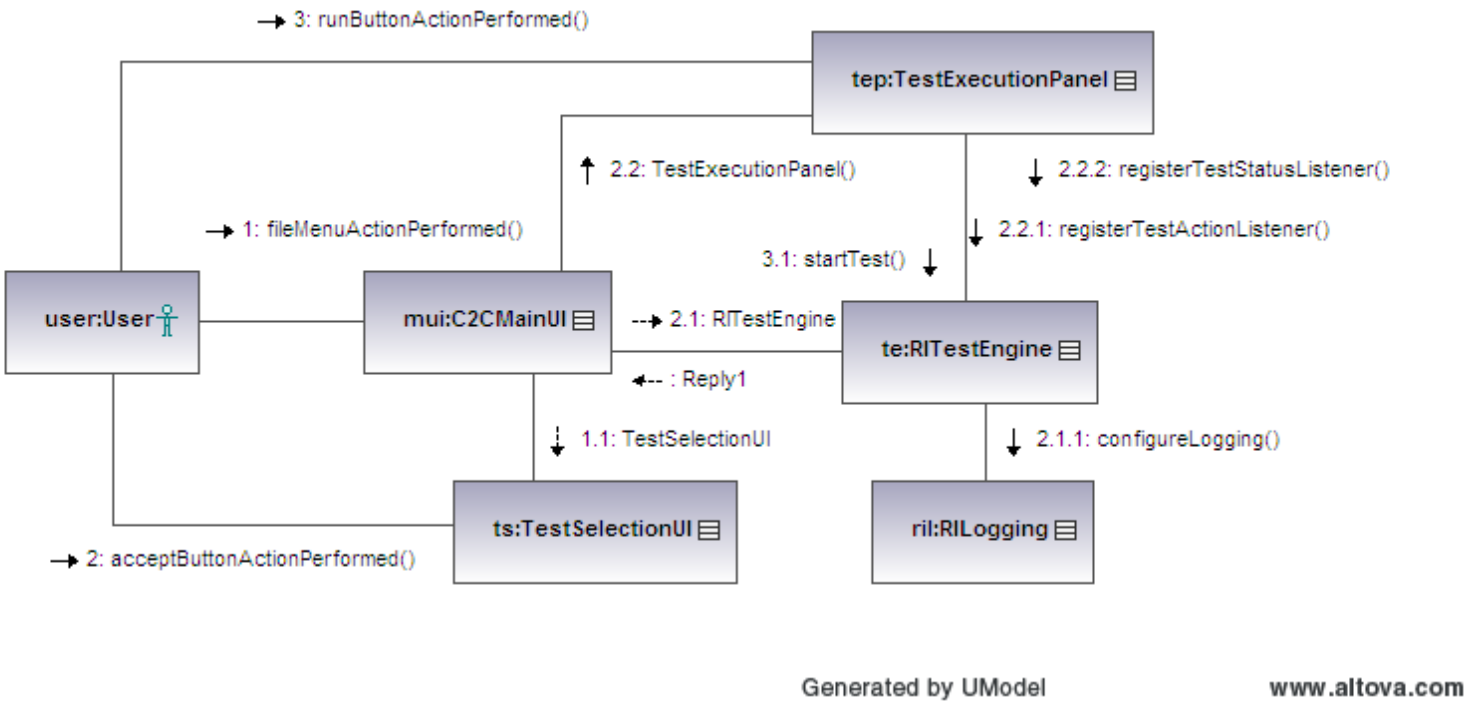

Figure 6-9: Run a Test Communication Diagram

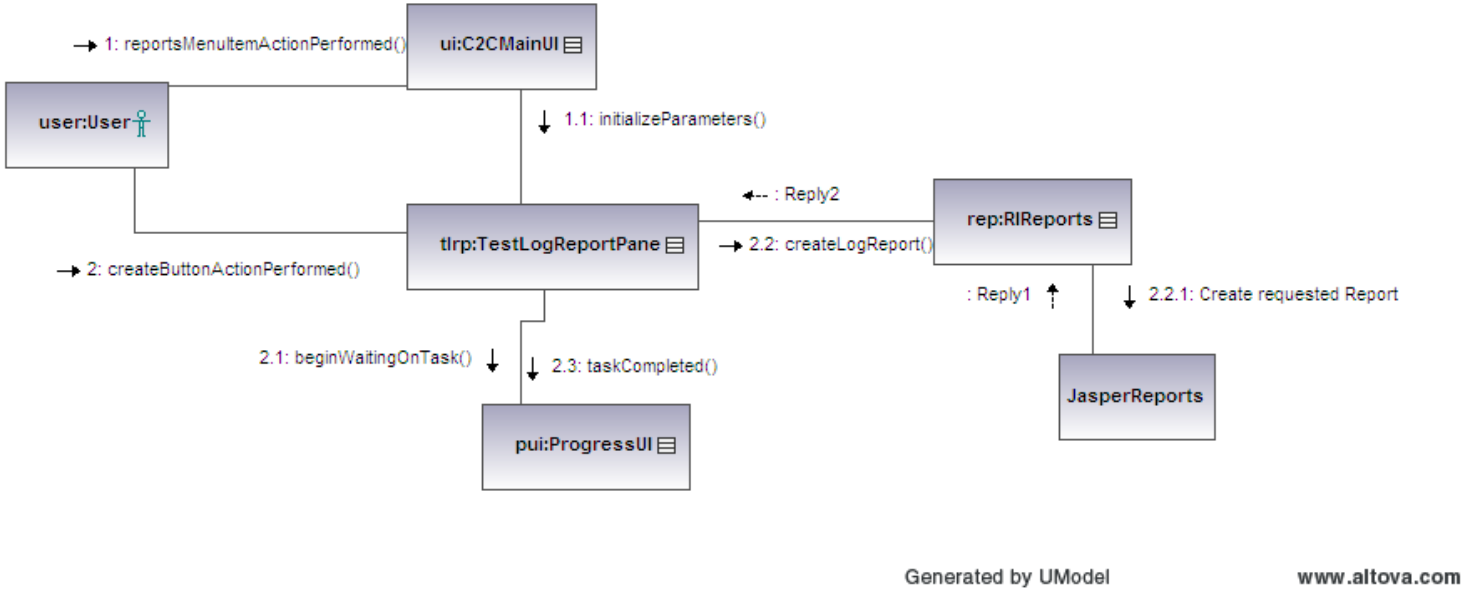

Figure 6-10: Create Log Report Communication Diagram

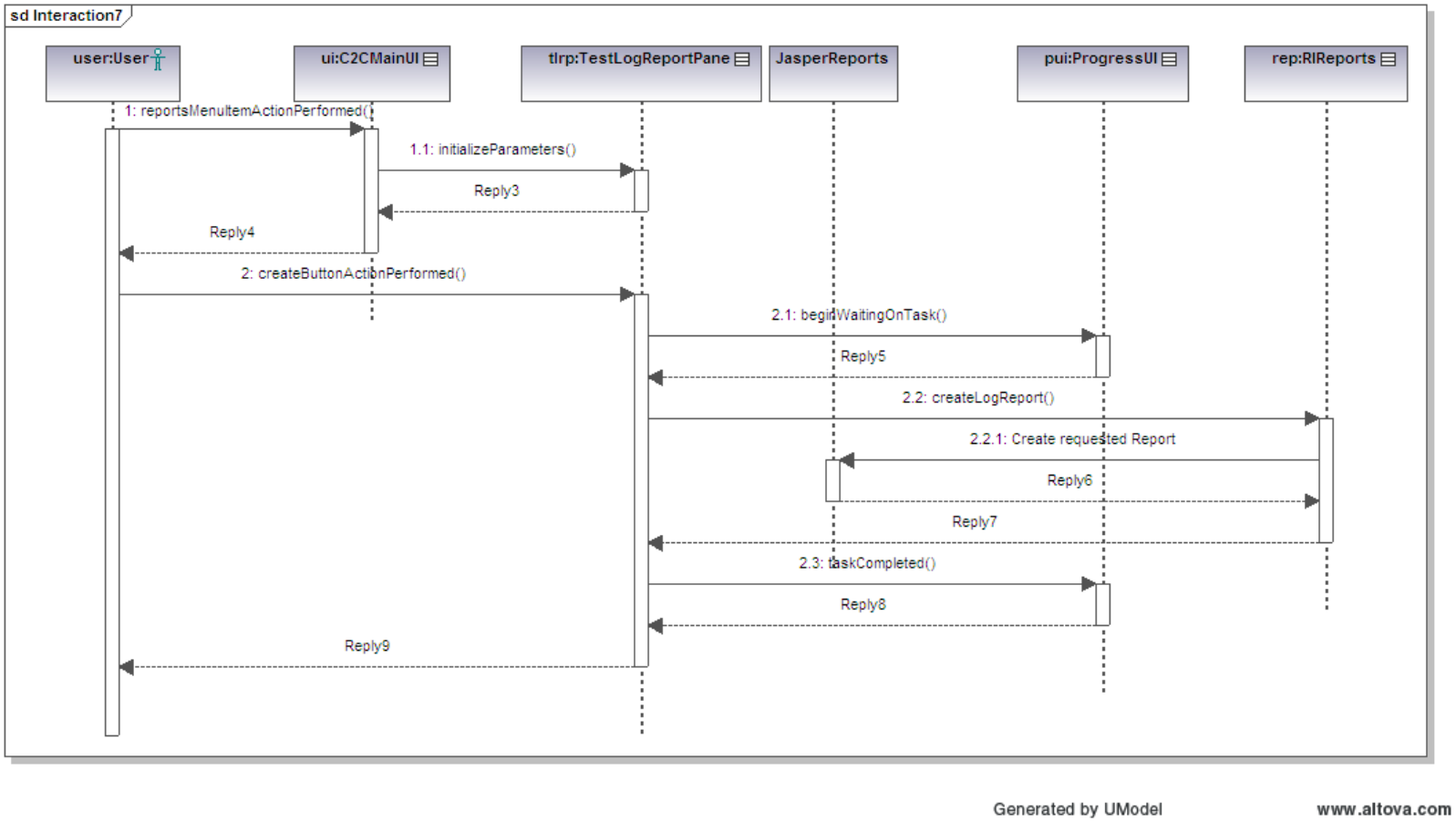

Figure 6-11: Create Log Report Sequence Diagram

7 APPENDIX A - REQUIREMENTS TRACEABILITY MATRIX

Requirement	SRS Section	Title	Design
SRS1: The RI System shall provide a user interface which incorporates the following attributes: SRS1.1: Pull-down menus on a top menu bar to navigate through the application SRS1.2: Dialog boxes, text boxes and buttons for obtaining user input SRS1.3: Keyboard and mouse control SRS1.4: Optional guided mode for the following steps SRS1.4.1: Develop a configuration file SRS1.4.2:Execute a test SRS1.4.3: Run reports	3.1.1	User Interfaces	C2CMainUI(5.1.1.2.1) ConfigurationBasePanel(5.1.2.2.1) ConfigurationPanel(5.1.2.2.6) HelpUI(5.1.1.2.5) InfoLayerPanel(5.1.2.2.4) InfoLayerTestCasePanel(5.1.2.2.5) MaintenanceUI(5.1.5.2.1) NavigationPanel(5.1.4.2.3) OptionsUI(5.1.1.2.6) PDFViewUI(5.1.3.2.5) SUTPanel(5.1.2.2.9) TestConfigCreateUI(5.1.2.2.11) TestExecutionPanel(5.1.4.2.2) C2CRIWizardPanel(5.1.2.2.13.2.1)
SRS2: The user interface shall be designed for operation on a display with a minimum 1024x768 pixel screen resolution.	3.1.1	User Interfaces	AppLayerPanel(5.1.2.2.2) AppLayerTestCasePanel(5.1.2.2.3) C2CMainUI(5.1.1.2.1) ConfigurationPanel(5.1.2.2.6) InfoLayerPanel(5.1.2.2.4) InfoLayerTestCasePanel(5.1.2.2.5) SUTPanel(5.1.2.2.9)
SRS3: The user interface shall provide the RI user with access to the functions to operate the RI as specified in Sections 3.2.1.1, 3.2.2.1 , 3.2.3.1 and 3.2.3.2. SRS3.1: “The user interface shall not disrupt or disable activated features of any operating system that are identified as accessibility features where the application programming interface for those accessibility features has been documented by the manufacturer of the operating system and is available to the product developer.” SRS3.2: “The user interface shall provide on-screen indication of the current focus that moves among interactive interface elements as the input focus changes.” SRS3.4: “The user interface shall allow product functions to be	3.1.1	User Interfaces	C2CMainUI(5.1.1.2.1) ConfigurationBasePanel(5.1.2.2.1) RIUICoordinator(5.1.1.2.2) TestConfigCreateUI(5.1.2.2.11) TestExecutionPanel(5.1.4.2.2) TestReportsPanel(5.1.3.2.3) TestSelectionUI(5.1.4.2.1)

Requirement	SRS Section	Title	Design
executable from the keyboard where the function itself and the result of performing a function can be discerned textually."			
SRS4: The RI shall be hosted and operate on a personal computer (PC) with: . SRS4.1: an Intel microprocessor architecture (minimum 1 gigahertz (GHz)) SRS4.2: 1280x1024 pixel resolution, 256 color monitor SRS4.3: Microsoft compatible keyboard and mouse SRS4.4: Minimum 512 MB RAM SRS4.5: Minimum 512 MB of available hard drive space on installation drive SRS4.6: Minimum 10 Mbps Network Interface Card	3.1.2	Hardware Interfaces	Java(5.11) The C2C RI runs within the Java environment, which is capable of running on a machine with the given specifications.
SRS5: The RI communications interface to the C2C network shall be Ethernet.	3.1.2	Hardware Interfaces	Java(5.11) The C2C RI Software runs within the Java environment. Java provides an interface to the servers ethernet interface through the host operating system.
SRS6: The RI Ethernet communications interface to the C2C network shall provide a minimum data rate of 10 Mbps.	3.1.2	Hardware Interfaces	Java(5.11) The C2C RI Software runs within the Java environment. Java provides an interface to the servers ethernet interface through the host operating system. Through this interface the C2C RI software can communicate with a SUT via a minimum data rate of 10 Mbps. However, this data rate may result in longer test time due to the slower transmission rate.

Requirement	SRS Section	Title	Design
SRS7: The RI shall be developed to run on: SRS7.1: Windows 10 SRS7.2: Linux – (Version to be specified in the future)	3.1.3.1	RI - OS Interface	Java(5.11)
SRS8: The RI shall provide a communications interface with the SUT that is conformant with the following standards: SRS8.1: Information Layer Standards SRS8.1.1: TMDD and MS/ETMCC (Version 3.0) – See Section 1.6 Conformance Statement SRS8.2: Application Layer Standards SRS8.2.2: NTCIP 2306 (both XML Direct and Web Services) – See Section 1.4 Conformance Clause	3.1.4.1	RI- External SUT Interface	C2CRIFunctionTag(5.3.1.2.29) C2CRISessionTag(5.3.1.2.30) FTPClient(5.9) FTPServer(5.1) HTTPClient(5.8.3.5.2.1) Jetty(5.11.1.1) Jetty(5.8.3.5.2.2) NTCIP2306ApplicationLayerOperationResults(5.8.1.2.3) NTCIP2306Controller(5.8.1.2.4) PublicationECReceiveTag(5.3.1.2.31) PublicationECReplyTag(5.3.1.2.32) PublicationOCTag(5.3.1.2.33) RequestResponseECTag(5.3.1.2.42) RequestResponseOCReceiveTag(5.3.1.2.43) RequestResponseOCReplyTag(5.3.1.2.44) RIMessageVerifyTag(5.3.1.2.25) RIMessageVerifyTag(5.3.1.2.25) SubscriptionECTag(5.3.1.2.45) SubscriptionOCReceiveTag(5.3.1.2.46) SubscriptionOCReplyTag(5.3.1.2.47) TMDDAuthenticationProcessor(5.4.2.2.2.1) TMDDErrorResponseProcessor(5.5.5.2.1) TMDDv303Controller(5.8.1.2.9) TMDDv303InformationLayerStandard(5.8.1.2.10) TMDDv303SubPubMapping(5.8.1.2.12) UserVerifyMessageTag(5.3.1.2.48) UserVerifyMessageTag(5.3.1.2.48)

Requirement	SRS Section	Title	Design
SRS9: The RI shall communicate with the SUT using one of the following: SRS9.1: TCP/IP SRS9.2: UDP/IP	3.1.4.1	RI- External SUT Interface	The C2C RI Software runs within the Java environment. Java provides an interface to the servers ethernet interface through the host operating system. Through this interface the C2C RI software can communicate with a SUT using the TCP/IP protocol.
SRS10: When the Test Developer requests that a new Test Configuration be created, the RI shall display a Test Configuration prompt to the Test Developer: SRS10.1: that shows a Test Configuration name field for the new Test Configuration SRS10.2: that shows a description field for the new Test Configuration SRS10.4: that shows an option to choose when creating the Test Configuration using name entered (see SRS10.1) SRS10.5: that shows an option to choose when cancelling the Test Configuration creation SRS10.7: that shows an option to select the Information Layer C2C standard Test Suite name to be used during the test from a list of pre-defined and user-defined Test Suites currently stored in the RI machine SRS10.8: that shows an option to select the Application Layer C2C standard Test Suite name to be used for the test from a list of pre-defined and user-defined Test Suites currently stored in the RI machine	3.2.1.1.1	Create, Select or Delete a Test Configuration	TestConfigCreateAction(5.1.2.2.10) TestConfigCreateUI(5.1.2.2.11)
SRS11: When the Test Developer enters a valid Test Configuration name and description and requests that the Test Configuration be created, the RI shall perform the following actions: SRS11.1: confirm that the Test Configuration Name is unique SRS11.2: create a new Test Configuration with the user specified name SRS11.3: display a Test Configuration Editing Window	3.2.1.1.1	Create, Select or Delete a Test Configuration	ConfigurationBasePanel(5.1.2.2.1) ConfigurationPanel(5.1.2.2.6) InfoLayerPanel(5.1.2.2.4) InfoLayerTestCasePanel(5.1.2.2.5) SUTPanel(5.1.2.2.9) TestConfigCreateUI(5.1.2.2.11)

Requirement	SRS Section	Title	Design
SRS12: When the Test Developer selects a Test Configuration name from the list and requests that the selection be accepted, the RI shall perform the following actions: SRS12.1: confirm that the Test Configuration File is valid; need to determine that the test configuration file is properly formed, contains meaningful content SRS12.2: display a Test Configuration Editing Window	3.2.1.1.1	Create, Select or Delete a Test Configuration	ConfigurationBasePanel(5.1.2.2.1) TestDefinitionCoordinator(5.3.6.2.3) Entity Emulation Overview(6.2)
SRS13: If the Test Developer enters an invalid Test Configuration name and requests that the name be accepted, the RI shall perform the following actions: SRS13.1: Display a message to the user that the Test Configuration name is invalid and describe the error with the name entered	3.2.1.1.1	Create, Select or Delete a Test Configuration	FilenameValidator(5.3.6.2.4) TestConfigCreateUI(5.1.2.2.11) TestDefinitionCoordinator(5.3.6.2.3) Entity Emulation Overview(6.2)
SRS14: If the Test Developer requests the cancellation of the creation of the Test Configuration, the RI shall remove the Test Configuration prompt.	3.2.1.1.1	Create, Select or Delete a Test Configuration	TestConfigCreateUI(5.1.2.2.11) TestDefinitionCoordinator(5.3.6.2.3)
SRS15: When the Test Developer selects a Test Configuration name from the list and requests that the selection be deleted, the RI shall perform the following actions: SRS15.1: confirm that the Test Configuration Name is valid SRS15.2: provide a confirmation box to the Test Developer to confirm that the Test Configuration should be removed SRS15.3: remove the Test Configuration from RI storage after receiving confirmation from the Test Developer	3.2.1.1.1	Create, Select or Delete a Test Configuration	MaintenanceUI(5.1.5.2.1)

Requirement	SRS Section	Title	Design
SRS16: When the Test Configuration Editing Window is being displayed, the RI shall provide the Test Developer with the following options: SRS16.2: an option to enter configuration settings for the SUT to include: SRS16.2.1: a field to enter The IP address of the SUT SRS16.2.2: a field to enter Port for communication to the SUT SRS16.2.3: a field to enter Host name of the SUT SRS16.2.4: a field to enter URL for the SUT Web Service SRS16.5: an option to select the Center Mode selection options SRS16.5.1: The default selection for the Center Mode shall be External Center. SRS16.8: an option to choose to save the Test Configuration with any changes made by the Test Developer SRS16.9: an option to choose to close the Test Configuration without saving any changes made by the Test Developer	3.2.1.1.2	Edit and Store a Test Configuration	AppLayerTestCasePanel(5.1.2.2.3) ConfigurationBasePanel(5.1.2.2.1) ConfigurationPanel(5.1.2.2.6) InfoLayerPanel(5.1.2.2.4) SUTPanel(5.1.2.2.9)
SRS17: The RI shall initialize all fields of the Test Configuration Editing Window with the information defined within the Test Configuration when displayed.	3.2.1.1.2	Edit and Store a Test Configuration	TestDefinitionCoordinator(5.3.6.2.3)
SRS18: For each standard selected by the Test Developer, the RI shall provide the Test Developer with an option to select requirements to be included within the test from a list of that standard's requirements. SRS18.1: For each Information Layer standard selected by the Test Developer, the RI shall provide an option for the Test Developer to select 1 or more requirements. SRS18.2: For each Application Layer standard selected by the Test Developer the RI shall provide an option for the Test Developer to select 1 or more requirements.	3.2.1.1.2	Edit and Store a Test Configuration	AppLayerPanel(5.1.2.2.2) InfoLayerPanel(5.1.2.2.4)

Requirement	SRS Section	Title	Design
SRS19: For each configuration setting entered by the Test Developer, the RI shall verify that the entry is valid. (see req. SRS16.2)	3.2.1.1.2	Edit and Store a Test Configuration	TestDefinitionCoordinator(5.3.6.2.3)
SRS20: When the Test Developer requests that the Test Configuration be saved (see req. SRS15.8), the RI shall SRS20.1: Verify the Test Configuration parameters entered to ensure all entries are valid. The definition of a valid entry will vary by Test Configuration parameter. For example, at least one Center Mode option must be selected. In addition each parameter will be checked to ensure it has a valid format, range, and referential integrity. For Information Layer and Application Layer related parameters the RI will need to include verification of option grouping as applicable. SRS20.2: Notify the Test Developer about any invalid entries SRS20.3: Provide an option for the Test Developer to overwrite the existing configuration if the Test Configuration was previously stored by the RI. SRS20.4: Store the Test Configuration information with the Test Developer provided information (see req. SRS16.2 through SRS16.5) and the Test Developer user name. SRS20.5: Provide an option for the Test Developer to save the Test Configuration file using a new user provided Test Configuration name.	3.2.1.1.2	Edit and Store a Test Configuration	TestDefinitionCoordinator(5.3.6.2.3)

Requirement	SRS Section	Title	Design
SRS21: When the Test Developer requests that the Test Configuration be closed (see req. SRS15.9), the RI shall SRS21.1: Provide a confirmation box to the Test Developer to confirm that the Test Configuration should be closed with or without saving. SRS21.2: Close the Test Configuration without storing any changes after receiving confirmation from the Test Developer. SRS21.3: Abort the close activity and display the Test Configuration Editor Window if the Test Developer does not confirm that the Test Configuration should be closed.	3.2.1.1.2	Edit and Store a Test Configuration	TestDefinitionCoordinator(5.3.6.2.3)
	3.2.1.1.3	User Identification	TestConfiguration(5.3.6.2.1)
	3.2.1.1.3	User Identification	OptionsUI(5.1.1.2.6)
	3.2.1.1.3	User Identification	Deleted

Requirement	SRS Section	Title	Design
	3.2.1.1.3	User Identification	Deleted
	3.2.1.1.3	User Identification	OptionsUI(5.1.1.2.6) ParameterFile(5.4.1.3.2.2) ParameterFile(5.4.1.3.2.2)
	3.2.1.1.3	User Identification	Deleted
SRS28: When the Test Developer requests that a Test Log be deleted, the RI shall display a Test Log Deletion prompt to the Test Developer: SRS28.1: that shows an option to select previously stored test logs from a displayed list of Test Logs SRS28.2: that shows an option to choose to cancel the Test Log deletion	3.2.1.1.4	Delete a Test Log	MaintenanceUI(5.1.5.2.1)
SRS29: When the Test Developer selects a Test Log from the list provided in the Test Log Deletion prompt and requests that the selection be deleted, the RI shall perform the following actions: SRS29.2: provide a confirmation box to the Test Developer to obtain confirmation that the Test Log should be removed	3.2.1.1.4	Delete a Test Log	MaintenanceUI(5.1.5.2.1)

Requirement	SRS Section	Title	Design
SRS30: The RI System shall utilize all pre-defined Test Suites satisfying the following conditions: SRS30.2: verify that the Test Suites conform to the defined Test Suite specification SRS30.4: pre-defined test suites are keyed so that the RI software can confirm the test suite was provided by an authorized organization	3.2.1.1.5	Test Suites	NRTM(5.3.4.2.1) TestCaseMatrix(5.3.2.2.1) TestCases(5.3.1.2.1) TestSuite(5.3.2.2.5.1) TestSuites(5.3.2.2.5)
SRS31.1: that shows an option to select a Test Configuration from a displayed list of previously stored Test Configurations SRS31.3: that shows a test name field. Note: The RI will add an add an extension to the filename which includes a timestamp. This extension will make each test name unique. SRS31.4: that shows a descriptive test notes field to enter information about the test being run SRS31.5: that shows an option to choose to cancel the request SRS31.6: that shows an option to choose to accept the request SRS31.7: that shows an option to choose to enable entity emulation during testing	3.2.2.1	User Control of a Test	TestSelectionUI(5.1.4.2.1)
SRS32: When the Test Operator requests that a selected Test Configuration be run the RI shall SRS32.1: create a Test Log for the Test with a name composed of the operator provided test name and an extension that includes the time and date that the file is created. SRS32.2: store the Test Configuration in the Test Log SRS32.3: store the name of the Test Operator in the Test Log SRS32.4: store the unique test name in the Test Log SRS32.5: store the current date and time in the Test Log SRS32.6: store any descriptive test notes in the Test Log SRS32.7: provide options to the user to choose to pause or terminate the test SRS32.9: execute all test scripts of the Test Suite associated with the	3.2.2.1	User Control of a Test	ConfigurationBasePanel(5.1.2.2.1) RIExecuteTestCase(5.2.1.1.2.1) RILogging(5.6.1.2.1) RITestEngine(5.2.1.1.2.2) TestExecutionCoordinator(5.2.1.1.2.3) TestExecutionPanel(5.1.4.2.2)

Requirement	SRS Section	Title	Design
user selected Information Layer and Application Layer test cases.			
SRS33: If the Test Operator requests a pause of the Test while the RI is executing a test script, the RI shall SRS33.1: stop executing the test script SRS33.2: provide options to the user to choose to resume or terminate the test SRS33.3: disable all other user actions other than resume and terminate	3.2.2.1	User Control of a Test	FunctionTag(5.3.1.2.4) RITestEngine(5.2.1.1.2.2) TestExecutionCoordinator(5.2.1.1.2.3) TestExecutionPanel(5.1.4.2.2) TestStepTag(5.3.1.2.5)
SRS34: If the Test Operator requests that the RI resume the Test, the RI shall SRS34.1: execute the test script that was running when paused starting with the next test action (step) that has not yet been performed during the test SRS34.2: provide options to the user to choose to pause or terminate the Test SRS34.3: disable user access to other RI functions other than pause and terminate	3.2.2.1	User Control of a Test	RITestEngine(5.2.1.1.2.2) TestExecutionCoordinator(5.2.1.1.2.3) TestExecutionPanel(5.1.4.2.2)
SRS35: If the Test Operator requests that the RI terminate the Test the RI shall SRS35.1: stop executing the current test script SRS35.2: provide a confirmation box to the Test Operator to confirm that the test is to be terminated SRS35.2.1: If the Test Operator confirms that the test is to be terminated the RI shall SRS35.2.1.1: cease the execution of all test scripts SRS35.2.1.2: disable the pause and resume user actions SRS35.2.1.3: enable all other user actions SRS35.3: continue the execution of the current and remaining test scripts if the Test Operator does not confirm	3.2.2.1	User Control of a Test	RITestEngine(5.2.1.1.2.2) TestExecutionCoordinator(5.2.1.1.2.3) TestExecutionPanel(5.1.4.2.2)

Requirement	SRS Section	Title	Design
that the test is to be terminated			
SRS36: When the RI has completed the execution of all test scripts (see req. SRS32.9), the RI shall SRS36.1: provide a notification to the Test Operator that the test is complete SRS36.2: disable the run, pause and resume user actions SRS36.3: enable user access to other RI user functions	3.2.2.1	User Control of a Test	C2CMainUI(5.1.1.2.1) RITestEngine(5.2.1.1.2.2) TestExecutionCoordinator(5.2.1.1.2.3) TestExecutionPanel(5.1.4.2.2)
SRS37: While the RI is running a Test, the RI shall store entries in the Test Log as follows: SRS37.2: The RI provided date and time stamp shall have a minimum precision of 10 ms. SRS37.4: Each message log entry stored shall include the following: SRS37.4.1: Message Source SRS37.4.2: Message Destination SRS37.4.3: Message Type SRS37.5: The RI shall store the name and result of each test script action that it performs in the Test Log. The result of the test script indicates whether the test script completed successfully. SRS37.6: The RI shall store the actual message data transmitted and received between the RI and the SUT in the Test Log. SRS37.7: Each data transmission log entry stored shall include the following: SRS37.7.1: Message Source SRS37.7.2: Message Destination SRS37.7.3: Message Protocol	3.2.2.2	Logging of Test Activities and Results	FunctionResult(5.3.1.2.6) FunctionTag(5.3.1.2.4) Java(5.11) log4j(5.6.3.2.1) Message(5.5.1.2.1) RILogging(5.6.1.2.1) RIXMLLayout(5.6.2.2.1)

Requirement	SRS Section	Title	Design
SRS38.4: Respond – send a response to the SUT acting as an External Center SRS38.5: Publish – publish a message to the SUT acting as an External Center SRS38.6: Validate – validate the format and content of the message received from the SUT acting as an External Center and store the results in the Test Log SRS38.7: Pause – pause the execution of the script and await a user action SRS38.8: Terminate – end the execution of the script	3.2.2.3	Perform Test Script Actions	FTPGetOCTag(5.3.1.2.10) HTTPGetOCTag(5.3.1.2.12) HTTPPostOCReplyTag(5.3.1.2.14) HTTPPostOCRequestTag(5.3.1.2.15) NTCIP2306FunctionTag(5.3.1.2.7) NTCIP2306SessionTag(5.3.1.2.8) RIPauseTag(5.3.1.2.26) RITerminateTag(5.3.1.2.28) RIValidateTag(5.3.1.2.16) SOAPPUBOCTag(5.3.1.2.18) SOAPRROCReplyTag(5.3.1.2.20) SOAPRROCRequestTag(5.3.1.2.21) SOAPSUBOCTag(5.3.1.2.23)
SRS39.4: Respond – send a response to the SUT acting as an Owner Center SRS39.5: Request – send a request to the SUT acting as an Owner Center. For testing conformance to a standards processing requirement, this action type also verifies and reports whether the processing requirement was fulfilled (i.e. a maximum wait period for a SUT response). SRS39.6: Subscribe – establish a subscription with the SUT acting as an Owner Center. For testing whether the Owner Center properly manages the subscription, this action type also verifies and reports whether the Owner Center satisfies specified publishing conditions (i.e. frequency of publication updates and total publication updates). SRS39.7: Validate – validate the format and content of the message received from the SUT acting as an Owner Center and store the results in the Test Log SRS39.8: Pause – pause the execution of the script and await a user action SRS39.9: Terminate – end the execution of the script SRS39.10: Receive – receive a message from the SUT	3.2.2.3	Perform Test Script Actions	FTPGetECTag(5.3.1.2.9) HTTPGetECTag(5.3.1.2.11) HTTPPostECTag(5.3.1.2.13) NTCIP2306FunctionTag(5.3.1.2.7) NTCIP2306SessionTag(5.3.1.2.8) RIPauseTag(5.3.1.2.26) RITerminateTag(5.3.1.2.28) RIValidateTag(5.3.1.2.16) SOAPPUBECTag(5.3.1.2.17) SOAPRRECTag(5.3.1.2.19) SOAPSUBECTag(5.3.1.2.22)

Requirement	SRS Section	Title	Design
SRS40: When the RI is running a Test, and Owner Center is the Center Mode specified in the Test Configuration, the RI shall perform Entity Emulation actions specified for the following entities and mandatory data classes specified in the currently adopted version of the TMDD: SRS40.1: CCTV Camera (see Section 3.2.2.4.1) SRS40.2: Dynamic Message Sign (See Section 3.2.2.4.3) SRS40.3: Environmental Sensor Station (See Section 3.2.2.4.4) SRS40.4: Gate (See Section 3.2.2.4.6) SRS40.5: Highway Advisory Radio (See Section 3.2.2.4.7) SRS40.6: Ramp Meter (See Section 3.2.2.4.9) SRS40.7: Traffic Detector (See Section 3.2.2.4.2) SRS40.8: Intersection Signal Controller (See Section 3.2.2.4.11) SRS40.9: Events (See Section 3.2.2.4.5) SRS40.10: Lane Control Signal (See Section 3.2.2.4.8) SRS40.11: Signal Section (See Section 3.2.2.4.10) SRS40.12: Video Switch (See Section 3.2.2.4.12) SRS40.13: Center Active Verification (see Section 3.2.2.4.13) SRS40.14: Link (see Section 3.2.2.4.14) SRS40.15: Node (see Section 3.2.2.4.15)	3.2.2.4	Entity Emulation Actions	NTCIP2306ApplicationLayerStandard(5.8.1.2.8) RIEmulation(5.4.1.1.2.5) Entity Emulation Overview(6.2) Entity Emulation functions include the following: establishing the initial set of data for each entity type, updating the entity data based on valid control requests, updating the entity data based on scripted commands to update, delete and modify entitities, providing responses for external center request messages and providing publication messages for one-time, periodic and event driven subscriptions. The RI emulation class is the primary class for this function. The various entity handlers are responsible for managing all entity related interactions. Further descriptions of the attributes and operation of these classes are found in their related class definition sections. Additional details are provided in the section of the document that provides addition overview information on Entity Emulation.
SRS41: When the RI is running a Test, the Owner Center is the Center Mode specified in the Test Configuration and Entity Emulation is selected, the RI shall perform the following Entity Emulation actions when specified in a test script action: SRS41.1: Initialize – initialize the entity SRS41.2: Update State – change the state of the entity SRS41.3: Update Attribute – change the attributes of the entity SRS41.4: Get State – obtain the current state of an entity	3.2.2.4	Entity Emulation Actions	ArchivedDataProcessingDocumentationMetaData(5.4.1.2.2.2.1) CCTVInventory(5.4.1.2.2.1.1) CCTVStatus(5.4.1.2.2.1.2) DetectorData(5.4.1.2.2.2.2) DetectorInventory(5.4.1.2.2.2.3) DetectorMaintenanceHistory(5.4.1.2.2.2.4) DetectorStatus(5.4.1.2.2.2.5) DMSFontTable(5.4.1.2.2.3.1) DMSInventory(5.4.1.2.2.3.2) DMSMessageAppearance(5.4.1.2.2.3.3) DMSMessageInventory(5.4.1.2.2.3.4)

Requirement	SRS Section	Title	Design
SRS41.5: Get Attribute – obtain the current value of an attribute of an entity SRS41.6: Add Entity – create an entity SRS41.7: Remove Entity – remove an entity			DMSStatus(5.4.1.2.2.3.5) ESSInventory(5.4.1.2.2.4.1) ESSObservationMetadata(5.4.1.2.2.4.2) ESSObservationReport(5.4.1.2.2.4.3) ESSStatus(5.4.1.2.2.4.4) EventsActionLog(5.4.1.2.2.5.2) EventsIndex(5.4.1.2.2.5.3) EventUpdates(5.4.1.2.2.5.1) GateControlSchedule(5.4.1.2.2.6.1) GateInventory(5.4.1.2.2.6.2) GateStatus(5.4.1.2.2.6.3) HARControlSchedule(5.4.1.2.2.7.1) HARInventory(5.4.1.2.2.7.2) HARMessageInventory(5.4.1.2.2.7.3) HARStatus(5.4.1.2.2.7.4) LaneControlSignalControlSchedule(5.4.1.2.2.8.1) LaneControlSignalInventory(5.4.1.2.2.8.2) LaneControlSignalStatus(5.4.1.2.2.8.3) NTCIP2306ApplicationLayerStandard(5.8.1.2.8) RampMeterControlSchedule(5.4.1.2.2.9.1) RampMeterInventory(5.4.1.2.2.9.2) RampMeterPlanInventory(5.4.1.2.2.9.3) RampMeterStatus(5.4.1.2.2.9.4) RIAddEntityElementTag(5.3.1.2.34) RIAddEntityTag(5.3.1.2.35) RIDeleteEntityElementTag(5.3.1.2.36) RIDeleteEntityTag(5.3.1.2.37) RIEmulation(5.4.1.1.2.5) RIGetEntityElementValueTag(5.3.1.2.38) RIInitializeEntityTag(5.3.1.2.39) RIUpdateEntityElementTag(5.3.1.2.41) SectionControlSchedule(5.4.1.2.2.10.1) SectionSignalTimingPatternInventory(5.4.1.2.2.10.2) SectionStatus(5.4.1.2.2.10.3) IntersectionSignalControllerControlSchedule(5.4.1.2.2.11.1) IntersectionSignalControllerInventory(5.4.1.2.2.11.2) IntersectionSignalControllerStatus(5.4.1.2.2.11.3)

Requirement	SRS Section	Title	Design
SRS42: When the RI is running a Test and the Test Script specifies a set of actions as a Test Script Pre-Condition, the RI shall perform the actions within the pre-condition.	3.2.2.5	Support Test Pre-Conditions	PreconditionTag(5.3.1.2.3)
SRS43: If every action of the Test Script Pre-Condition is not successfully completed, the RI will not perform any more actions within the Test Script. Multiple Test Scripts may be executed within a test, and they may be nested. This requirement applies only to the test script with the failed pre-condition.	3.2.2.5	Support Test Pre-Conditions	PreconditionTag(5.3.1.2.3)
SRS44: When the RI is running a Test and the subscript specifies a set of actions as a Test Case Pre-Condition, the RI shall perform the actions within the pre-condition.	3.2.2.5	Support Test Pre-Conditions	RIExecuteScriptTag(5.3.1.2.24) RISubscriptTag(5.3.1.2.27)
SRS45: If every action of the subscript Pre-Condition is not successfully completed, the RI will not perform any more actions within the subscript.	3.2.2.5	Support Test Pre-Conditions	FunctionTag(5.3.1.2.4) PreconditionTag(5.3.1.2.3) RISubscriptTag(5.3.1.2.27)
SRS46: When the RI is running a Test and the Test Script specifies a set of actions as a Test Script Post-Condition, the RI shall perform the actions within the post-condition. SRS46.1: when a previous test step is not successfully completed SRS46.2: when all test steps within the Test Script, prior to the post-condition, were successfully completed	3.2.2.6	Support Test Post-Conditions	FunctionTag(5.3.1.2.4) PostconditionTag(5.3.1.2.2)

Requirement	SRS Section	Title	Design
	3.2.2.6	Support Test Post-Conditions	PostconditionTag(5.3.1.2.2) RITestEngine(5.2.1.1.2.2) TestExecutionCoordinator(5.2.1.1.2.3)
SRS48: When the Test Analyst requests to create a Test Report, the RI shall display a Create Test Report prompt to the Test Analyst: SRS48.1: that shows an option to select a Test Configuration from a displayed list of previously stored Test Configurations SRS48.2: that shows an option to select a Test Log from a displayed list of previously stored Test Logs SRS48.3: that shows an option to select a report type from a list including (Conformance/Compliance Report, Test Case Summary, Test Case Details, Message Summary, Message Detail and Script Log) SRS48.5: that shows an option to choose to creating the selected Test Configuration Report SRS48.6: that shows an option to choose to create the selected Test Log Report SRS48.7: that shows a field to enter a unique report name SRS48.8: that shows an option to choose to cancel the request SRS48.9: that shows an option to choose to export the Test Log file to a CSV formatted file SRS48.10: that shows an option to choose to view the created Test Log Report SRS48.11: that shows an option to choose to view the created Test Configuration Report	3.2.3.1	Create Test Reports	ReportsCoordinator(5.1.3.2.1) RIUICoordinator(5.1.1.2.2) TestConfigurationList(5.3.6.2.2) TestConfigurationReportPane(5.1.3.2.2) TestLogReportPane(5.1.3.2.4) TestReportsPanel(5.1.3.2.3)

Requirement	SRS Section	Title	Design
SRS49: If the Test Analyst requests a Test Configuration Report be generated, the RI shall create a report containing: SRS49.1: the Test Developer selected settings of the Test Configuration, if the Test Analyst has selected Configuration File Details to be included in the report. SRS49.2: the Test Scripts which are included in the selected Test Suite, if the Test Analyst has selected that a Test Scripts Report to be generated. SRS49.3: the Entity Emulation Parameters which are included in the selected Configuration File.	3.2.3.1	Create Test Reports	JasperReports(5.7.1.2.2) ReportsCoordinator(5.1.3.2.1) RIReports(5.7.1.2.3)
SRS50: If the Test Analyst requests a Test Log Report be generated the RI shall: SRS50.1: If the Script Log report option is selected, create a report listing in chronological order the test script actions that were performed and the pass/fail results of each. SRS50.2: If the Message Summary report option is selected, create a report listing in chronological order the message names that were transmitted and received. SRS50.3: If the Message Detail report option is selected, create a report listing in chronological order the message names, the message fields and values that were transmitted and received. SRS50.4: If the Conformance/Compliance Report option is selected, create a report showing an Information Layer Standard Compliance Report section and an Application Layer Standard Compliance Report section showing compliance status result for each user need and its associated requirements. SRS50.5: If the Test Case Summary option is selected, create a report listing the test cases that were performed and the pass/fail result of each. SRS50.6: If the Test Case Details option is selected, create a report listing the test cases and their associated test steps that were	3.2.3.1	Create Test Reports	JasperReports(5.7.1.2.2) LogFileProcessor(5.6.1.2.2) LogFileProcessor(5.6.1.2.2) ReportsCoordinator(5.1.3.2.1) RIReports(5.7.1.2.3) ReportGeneration(6.1) (SRS 50.7) The C2C RI verifies a test systems conformance to TMDD requirements through the execution of validation techniques that are included within defined test procedures. It stores the verification status of each TMDD requirement that was selected in the project specification (NRTM). Section 3 of the RTSMIP DXFS specifies the requirements that are fulfilled by requirements defined within TMDD. The mapping between the requirements of these standards will be stored within the Reports Database. When the "Section 1201 report" is executed, it's underlying query will check each TMDD requirement for a matching RTSMIP DXSF requirement. If a matching RTSMIP DXSF requirement is found, then it will be shown in the "Related DXSF Requirement" column of the report. If no matches are found, then "Not Applicable" will be shown in the column. Figure XX provides an example of the report. (SRS50.8) A "Section 1201 report" will be added to the options on the

Requirement	SRS Section	Title	Design
performed and the pass/fail result of each test case and test step. SRS50.7: In support of Section 1201 of the Safe, Accountable, Flexible, Efficient Transportation Equity Act: A Legacy for Users (SAFETEA-LU) the C2C RI shall verify a test systems conformance with the requirements traced to the TMDD standard in the Real-Time System Management Information Program Data Exchange Format Specification provided on the USDOT/FHWA website http://www.ops.fhwa.dot.gov/publications/fhwahop13047/seces.htm. SRS50.8: The C2C RI shall provide a report which indicates the pass/fail test status for each requirement defined in Section 3.4 of the Real-Time System Management Information Program Data Exchange Format Specification that relates to a project requirement from the TMDD Information Layer Standard. SRS50.9: Each log report shall include a field indicating whether Entity Emulation was enabled for the test. SRS50.10: Each log report shall include a field indicating whether entity emulation before test case was enabled for the test.			LogReports GUI. When the user selects this report and clicks the run button, the normal report processing will be performed using the "Section 1201 report" Jasper report template files.
	3.2.3.1	Create Test Reports	RIReports(5.7.1.2.3)
SRS52: While the report is being generated, the RI shall provide a visual confirmation to the user that the report is being generated.	3.2.3.1	Create Test Reports	BasicGUIActionWrapper(5.1.1.2.3) ProgressUI(5.1.1.2.4) ReportsCoordinator(5.1.3.2.1)

Requirement	SRS Section	Title	Design
SRS53: The RI shall provide a visual confirmation to the Test Analyst when the RI has completed the generation of the report.	3.2.3.1	Create Test Reports	RIReports(5.7.1.2.3)
SRS54: The RI shall store the generated report in PDF format.	3.2.3.1	Create Test Reports	JasperReports(5.7.1.2.2) RIReports(5.7.1.2.3)
SRS55: After the RI has generated the requested report, the RI shall provide the Test Analyst with an option to show the report on the RI display.	3.2.3.1	Create Test Reports	ReportsCoordinator(5.1.3.2.1) TestReportsPanel(5.1.3.2.3) The TestReportPanel provides a pushbutton for the operator to click in order to view the report. When this button is clicked, the ReportsCoordinator brings up a viewer with the report.
SRS56: If the Test Analyst requests that the report be displayed, the RI shall present the report on the RI display. SRS56.1: When report information exceeds the limits of the display, the RI shall display scroll bars (vertical and horizontal) which provide the Test Analyst with access to all report information.	3.2.3.1	Create Test Reports	IcePDF(5.7.1.2.1) PDFViewUI(5.1.3.2.5)
SRS57: If the Test Analyst requests that the report be printed, the RI shall print the report with the following characteristics: SRS57.1: Field headings SRS57.3: Report footers which include the date and time of the report SRS57.4: Report footers which include the page number of the report in (x of y) format where x is the current page number and y is the total page count.	3.2.3.1	Create Test Reports	IcePDF(5.7.1.2.1) JasperReports(5.7.1.2.2)

Requirement	SRS Section	Title	Design
SRS58: When the Test Analyst requests to export a Test Log, the RI shall display to the Test Analyst the following options: SRS58.1: an option to select a Test Log from a displayed list of previously stored Test Logs SRS58.2: an option to export the selected Test Log to a file SRS58.3: that shows a field to enter a unique export file name SRS58.5: an option to choose to cancel the request	3.2.3.2	Export Test Log	TestLogReportPane(5.1.3.2.4) TestReportsPanel(5.1.3.2.3)
SRS59: When the Test Analyst requests to export the selected Test Log with a file name that is valid for the underlying Operating System, the RI shall store the selected Test Log in CSV format.	3.2.3.2	Export Test Log	JasperReports(5.7.1.2.2)
SRS60: The RI shall store the Test Log export file using the unique report name as the export file name.	3.2.3.2	Export Test Log	RIReports(5.7.1.2.3) TestLogReportPane(5.1.3.2.4)
SRS61: Within 2 seconds of a user request, the RI shall either: SRS61.1: complete the request SRS61.2: display feedback to the user on the status of the request	3.3.1	User Responsiveness	ProgressUI(5.1.1.2.4)
SRS62: The RI shall obtain confirmation from the user prior to the deletion of any file.	3.4.1	User Interface	MaintenanceUI(5.1.5.2.1) ReportsCoordinator(5.1.3.2.1) TestConfigCreateUI(5.1.2.2.11) TestDefinitionCoordinator(5.3.6.2.3)

Requirement	SRS Section	Title	Design
SRS63: The RI shall obtain confirmation from the user prior to replacement of a file.	3.4.1	User Interface	MaintenanceUI(5.1.5.2.1) ReportsCoordinator(5.1.3.2.1) TestConfigCreateUI(5.1.2.2.11) TestDefinitionCoordinator(5.3.6.2.3)
SRS64: The RI design shall include a provision to allow for the addition of updated or additional ITS C2C protocols.	3.4.2	C2C Protocols	ApplicationLayerService(5.3.2.2.2) ApplicationLayerStandardFactory(5.3.2.2.3) InformationLayerService(5.8.1.2.1) InformationLayerStandardFactory(5.8.1.2.2) TestSuite(5.3.2.2.5.1) TestSuites(5.3.2.2.5)
SRS65: The RI software files shall be packaged as a single compressed file (such as windows installer, rpm, etc.). SRS65.1: The uncompressed set of files shall contain a README file containing guidance for installing and running the software.	3.5.3	Packaging Requirements	The RI software is developed in Java using a combination of custom software and open source software. All of this software is bundled into a single installer.
SRS66: RI software shall be provided as a self-installing executable on CD(s) or DVD(s) SRS66.1: All necessary utilities, COTS products, and documentation will be installed with the RI software.	3.5.3	Packaging Requirements	The RI software is developed in Java using a combination of custom software and open source software. All of this software is bundled into a single installer.
	3.5.4	Legal Requirements	The RI software is developed in Java using a combination of custom software and open source software. Copyright laws and license agreements will be honored by the usage of the software and the manner in which it will be deployed.

Requirement	SRS Section	Title	Design
SRS68: The RI shall be designed to store conformance tests for multiple versions of a standard.	3.5.5	Maintainability	TestSuite(5.3.2.2.5.1) TestSuites(5.3.2.2.5) The software has been commented according to the software standards selected for the project.
SRS69: All code shall be fully documented both within the source and within software development documentation according to the software standards selected for the project.	3.5.5	Maintainability	The software has been commented according to the software standards selected for the project.
SRS70: During the Design Process, the following RI elements shall be prototyped for review and approval, prior to implementation: SRS70.1: RI user interfaces SRS70.3: Test Reports	3.6	PROJECT REQUIREMENTS	The RI software is developed in Java using a combination of custom software and open source software. All of this software is bundled into a single installer.
	3.6	PROJECT REQUIREMENTS	Deleted
SRS72: A software documentation and coding standard shall be selected by the developer and approved by FHWA prior to the development phase of the RI software. SRS72.1: The coding standard will include the following: SRS72.1.1: Each function shall be commented with pre- and post-conditions. SRS72.1.2: All program files shall include comments concerning authorship and date of last change.	3.6	PROJECT REQUIREMENTS	A software standards plan was selected for the project during earlier releases.

Requirement	SRS Section	Title	Design
	3.6	PROJECT REQUIREMENTS	Deleted
SRS74: "The current user interface focus shall be programmatically exposed so that assistive technology present in the operating system can track focus and focus changes."	3.4.1	User Interface	Java(5.11) Java supports the assistive technology present within the underlying operating system, which allows this requirement to be fulfilled.
SRS75: "Sufficient information about a user interface element including the identity, operation and state of the element shall be available to assistive technology present in the operating system."	3.4.1	User Interface	Java(5.11) Java supports the assistive technology present within the underlying operating system, which allows this requirement to be fulfilled.
SRS76: "When an image represents a program element, the information conveyed by the image shall also be available in text."	3.4.1	User Interface	Java(5.11) Java supports the assistive technology present within the underlying operating system, which allows this requirement to be fulfilled.
SRS77: "When bitmap images are used to identify controls, status indicators, or other programmatic elements, the meaning assigned to those images shall be consistent throughout an RI's performance."	3.4.1	User Interface	Java(5.11) Java supports the assistive technology present within the underlying operating system, which allows this requirement to be fulfilled.

Requirement	SRS Section	Title	Design
SRS78: "Textual information shall be provided through operating system functions for displaying text." SRS78.1: "The minimum information that shall be made available is text content, text input caret location, and text attributes."	3.4.1	User Interface	Java(5.11) Java supports the assistive technology present within the underlying operating system, which allows this requirement to be fulfilled.
SRS79: "The user interface shall not override user selected contrast and color selections and other individual display attributes."	3.4.1	User Interface	Java(5.11) The software design does not include any provisions to override contrast and color selections.
SRS80: "When animation is displayed, the information shall be displayable in at least one non-animated presentation mode at the option of the user."	3.4.1	User Interface	Java(5.11) The C2C RI's interaction with the User Interface goes through Java. No animation is used within the C2C RI Software design.
SRS81: "Color coding shall not be used as the only means of conveying information, indicating an action, prompting a response, or distinguishing a visual element."	3.4.1	User Interface	Java(5.11) The C2C RI's interaction with the User Interface goes through Java. Within the GUI design no color coding is used.
SRS82: The user interface shall not cause objects to flash or blink at a frequency greater than 2 Hz and lower than 55 Hz.	3.4.1	User Interface	Java(5.11) The C2C RI's interaction with the User Interface goes through Java. The GUI does not include features that flash or blink.

Requirement	SRS Section	Title	Design
SRS83: The RI shall provide a communications interface with the SUT that is conformant with the following standards: SRS83.1: Information Layer Standards SRS83.1.1: ATIS (SAE J2354) - (Version to be specified in the future) SRS83.1.2: IEEE 1512 - (Version to be specified in the future) SRS83.2: Application Layer Standards SRS83.2.1: NTCIP 2304 –(Version to be specified in the future)	3.1.4.1	RI- External SUT Interface	This work has not been authorized.
SRS84: When the Test Developer requests that an existing Test Configuration be opened, the RI shall display a Test Configuration prompt to the Test Developer: SRS84.1: that shows an option to select previously stored test configuration from a displayed list of Test Configurations SRS84.2: that shows an option to choose when opening the Test Configuration using name selected (see SRS84.1) SRS84.3: that shows an option to choose when cancelling the request to open the Test Configuration	3.2.1.1.1	Create, Select or Delete a Test Configuration	C2CMainUI(5.1.1.2.1) Java(5.11) RIUICoordinator(5.1.1.2.2) TestDefinitionCoordinator(5.3.6.2.3)
SRS85: The RI System shall utilize all user-defined Test Suites satisfying the following conditions: SRS85.1: the user-defined Test Suites are stored in a pre-designated storage location prior to RI application startup SRS85.2: verify that the Test Suites conform to the defined Test Suite specification	3.2.1.1.5	Test Suites	TestSuite(5.3.2.2.5.1) TestSuites(5.3.2.2.5)
SRS86: When the RI is running a Test, NTCIP 2304 is selected as the Application Layer Standard and Owner Center is the Center Mode specified in the Test Configuration, the RI shall perform the following types of actions as specified in a Test Script: SRS86.1: Connect – establish a connection with the SUT acting as an External Center SRS86.2: Disconnect – disable the connection with the SUT acting as	3.2.2.3	Perform Test Script Actions	This work has not been authorized.

Requirement	SRS Section	Title	Design
an External Center SRS86.3: Authenticate – perform steps to authenticate the SUT SRS86.4: Respond – send a response to the SUT acting as an External Center SRS86.5: Publish – publish a message to the SUT acting as an External Center SRS86.6: Validate – validate the format and content of the message received from the SUT acting as an External Center and store the results in the Test Log SRS86.7: Pause – pause the execution of the script and await a user action SRS86.8: Terminate – end the execution of the script SRS86.9: Receive – receive a message from the SUT			
SRS87: When the RI is running a Test, NTCIP 2304 is selected as the Application Layer Standard and External Center is the Center Mode specified in the Test Configuration, the RI shall perform the following types of actions as specified in a Test Script: SRS87.1: Connect – establish a connection with the SUT acting as an Owner Center SRS87.2: Disconnect – disable the connection with the SUT acting as an Owner Center SRS87.3: Authenticate – perform steps to authenticate the RI with the SUT SRS87.4: Respond – send a response to the SUT acting as an Owner Center SRS87.5: Request – send a request to the SUT acting as an Owner Center. For testing conformance to a standards processing requirement, this action type also verifies and reports whether the processing requirement was fulfilled. SRS87.6: Subscribe – establish a subscription with the SUT acting as an Owner Center. For testing whether the Owner Center properly manages the subscription, this action type also verifies and reports	3.2.2.3	Perform Test Script Actions	This work has not been authorized.

Requirement	SRS Section	Title	Design
whether the Owner Center satisfies specified publishing conditions. SRS87.7: Validate – validate the format and content of the message received from the SUT acting as an Owner Center and store the results in the Test Log. SRS87.8: Pause – pause the execution of the script until and await a user action takes place SRS87.9: Terminate – end the execution of the script SRS87.10: Receive – receive a message from the SUT			
SRS88: When the RI is running a Test and a subscript specifies a set of actions as a subscript Post-Condition, the RI shall perform the actions within the post-condition SRS88.1: when a previous test step is not successfully completed SRS88.2: when all test steps within the subscript, prior to the post-condition, were successfully completed	3.2.2.6	Support Test Post-Conditions	PostconditionTag(5.3.1.2.2) RISubscriptTag(5.3.1.2.27)
SRS89: If every action of the subscript Post-Condition is not successfully completed, the RI shall Pause the test and notify the Test Operator.	3.2.2.6	Support Test Post-Conditions	Java(5.11) PostconditionTag(5.3.1.2.2) RITestEngine(5.2.1.1.2.2) TestExecutionCoordinator(5.2.1.1.2.3)
SRS90: Pre-defined test suites shall be provided with the RI software distribution.	3.5.3	Packaging Requirements	As part of the previous phase of the project, TMDDv303c, v303d and NTCIP 2306 test suites were developed and tested. They are included with the project distribution.
SRS91: The RI shall provide an indicator to the operator that the pre-defined test suite associated with the information or application layer test suite selected was determined to come from an authorized source.	3.2.1.1.2	Edit and Store a Test Configuration	ConfigurationPanel(5.1.2.2.6) TestConfigCreateUI(5.1.2.2.11) TestSuite(5.3.2.2.5.1)

Requirement	SRS Section	Title	Design
SRS92: When the RI is running a test, TMDDv3.03c is selected as the Information Layer Standard, and a message is received with a data element which does not comply with validation criteria specified in the Test Configuration, the RI shall report the error.	3.2.2.3	Perform Test Script Actions	PublicationECReceiveTag(5.3.1.2.31) PublicationECReplyTag(5.3.1.2.32) PublicationOCTag(5.3.1.2.33) RequestResponseECTag(5.3.1.2.42) RequestResponseOCReceiveTag(5.3.1.2.43) RequestResponseOCReplyTag(5.3.1.2.44) RIMessageVerifyTag(5.3.1.2.25) RIMessageVerifyTag(5.3.1.2.25) SubscriptionECTag(5.3.1.2.45) SubscriptionOCReceiveTag(5.3.1.2.46) SubscriptionOCReplyTag(5.3.1.2.47) TMDDAuthenticationProcessor(5.4.2.2.2.1) TMDDErrorResponseProcessor(5.5.5.2.1) TMDDv303OperationResults(5.8.1.2.11) UserVerifyMessageTag(5.3.1.2.48) UserVerifyMessageTag(5.3.1.2.48)
SRS93: The C2C RI shall allow the user to choose whether device emulation will be used during test case execution at the start of a test.	3.2.2.4	Entity Emulation Actions	RIEmulation(5.4.1.1.2.5) RIEmulationParameters(5.4.1.3.2.1) RITestEngine(5.2.1.1.2.2) TestSelectionUI(5.1.4.2.1) The user selects this parameter using the test selection user interface. The RITestEngine class logs this information in the log file along with other required test information. This logged information is included in test log reports when the RI performs tests as an OC. The RITestEngine will initialize the RIEmulation class at the start of the test with the RIEmulation parameters.
SRS94: The C2C RI shall allow the user to choose whether device emulation will be initialized at the start of each test case.	3.2.2.4	Entity Emulation Actions	RIParameters(5.2.1.1.2.4) RITestEngine(5.2.1.1.2.2) TestSelectionUI(5.1.4.2.1) The user selects this parameter using the test selection user interface. The RITestEngine class logs this information in the log file along with other required test information. This logged information is included in test log reports when the RI performs tests as an OC. If this field is set the RITestEngine

Requirement	SRS Section	Title	Design
			will initialize the RIEmulation class at the start of each selected test case with the RIEmulation parameters.
SRS95: The TMDD Pre-Defined Test Suite shall include default entity data for each of the entity emulation types (CCTV, DMS, ESS, Gate, HAR, Ramp Meter, Traffic Detector, Intersection Signal Controller, Events, Lane Control Signal, Signal Section, Video Switch) and mandatory data classes (Center Active Verification, Link, Node). The valid values for each data type are defined within the design details of TMDD Volume II: CCTV Inventory - Section 3.2.2.2 CCTV Status - Section 3.2.2.3 Center Active Verification – Section 3.2.3.1 Archived Detector Data Processing Documentation Meta Data – Section 3.2.1.2 Detector Data - Section 3.2.4.1 Detector Inventory - Section 3.2.4.3 Detector Maintenance History - Section 3.2.4.5 Detector Status - Section 3.2.4.6 DMS Font Table - Section 3.2.6.2 DMS Inventory - Section 3.2.6.4 DMS Message Appearance - Section 3.2.6.5 DMS Message Inventory - Section 3.2.6.7 DMS Status - Section 3.2.6.10 ESS Inventory - Section 3.2.7.1 ESS Observation Metadata – Section 3.2.7.2 ESS Observation Report - Section 3.2.7.3 ESS Status - Section 3.2.7.4 Events Action Log - Section 3.2.8.1 Events Index - Section 3.2.8.2 Event Updates - Section 3.2.8.4 Gate Control Schedule - Section 3.2.9.2 Gate Inventory - Section 3.2.9.3 Gate Status - Section 3.2.9.4	3.2.2.4	Entity Emulation Actions	TMDDv303InformationLayerStandard(5.8.1.2.10) The TMDD Test Suite will include default definitions for the following types of entity data items: Center Active Verification CCTVInventory CCTVStatus ArchivedDataProcessingDocumentationMetaData DetectorData DetectorInventory DetectorMaintenanceHistory DetectorStatus DMSFontTable DMSInventory DMSMessageAppearance DMSMessageInventory DMSStatus ESSInventory ESSObservationMetadata ESSObservationReport ESSStatus EventsActionLog EventsIndex EventUpdates GateControlSchedule GateInventory GateStatus HARControlSchedule HARInventory HARMessageInventory HARStatus IntersectionSignalControllerControlSchedule IntersectionSignalControllerInventory IntersectionSignalControllerStatus IntersectionSignalControllerTimingPatternInventory LaneControlSignalControlSchedule LaneControlSignalInventory LaneControlSignalStatus LinkInventory

Requirement	SRS Section	Title	Design
HAR Control Schedule - Section 3.2.10.2 HAR Inventory - Section 3.2.10.3 HAR Message Inventory - Section 3.2.10.4 HAR Status - Section 3.2.10.6 Intersection Signal Controller Control Schedule - Section 3.2.11.3 Intersection Signal Controller Inventory - Section 3.2.11.4 Intersection Signal Controller Status - Section 3.2.11.6 Intersection Signal Controller Timing Pattern Inventory - Section 3.2.11.7 Lane Control Signal Control Schedule - Section 3.2.12.2 Lane Control Signal Inventory - Section 3.2.12.3 Lane Control Signal Status - Section 3.2.12.4 Link Inventory – Section 3.2.13.1 Link Status – Section 3.2.13.2 Node Inventory – Section 3.2.14.1 Node Status – Section 3.2.14.2 Ramp Meter Control Schedule - Section 3.2.16.2 Ramp Meter Inventory - Section 3.2.16.3 Ramp Meter Plan Inventory - Section 3.2.16.4 Ramp Meter Status - Section 3.2.16.6 Section Control Schedule - Section 3.2.18.3 Section Status - Section 3.2.18.6 Section Signal Timing Pattern Inventory - Section 3.2.18.7 Video Switch Inventory - Section 3.2.20.2 Video Switch Status - Section 3.2.20.			LinkStatus NodeInventory NodeStatus RampMeterControlSchedule RampMeterInventory RampMeterPlanInventory RampMeterStatus SectionControlSchedule SectionStatus SectionSignalTimingPatternInventory VideoSwitchInventory VideoSwitchStatus The TMDD Test Suite will include default definitions for the following types of mandatory data items: CenterActiveVerification LinkInventory LinkStatus NodeInventory NodeStatus The definition of the parameters and valid values of each set of entity data is defined in TMDD Volume II. The RIEmulationParameters constructor will initially load these values from the selected TMDDv303c or newer test suite file distributed with the application.
SRS96: The C2CRI shall allow the user to override the default entity data provided by the TMDD Test Suite. See SRS95 for the list of Entity Types and references to the data elements.	3.2.2.4	Entity Emulation Actions	EntityDataViewerUI(5.1.2.2.8) EntityEmulationPanel(5.1.2.2.7) A new panel is being added to the Test Configuration editing screens to allow the user to select files with information that overrides the default entity data.

Requirement	SRS Section	Title	Design
	3.2.2.4	Entity Emulation Actions	Deleted
	3.2.2.4	Entity Emulation Actions	Deleted
SRS99: When the C2CRI receives an entity request message from an external center, when operating in device emulation mode, the C2C RI shall send a response message to the external center. The messages involved are part of the request-response dialogs associated with each data type and are defined within the design details ofTMDD Volume II (CCTV - Section 3.1.2, DMS - Section 3.1.6, ESS - Section 3.1.7, Gate - Section 3.1.9, HAR - Section 3.1.10, Ramp Meter - Section 3.1.16, Traffic Detector - Section 3.1.4, IntersectionSignal Controller - Section 3.1.11, Events - Section 3.1.8, Lane Control Signal - Section 3.1.12, Signal Section - Section 3.1.18, Video Switch - Section 3.1.20). Basic Control related request response dialogs are specified within Section 3.1.5.	3.2.2.4	Entity Emulation Actions	CCTVHandler(5.4.1.1.2.6) DetectorHandler(5.4.1.1.2.7) DMSHandler(5.4.1.1.2.8) ESSHandler(5.4.1.1.2.9) EventHandler(5.4.1.1.2.10) GateHandler(5.4.1.1.2.11) HARHandler(5.4.1.1.2.12) LaneControlSignalHandler(5.4.1.1.2.13) RampMeterHandler(5.4.1.1.2.14) RequestResponseOCReceiveTag(5.3.1.2.43) RequestResponseOCReplyTag(5.3.1.2.44) RIEmulation(5.4.1.1.2.5) SignalSectionHandler(5.4.1.1.2.15) TrafficControllerHandler(5.4.1.1.2.16) VideoSwitchHandler(5.4.1.1.2.17) Entity Emulation Overview(6.2)
	3.2.2.4	Entity Emulation Actions	Deleted

Requirement	SRS Section	Title	Design
SRS101: For a valid request (as per SRS99), the response message sent from the RI shall conform to the project specification defined within the Test Configuration.	3.2.2.4	Entity Emulation Actions	CCTVHandler(5.4.1.1.2.6) DetectorHandler(5.4.1.1.2.7) DMSHandler(5.4.1.1.2.8) ESSHandler(5.4.1.1.2.9) EventHandler(5.4.1.1.2.10) GateHandler(5.4.1.1.2.11) HARHandler(5.4.1.1.2.12) LaneControlSignalHandler(5.4.1.1.2.13) RampMeterHandler(5.4.1.1.2.14) SignalSectionHandler(5.4.1.1.2.15) TrafficControllerHandler(5.4.1.1.2.16) VideoSwitchHandler(5.4.1.1.2.17) Entity Emulation Overview(6.2)
SRS102: The C2C RI TMDD entity emulation shall support filtering requests from the external center. Only items that are included in the request will be included in the response. The TMDD does not specify the type of response the Owner Center should send when one or more information matches in the requested filter is not available. In this event, an error response will be provided indicating that no data is available.	3.2.2.4	Entity Emulation Actions	CCTVHandler(5.4.1.1.2.6) DetectorHandler(5.4.1.1.2.7) DMSHandler(5.4.1.1.2.8) ESSHandler(5.4.1.1.2.9) EventHandler(5.4.1.1.2.10) GateHandler(5.4.1.1.2.11) HARHandler(5.4.1.1.2.12) LaneControlSignalHandler(5.4.1.1.2.13) RampMeterHandler(5.4.1.1.2.14) SignalSectionHandler(5.4.1.1.2.15) TrafficControllerHandler(5.4.1.1.2.16) VideoSwitchHandler(5.4.1.1.2.17) Entity Emulation Overview(6.2) Section 6.2 includes a table describing dialog handling. This description includes some discussion on filtering.
SRS103: The C2C RI TMDD entity emulation shall support device command queues for DMS, HAR, Ramp Meter, Intersection Signal Controller and Signal Section device types.	3.2.2.4	Entity Emulation Actions	CCTVHandler(5.4.1.1.2.6) CommandQueue(5.4.1.1.2.4) DMSHandler(5.4.1.1.2.8) GateHandler(5.4.1.1.2.11) HARHandler(5.4.1.1.2.12) LaneControlSignalHandler(5.4.1.1.2.13) RampMeterHandler(5.4.1.1.2.14) SignalSectionHandler(5.4.1.1.2.15) TrafficControllerHandler(5.4.1.1.2.16) VideoSwitchHandler(5.4.1.1.2.17) Entity Emulation Overview(6.2)

Requirement	SRS Section	Title	Design
SRS104: The C2CRI shall provide a means for the user to specify the length for device command control queues for TMDD device emulation of DMS, HAR, Ramp Meter, Intersection Signal Controller and Signal Section device types. The default length is 0.	3.2.2.4	Entity Emulation Actions	CommandQueue(5.4.1.1.2.4) EntityDataViewerUI(5.1.2.2.8) RIEmulationParameters(5.4.1.3.2.1)
SRS105: The C2CRI shall queue command control requests based on the specified queue length. (Reference SRS104)	3.2.2.4	Entity Emulation Actions	CCTVHandler(5.4.1.1.2.6) CommandQueue(5.4.1.1.2.4) DMSHandler(5.4.1.1.2.8) GateHandler(5.4.1.1.2.11) HARHandler(5.4.1.1.2.12) LaneControlSignalHandler(5.4.1.1.2.13) RampMeterHandler(5.4.1.1.2.14) SignalSectionHandler(5.4.1.1.2.15) TrafficControllerHandler(5.4.1.1.2.16) VideoSwitchHandler(5.4.1.1.2.17) Entity Emulation Overview(6.2) Section 6.2 includes a discussion on command handling. This section includes a description on how command queues will be managed.
SRS106: Once the command queue length has been exceeded, the C2CRI device emulation shall change the state of the device based off of the next valid queued control request.	3.2.2.4	Entity Emulation Actions	CCTVHandler(5.4.1.1.2.6) CommandQueue(5.4.1.1.2.4) DMSHandler(5.4.1.1.2.8) GateHandler(5.4.1.1.2.11) HARHandler(5.4.1.1.2.12) LaneControlSignalHandler(5.4.1.1.2.13) RampMeterHandler(5.4.1.1.2.14) SignalSectionHandler(5.4.1.1.2.15) TrafficControllerHandler(5.4.1.1.2.16) VideoSwitchHandler(5.4.1.1.2.17) Entity Emulation Overview(6.2) Section 6.2 includes a discussion on command handling. This section includes a description on how command queues will be managed.

Requirement	SRS Section	Title	Design
	3.2.2.4	Entity Emulation Actions	Deleted
	3.2.2.4	Entity Emulation Actions	Deleted
SRS109: When acting as an Owner Center and operating in TMDD Entity Emulation mode, the C2C RI shall begin responding to requests for CCTV information and control, per TMDD Volume 1 Section 3.4.2, after receipt from the External Center. SRS109.1: When the C2C RI receives a valid cCTVControlRequestMsg Message for a CCTV device from an External Center as part of a dlCCTVControlRequest dialog the C2C RI shall send a deviceControlResponseMsg response Message to the External Center. SRS109.1.1: The C2C RI shall update the CCTV Status information (See SRS95 and SRS96) when a valid control command has been completed. SRS109.2: When the C2C RI receives a valid deviceInformationRequestMsg Message for a CCTV device from an External Center as part of a dlCCTVInventoryRequest dialog the C2C RI shall send a cCTVInventoryMsg response Message to the External Center. SRS109.3: When the C2C RI receives a valid deviceInformationRequestMsg	3.2.2.4.1	CCTV	CCTVHandler(5.4.1.1.2.6) RequestResponseOCReceiveTag(5.3.1.2.43) RequestResponseOCReplyTag(5.3.1.2.44) RIEmulation(5.4.1.1.2.5) ApplicationLayerStandard(5.3.2.2.4) Entity Emulation Overview(6.2) Note that the C2C RI's ability to satisfy this performance requirement will be impacted by the performance of the underlying selected application layer standard.

Requirement	SRS Section	Title	Design
Message for a CCTV device from an External Center as part of a dlCCTVStatusRequest dialog the C2C RI shall send a cCTVStatusMsg response Message to the External Center. SRS109.4: When the C2C RI receives a valid deviceCancelControlRequestMsg Message for a CCTV device from an External Center as part of a dlDeviceCancelControlRequest dialog the C2C RI shall send a deviceControlResponseMsg response Message to the External Center. SRS109.5: When the C2C RI receives a valid deviceControlStatusRequestMsg Message for a CCTV device from an External Center as part of a dlDeviceControlStatusRequest dialog the C2C RI shall send a deviceControlResponseMsg response Message to the External Center.			
SRS110: When acting as an Owner Center and TMDD Entity Emulation is selected, the C2C RI shall begin responding to a subscription request for CCTV information, per TMDD Volume 1 Section 3.4.2, after receipt from the External Center. SRS110.1: When the C2C RI receives a valid deviceInformationRequestMsg Message for a CCTV device from an External Center as part of a dlDeviceInformationSubscription dialog the C2C RI shall send a c2cMessageReceipt response Message to the External Center.	3.2.2.4.1	CCTV	CCTVHandler(5.4.1.1.2.6) SubscriptionOCReceiveTag(5.3.1.2.46) SubscriptionOCReplyTag(5.3.1.2.47) ApplicationLayerStandard(5.3.2.2.4) Entity Emulation Overview(6.2) Note that the C2C RI's ability to satisfy this performance requirement will be impacted by the performance of the underlying selected application layer standard.
SRS111: When acting as an Owner Center and TMDD Entity Emulation is selected, the C2C RI shall publish CCTV information updates to a subscribing External Center. SRS111.1: The C2C RI shall publish a cCTVInventoryMsg Message as part of a dlCCTVInventoryUpdate dialog to an External Center in accordance with current subscriptions. SRS111.1.1: If the current	3.2.2.4.1	CCTV	CCTVHandler(5.4.1.1.2.6) RIEmulation(5.4.1.1.2.5) ApplicationLayerStandard(5.3.2.2.4) Entity Emulation Overview(6.2) Note that the C2C RI's ability to satisfy this performance requirement will be impacted by the performance of the underlying selected application layer standard.

Requirement	SRS Section	Title	Design
subscription is for event-driven updates, the C2C RI shall initiate the publication within 1 minute of a change in CCTV Inventory (See Requirement SRS95) when the changed data item(s) are included within the original subscription request. SRS111.2: The C2C RI shall publish a cCTVStatusMsg Message as part of a dlCCTVStatusUpdate dialog to an External Center in accordance with current subscriptions. SRS111.2.1: If the current subscription is for event-driven updates, the C2C RI shall initiate the publication within 1 minute of a change in CCTV Status (See Requirement SRS95) when the changed data item(s) are included within the original subscription request.			
SRS112: When acting as an Owner Center and TMDD Entity Emulation is selected, the C2C RI shall begin responding to requests for Detector information and control, per TMDD Volume 1 Section 3.4.2, after receipt from the External Center. SRS112.1: When the C2C RI receives a valid archivedDataProcessingDocumentationMetadataRequestMsg Message for a Detector device from an External Center as part of a dlArchivedDataProcessingDocumentationMetadataRequest dialog the C2C RI shall send a archivedDataProcessingDocumentationMetadataMsg response Message to the External Center. SRS112.2: When the C2C RI receives a valid detectorDataRequestMsg Message for a Detector device from an External Center as part of a dlDetectorDataRequest dialog the C2C RI shall send a detectorDataMsg response Message to the External Center. SRS112.3: When the C2C RI receives a valid deviceInformationRequestMsg Message for a Detector device from	3.2.2.4.2	Detector	DetectorHandler(5.4.1.1.2.7) RIEmulation(5.4.1.1.2.5) ApplicationLayerStandard(5.3.2.2.4) Entity Emulation Overview(6.2) Note that the C2C RI's ability to satisfy this performance requirement will be impacted by the performance of the underlying selected application layer standard.

Requirement	SRS Section	Title	Design
an External Center as part of a dlDetectorInventoryRequest dialog the C2C RI shall send a detectorInventoryMsg response Message to the External Center. SRS112.4: When the C2C RI receives a valid detectorMaintenanceHistoryRequestMsg Message for a Detector device from an External Center as part of a dlDetectorMaintenanceHistoryRequest dialog the C2C RI shall send a detectorMaintenanceHistoryMsg response Message to the External Center. SRS112.5: When the C2C RI receives a valid deviceInformationRequestMsg Message for a Detector device from an External Center as part of a dlDetectorStatusRequest dialog the C2C RI shall send a detectorStatusMsg response Message to the External Center.bscription. Updates to Detector data will not be supported.			
SRS113: When acting as an Owner Center and TMDD Entity Emulation is selected, the C2C RI shall begin responding to a subscription request for Detector information, per TMDD Volume 1 Section 3.4.2, after receipt from the External Center. SRS113.1: When the C2C RI receives a valid deviceInformationRequestMsg Message for a Detector device from an External Center as part of a dlDetectorDataSubscription dialog the C2C RI shall send a c2cMessageReceipt response Message to the External Center. SRS113.2: When the C2C RI receives a valid deviceInformationRequestMsg Message for a Detector device from an External Center as part of a dlDeviceInformationSubscription dialog the C2C RI shall send a c2cMessageReceipt response Message to the External Center.	3.2.2.4.2	Detector	DetectorHandler(5.4.1.1.2.7) RIEmulation(5.4.1.1.2.5) SubscriptionOCReceiveTag(5.3.1.2.46) SubscriptionOCReplyTag(5.3.1.2.47) ApplicationLayerStandard(5.3.2.2.4) Entity Emulation Overview(6.2) Note that the C2C RI's ability to satisfy this performance requirement will be impacted by the performance of the underlying selected application layer standard.

Requirement	SRS Section	Title	Design
SRS114: When acting as an Owner Center and TMDD Entity Emulation is selected, the C2C RI shall publish Detector information updates to a subscribing External Center. SRS114.1: The C2C RI shall publish a detectorDataMsg Message as part of a dlDetectorDataUpdate dialog to an External Center in accordance with current subscriptions. SRS114.1.1: If the current subscription is for event-driven updates, the C2C RI shall initiate the publication within 1 minute of a change in Detector Data (See Requirement SRS95) when the changed data item(s) are included within the original subscription request. SRS114.2: The C2C RI shall publish a detectorInventoryMsg Message as part of a dlDetectorInventoryUpdate dialog to an External Center in accordance with current subscriptions. SRS114.2.1: If the current subscription is for event-driven updates, the C2C RI shall initiate the publication within 1 minute of a change in Detector Inventory (See Requirement SRS95) when the changed data item(s) are included within the original subscription request. SRS114.3: The C2C RI shall publish a detectorStatusMsg Message as part of a dlDetectorStatusUpdate dialog to an External Center in accordance with current subscriptions. SRS114.3.1: If the current subscription is for event-driven updates, the C2C RI shall initiate the publication within 1 minute of a change in Detector Status (See Requirement SRS95) when the changed data item(s) are included within the original subscription request.	3.2.2.4.2	Detector	DetectorHandler(5.4.1.1.2.7) PublicationOCTag(5.3.1.2.33) RIEmulation(5.4.1.1.2.5) ApplicationLayerStandard(5.3.2.2.4) Entity Emulation Overview(6.2) Note that the C2C RI's ability to satisfy this performance requirement will be impacted by the performance of the underlying selected application layer standard.

Requirement	SRS Section	Title	Design
SRS115: When acting as an Owner Center and TMDD Entity Emulation is selected, the C2C RI shall begin responding to requests for DMS information and control, per TMDD Volume 1 Section 3.4.2, after receipt from the External Center. SRS115.1: When the C2C RI receives a valid deviceCancelControlRequestMsg Message for a DMS device from an External Center as part of a dlDeviceCancelControlRequest dialog the C2C RI shall send a deviceControlResponseMsg response Message to the External Center. SRS115.2: When the C2C RI receives a valid deviceControlStatusRequestMsg Message for a DMS device from an External Center as part of a dlDeviceControlStatusRequest dialog the C2C RI shall send a deviceControlResponseMsg response Message to the External Center. SRS115.3: When the C2C RI receives a valid dMSControlRequestMsg Message for a DMS device from an External Center as part of a dlDMSControlRequest dialog the C2C RI shall send a deviceControlResponseMsg response Message to the External Center. SRS115.3.1: The C2C RI shall update the DMS Status information (See SRS95 and SRS96) when a valid control command has been completed. SRS115.4: When the C2C RI receives a valid dMSFontTableRequestMsg Message for a DMS device from an External Center as part of a dlDMSFontTableRequest dialog the C2C RI shall send a dMSFontTableMsg response Message to the External Center. SRS115.5: When the C2C RI receives a valid deviceInformationRequestMsg	3.2.2.4.3	Dynamic Message Sign (DMS)	DMSHandler(5.4.1.1.2.8) RequestResponseOCReceiveTag(5.3.1.2.43) RequestResponseOCReplyTag(5.3.1.2.44) RIEmulation(5.4.1.1.2.5) ApplicationLayerStandard(5.3.2.2.4) Entity Emulation Overview(6.2) Section 6.2 includes a table which describe how control request information gets mapped to data that will show up in response messages. Note that the C2C RI's ability to satisfy this performance requirement will be impacted by the performance of the underlying selected application layer standard.

Requirement	SRS Section	Title	Design
Message for a DMS device from an External Center as part of a dlDMSInventoryRequest dialog the C2C RI shall send a dMSInventoryMsg response Message to the External Center. SRS115.6: When the C2C RI receives a valid dMSMessageAppearanceRequestMsg Message for a DMS device from an External Center as part of a dlDMSMessageAppearanceRequest dialog the C2C RI shall send a dMSMessageAppearanceMsg response Message to the External Center. SRS115.7: When the C2C RI receives a valid dMSMessageInventoryRequestMsg Message for a DMS device from an External Center as part of a dlDMSMessageInventoryRequest dialog the C2C RI shall send a dMSMessageInventoryMsg response Message to the External Center. SRS115.8: When the C2C RI receives a valid devicePriorityQueueRequestMsg Message for a DMS device from an External Center as part of a dlDMSPriorityQueueRequest dialog the C2C RI shall send a dMSPriorityQueueMsg response Message to the External Center. SRS115.9: When the C2C RI receives a valid deviceInformationRequestMsg Message for a DMS device from an External Center as part of a dlDMSStatusRequest dialog the C2C RI shall send a dMSStatusMsg response Message to the External Center.			
SRS116: When acting as an Owner Center and TMDD Entity Emulation is selected, the C2C RI shall begin responding to a subscription request for DMS information, per TMDD Volume 1 Section 3.4.2, after receipt from the External Center. SRS116.1: When the C2C RI receives a valid deviceInformationRequestMsg Message for a DMS device from an	3.2.2.4.3	Dynamic Message Sign (DMS)	CCTVHandler(5.4.1.1.2.6) CCTVInventory(5.4.1.2.2.1.1) CCTVStatus(5.4.1.2.2.1.2) DMSHandler(5.4.1.1.2.8) RIEmulation(5.4.1.1.2.5) SubscriptionOCReceiveTag(5.3.1.2.46) SubscriptionOCReplyTag(5.3.1.2.47) ApplicationLayerStandard(5.3.2.2.4) Entity Emulation Overview(6.2)

Requirement	SRS Section	Title	Design
External Center as part of a dlDeviceInformationSubscription dialog the C2C RI shall send a c2cMessageReceipt response Message to the External Center. SRS116.2: When the C2C RI receives a valid dMSMessageInventoryRequestMsg Message for a DMS device from an External Center as part of a dlDMSMessageInventorySubscription dialog the C2C RI shall send a c2cMessageReceipt response Message to the External Center.			Section 6.2 includes a description on how the messages (and data) in the dialogs are processed. The content is based on the message definitions included in TMDD Volume II. Unless the values are modified as part of the control request processing or a script initiated value change, the parameter values will be as specified in the Test Configuration at the start of a test. Note that the C2C RI's ability to satisfy this performance requirement will be impacted by the performance of the underlying selected application layer standard.
SRS117: When acting as an Owner Center and TMDD Entity Emulation is selected, the C2C RI shall begin responding to requests for ESS information and control, per TMDD Volume 1 Section 3.4.2, after receipt from the External Center. SRS117.1: When the C2C RI receives a valid deviceInformationRequestMsg Message for a ESS device from an External Center as part of a dlESSInventoryRequest dialog the C2C RI shall send a eSSInventoryMsg response Message to the External Center. SRS117.2: When the C2C RI receives a valid deviceInformationRequestMsg Message for a ESS device from an External Center as part of a dlESSObservationMetadataRequest dialog the C2C RI shall send a eSSObservationMetadataMsg response Message to the External Center. SRS117.3: When the C2C RI receives a valid deviceInformationRequestMsg Message for a ESS device from an External Center as part of a dlESSObservationReportRequest dialog the C2C RI shall send a eSSObservationReportMsg response Message to the External Center. SRS117.4: When the C2C RI	3.2.2.4.4	Environmental Sensor Station (ESS)	CCTVHandler(5.4.1.1.2.6) ESSHandler(5.4.1.1.2.9) RequestResponseOCReceiveTag(5.3.1.2.43) RequestResponseOCReplyTag(5.3.1.2.44) RIEmulation(5.4.1.1.2.5) ApplicationLayerStandard(5.3.2.2.4) Entity Emulation Overview(6.2) Section 6.2 includes a description on how the messages (and data) in the dialogs are processed. The content is based on the message definitions included in TMDD Volume II. Unless the values are modified as part of the control request processing or a script initiated value change, the parameter values will be as specified in the Test Configuration at the start of a test. Note that the C2C RI's ability to satisfy this performance requirement will be impacted by the performance of the underlying selected application layer standard.

Requirement	SRS Section	Title	Design
receives a valid deviceInformationRequestMsg Message for a ESS device from an External Center as part of a dlESSStatusRequest dialog the C2C RI shall send a eSSStatusMsg response Message to the External Center.			
SRS118: When acting as an Owner Center and TMDD Entity Emulation is selected, the C2C RI shall publish ESS information updates to a subscribing External Center. SRS118.1: The C2C RI shall publish a eSSInventoryMsg Message as part of a dlESSInventoryUpdate dialog to an External Center in accordance with current subscriptions. SRS118.1.1: If the current subscription is for event-driven updates, the C2C RI shall initiate the publication within 1 minute of a change in ESS Inventory (See Requirement SRS95) when the changed data item(s) are included within the original subscription request. SRS118.2: The C2C RI shall publish a eSSObservationReportMsg Message as part of a dlESSObservationReportUpdate dialog to an External Center in accordance with current subscriptions. SRS118.2.1: If the current subscription is for event-driven updates, the C2C RI shall initiate the publication within 1 minute of a change in ESS Observation Report (See Requirement SRS95) when the changed data item(s) are included within the original subscription request. SRS118.3: The C2C RI shall publish a eSSStatusMsg Message as part of a dlESSStatusUpdate dialog to an External Center in accordance with current subscriptions. SRS118.3.1: If the current subscription is for event-driven updates, the C2C RI shall initiate the publication within 1 minute of a change in ESS Status (See	3.2.2.4.4	Environmental Sensor Station (ESS)	ArchivedDataProcessingDocumentationMetaData(5.4.1.2.2.2.1) CCTVHandler(5.4.1.1.2.6) CCTVInventory(5.4.1.2.2.1.1) CCTVStatus(5.4.1.2.2.1.2) DetectorData(5.4.1.2.2.2.2) DetectorHandler(5.4.1.1.2.7) DetectorInventory(5.4.1.2.2.2.3) DetectorMaintenanceHistory(5.4.1.2.2.2.4) DetectorStatus(5.4.1.2.2.2.5) ESSHandler(5.4.1.1.2.9) PublicationOCTag(5.3.1.2.33) RIEmulation(5.4.1.1.2.5) ApplicationLayerStandard(5.3.2.2.4) Entity Emulation Overview(6.2) Section 6.2 includes a description on how the messages (and data) in the dialogs are processed. The content is based on the message definitions included in TMDD Volume II. Unless the values are modified as part of the control request processing or a script initiated value change, the parameter values will be as specified in the Test Configuration at the start of a test. Note that the C2C RI's ability to satisfy this performance requirement will be impacted by the performance of the underlying selected application layer standard.

Requirement	SRS Section	Title	Design
Requirement SRS95) when the changed data item(s) are included within the original subscription request.			
SRS119: When acting as an Owner Center and TMDD Entity Emulation is selected, the C2C RI shall begin responding to requests for Event information, per TMDD Volume 1 Section 3.4.2, after receipt from the External Center. SRS119.1: When the C2C RI receives a valid eventRequestMsg Message for a Event device from an External Center as part of a dlActionLogRequest dialog the C2C RI shall send a actionLogMsg response Message to the External Center. SRS119.2: When the C2C RI receives a valid eventRequestMsg Message for a Event device from an External Center as part of a dlEventIndexRequest dialog the C2C RI shall send a eventIndexMsg response Message to the External Center. SRS119.3: When the C2C RI receives a valid eventRequestMsg Message for a Event device from an External Center as part of a dlFullEventUpdateRequest dialog the C2C RI shall send a fEUMsg response Message to the External Center.	3.2.2.4.5	Event	EventHandler(5.4.1.1.2.10) RequestResponseOCReceiveTag(5.3.1.2.43) RequestResponseOCReplyTag(5.3.1.2.44) RIEmulation(5.4.1.1.2.5) ApplicationLayerStandard(5.3.2.2.4) Entity Emulation Overview(6.2) Section 6.2 includes a description on how the messages (and data) in the dialogs are processed. The content is based on the message definitions included in TMDD Volume II. Unless the values are modified as part of the control request processing or a script initiated value change, the parameter values will be as specified in the Test Configuration at the start of a test. Note that the C2C RI's ability to satisfy this performance requirement will be impacted by the performance of the underlying selected application layer standard.
SRS120: When acting as an Owner Center and TMDD Entity Emulation is selected, the C2C RI shall begin responding to a subscription request for Event information, per TMDD Volume 1 Section 3.4.2, after receipt from the External Center. SRS120.1: When the C2C RI receives a valid eventRequestMsg Message for a Event device from an External Center as part of a dlActionLogSubscription dialog the C2C RI shall send a c2cMessageReceipt response	3.2.2.4.5	Event	DetectorData(5.4.1.2.2.2.2) DetectorHandler(5.4.1.1.2.7) DetectorInventory(5.4.1.2.2.2.3) DetectorStatus(5.4.1.2.2.2.5) EventHandler(5.4.1.1.2.10) RIEmulation(5.4.1.1.2.5) SubscriptionOCReceiveTag(5.3.1.2.46) SubscriptionOCReplyTag(5.3.1.2.47) ApplicationLayerStandard(5.3.2.2.4) Entity Emulation Overview(6.2)

Requirement	SRS Section	Title	Design
Message to the External Center. SRS120.2: When the C2C RI receives a valid eventRequestMsg Message for a Event device from an External Center as part of a dlEventIndexSubscription dialog the C2C RI shall send a c2cMessageReceipt response Message to the External Center. SRS120.3: When the C2C RI receives a valid eventRequestMsg Message for a Event device from an External Center as part of a dlFullEventUpdateSubscription dialog the C2C RI shall send a c2cMessageReceipt response Message to the External Center.			Section 6.2 includes a description on how the messages (and data) in the dialogs are processed. The content is based on the message definitions included in TMDD Volume II. Unless the values are modified as part of the control request processing or a script initiated value change, the parameter values will be as specified in the Test Configuration at the start of a test. Note that the C2C RI's ability to satisfy this performance requirement will be impacted by the performance of the underlying selected application layer standard.
SRS121: When acting as an Owner Center and TMDD Entity Emulation is selected, the C2C RI shall publish Event information updates to a subscribing External Center. SRS121.1: The C2C RI shall publish an actionLogMsg Message as part of a dlActionLogUpdate dialog to an External Center in accordance with current subscriptions. SRS121.1.1: If the current subscription is for event-driven updates, the C2C RI shall initiate the publication within 1 minute of a change in Events Action Log (See Requirement SRS95) when the changed data item(s) are included within the original subscription request. SRS121.2: The C2C RI shall publish an eventIndexMsg Message as part of a dlEventIndexUpdate dialog to an External Center in accordance with current subscriptions. SRS121.2.1: If the current subscription is for event-driven updates, the C2C RI shall initiate the publication within 1 minute of a change in Events Index (See Requirement SRS95) when the changed data item(s) are included within the original subscription request. SRS121.3: The C2C RI shall publish a fEUMsg Message as part of a dlFullEventUpdateUpdate dialog to an External Center in accordance	3.2.2.4.5	Event	DetectorData(5.4.1.2.2.2.2) DetectorHandler(5.4.1.1.2.7) DetectorInventory(5.4.1.2.2.2.3) DetectorMaintenanceHistory(5.4.1.2.2.2.4) EventHandler(5.4.1.1.2.10) PublicationOCTag(5.3.1.2.33) RIEmulation(5.4.1.1.2.5) ApplicationLayerStandard(5.3.2.2.4) Entity Emulation Overview(6.2) Section 6.2 includes a description on how the messages (and data) in the dialogs are processed. The content is based on the message definitions included in TMDD Volume II. Unless the values are modified as part of the control request processing or a scritp initiated value change, the parameter values will be as specified in the Test Configuration at the start of a test. Note that the C2C RI's ability to satisfy this performance requirement will be impacted by the performance of the underlying selected application layer standard.

Requirement	SRS Section	Title	Design
with current subscriptions. SRS121.3.1: If the current subscription is for event-driven updates, the C2C RI shall initiate the publication within 1 minute of a change in Events Updates (See Requirement SRS95) when the changed data item(s) are included within the original subscription request.			
SRS122: When acting as an Owner Center and TMDD Entity Emulation is selected, the C2C RI shall begin responding to requests for Gate information and control, per TMDD Volume 1 Section 3.4.2, after receipt from the External Center. SRS122.1: When the C2C RI receives a valid deviceCancelControlRequestMsg Message for a Gate device from an External Center as part of a dlDeviceCancelControlRequest dialog the C2C RI shall send a deviceControlResponseMsg response Message to the External Center. SRS122.2: When the C2C RI receives a valid deviceControlStatusRequestMsg Message for a Gate device from an External Center as part of a dlDeviceControlStatusRequest dialog the C2C RI shall send a deviceControlResponseMsg response Message to the External Center. SRS122.3: When the C2C RI receives a valid gateControlRequestMsg Message for a Gate device from an External Center as part of a dlGateControlRequest dialog the C2C RI shall send a deviceControlResponseMsg response Message to the External Center. SRS122.3.1: The C2C RI shall update the Gate Status information (See SRS95 and SRS96) when a valid control command has been completed. SRS122.4: When the C2C RI receives a valid	3.2.2.4.6	Lane Closure Gate	DMSFontTable(5.4.1.2.2.3.1) DMSHandler(5.4.1.1.2.8) DMSInventory(5.4.1.2.2.3.2) DMSMessageAppearance(5.4.1.2.2.3.3) DMSMessageInventory(5.4.1.2.2.3.4) DMSStatus(5.4.1.2.2.3.5) GateHandler(5.4.1.1.2.11) RequestResponseOCReceiveTag(5.3.1.2.43) RequestResponseOCReplyTag(5.3.1.2.44) RIEmulation(5.4.1.1.2.5) ApplicationLayerStandard(5.3.2.2.4) Entity Emulation Overview(6.2) Section 6.2 includes a description on how the messages (and data) in the dialogs are processed. The content is based on the message definitions included in TMDD Volume II. Unless the values are modified as part of the control request processing or a script initiated value change, the parameter values will be as specified in the Test Configuration at the start of a test. Note that the C2C RI's ability to satisfy this performance requirement will be impacted by the performance of the underlying selected application layer standard.

Requirement	SRS Section	Title	Design
deviceInformationRequestMsg Message for a Gate device from an External Center as part of a dlGateControlScheduleRequest dialog the C2C RI shall send a gateControlScheduleMsg response Message to the External Center. SRS122.5: When the C2C RI receives a valid deviceInformationRequestMsg Message for a Gate device from an External Center as part of a dlGateInventoryRequest dialog the C2C RI shall send a gateInventoryMsg response Message to the External Center. SRS122.6: When the C2C RI receives a valid deviceInformationRequestMsg Message for a Gate device from an External Center as part of a dlGateStatusRequest dialog the C2C RI shall send a gateStatusMsg response Message to the External Center.			
SRS123: When acting as an Owner Center and TMDD Entity Emulation is selected, the C2C RI shall begin responding to a subscription request for Gate information, per TMDD Volume 1 Section 3.4.2, after receipt from the External Center. SRS123.1: When the C2C RI receives a valid deviceInformationRequestMsg Message for a Gate device from an External Center as part of a dlDeviceInformationSubscription dialog the C2C RI shall send a c2cMessageReceipt response Message to the External Center.	3.2.2.4.6	Lane Closure Gate	GateHandler(5.4.1.1.2.11) RIEmulation(5.4.1.1.2.5) SubscriptionOCReceiveTag(5.3.1.2.46) SubscriptionOCReplyTag(5.3.1.2.47) ApplicationLayerStandard(5.3.2.2.4) Entity Emulation Overview(6.2) Section 6.2 includes a description on how the messages (and data) in the dialogs are processed. The content is based on the message definitions included in TMDD Volume II. Unless the values are modified as part of the control request processing or a script initiated value change, the parameter values will be as specified in the Test Configuration at the start of a test. Note that the C2C RI's ability to satisfy this performance requirement will be impacted by the performance of the underlying selected application layer standard.

Requirement	SRS Section	Title	Design
SRS124: When acting as an Owner Center and TMDD Entity Emulation is selected, the C2C RI shall publish Gate information updates to a subscribing External Center. SRS124.1: The C2C RI shall publish a gateControlScheduleMsg Message as part of a dlGateControlScheduleUpdate dialog to an External Center in accordance with current subscriptions. SRS124.1.1: If the current subscription is for event-driven updates, the C2C RI shall initiate the publication within 1 minute of a change in Gate Control Schedule (See Requirement SRS95) when the changed data item(s) are included within the original subscription request. SRS124.2: The C2C RI shall publish a gateInventoryMsg Message as part of a dlGateInventoryUpdate dialog to an External Center in accordance with current subscriptions. SRS124.2.1: If the current subscription is for event-driven updates, the C2C RI shall initiate the publication within 1 minute of a change in Gate Inventory (See Requirement SRS95) when the changed data item(s) are included within the original subscription request. SRS124.3: The C2C RI shall publish a gateStatusMsg Message as part of a dlGateStatusUpdate dialog to an External Center in accordance with current subscriptions. SRS124.3.1: If the current subscription is for event-driven updates, the C2C RI shall initiate the publication within 1 minute of a change in Gate Status (See Requirement SRS95) when the changed data item(s) are included within the original subscription request.	3.2.2.4.6	Lane Closure Gate	DetectorStatus(5.4.1.2.2.2.5) DMSHandler(5.4.1.1.2.8) DMSInventory(5.4.1.2.2.3.2) DMSMessageInventory(5.4.1.2.2.3.4) DMSStatus(5.4.1.2.2.3.5) ESSHandler(5.4.1.1.2.9) ESSInventory(5.4.1.2.2.4.1) ESSObservationMetadata(5.4.1.2.2.4.2) ESSObservationReport(5.4.1.2.2.4.3) ESSStatus(5.4.1.2.2.4.4) GateHandler(5.4.1.1.2.11) PublicationOCTag(5.3.1.2.33) RIEmulation(5.4.1.1.2.5) ApplicationLayerStandard(5.3.2.2.4) Entity Emulation Overview(6.2) Section 6.2 includes a description on how the messages (and data) in the dialogs are processed. The content is based on the message definitions included in TMDD Volume II. Unless the values are modified as part of the control request processing or a script initiated value change, the parameter values will be as specified in the Test Configuration at the start of a test. Note that the C2C RI's ability to satisfy this performance requirement will be impacted by the performance of the underlying selected application layer standard.

Requirement	SRS Section	Title	Design
SRS125: When acting as an Owner Center and TMDD Entity Emulation is selected, the C2C RI shall begin responding to requests for HAR information and control, per TMDD Volume 1 Section 3.4.2, after receipt from the External Center. SRS125.1: When the C2C RI receives a valid deviceCancelControlRequestMsg Message for a HAR device from an External Center as part of a dlDeviceCancelControlRequest dialog the C2C RI shall send a deviceControlResponseMsg response Message to the External Center. SRS125.2: When the C2C RI receives a valid deviceControlStatusRequestMsg Message for a HAR device from an External Center as part of a dlDeviceControlStatusRequest dialog the C2C RI shall send a deviceControlResponseMsg response Message to the External Center. SRS125.3: When the C2C RI receives a valid hARControlRequestMsg Message for a HAR device from an External Center as part of a dlHARControlRequest dialog the C2C RI shall send a deviceControlResponseMsg response Message to the External Center. SRS125.3.1: The C2C RI shall update the HAR Status information (See SRS95 and SRS96) when a valid control command has been completed. SRS125.4: When the C2C RI receives a valid deviceInformationRequestMsg Message for a HAR device from an External Center as part of a dlHARControlScheduleRequest dialog the C2C RI shall send a hARControlScheduleMsg response Message to the External Center. SRS125.5: When the C2C RI receives a valid deviceInformationRequestMsg	3.2.2.4.7	Highway Advisory Radio (HAR)	ESSHandler(5.4.1.1.2.9) ESSInventory(5.4.1.2.2.4.1) ESSObservationReport(5.4.1.2.2.4.3) ESSStatus(5.4.1.2.2.4.4) EventHandler(5.4.1.1.2.10) EventsActionLog(5.4.1.2.2.5.2) EventsIndex(5.4.1.2.2.5.3) EventUpdates(5.4.1.2.2.5.1) HARHandler(5.4.1.1.2.12) RequestResponseOCReceiveTag(5.3.1.2.43) RequestResponseOCReplyTag(5.3.1.2.44) RIEmulation(5.4.1.1.2.5) ApplicationLayerStandard(5.3.2.2.4) Entity Emulation Overview(6.2) Section 6.2 includes a description on how the messages (and data) in the dialogs are processed. The content is based on the message definitions included in TMDD Volume II. Unless the values are modified as part of the control request processing or a script initiated value change, the parameter values will be as specified in the Test Configuration at the start of a test. Note that the C2C RI's ability to satisfy this performance requirement will be impacted by the performance of the underlying selected application layer standard.

Requirement	SRS Section	Title	Design
Message for a HAR device from an External Center as part of a dlHARInventoryRequest dialog the C2C RI shall send a hARInventoryMsg response Message to the External Center. SRS125.6: When the C2C RI receives a valid deviceInformationRequestMsg Message for a HAR device from an External Center as part of a dlHARMessageInventoryRequest dialog the C2C RI shall send a hARMessageInventoryMsg response Message to the External Center. SRS125.7: When the C2C RI receives a valid devicePriorityQueueRequestMsg Message for a HAR device from an External Center as part of a dlHARPriorityQueueRequest dialog the C2C RI shall send a hARPriorityQueueMsg response Message to the External Center. SRS125.8: When the C2C RI receives a valid deviceInformationRequestMsg Message for a HAR device from an External Center as part of a dlHARStatusRequest dialog the C2C RI shall send a hARStatusMsg response Message to the External Center.			
SRS126: When acting as an Owner Center and TMDD Entity Emulation is selected, the C2C RI shall begin responding to a subscription request for HAR information, per TMDD Volume 1 Section 3.4.2, after receipt from the External Center. SRS126.1: When the C2C RI receives a valid deviceInformationRequestMsg Message for a HAR device from an External Center as part of a dlDeviceInformationSubscription dialog the C2C RI shall send a c2cMessageReceipt response Message to the External Center.	3.2.2.4.7	Highway Advisory Radio (HAR)	HARHandler(5.4.1.1.2.12) RIEmulation(5.4.1.1.2.5) SubscriptionOCReceiveTag(5.3.1.2.46) SubscriptionOCReplyTag(5.3.1.2.47) ApplicationLayerStandard(5.3.2.2.4) Entity Emulation Overview(6.2) Section 6.2 includes a description on how the messages (and data) in the dialogs are processed. The content is based on the message definitions included in TMDD Volume II. Unless the values are modified as part of the control request processing or a script initiated value change, the parameter values will be as specified in the Test Configuration at the start of a test.

Requirement	SRS Section	Title	Design
			Note that the C2C RI's ability to satisfy this performance requirement will be impacted by the performance of the underlying selected application layer standard.
SRS127: When acting as an Owner Center and TMDD Entity Emulation is selected, the C2C RI shall publish HAR information updates to a subscribing External Center. SRS127.1: The C2C RI shall publish a hARControlScheduleMsg Message as part of a dlHARControlScheduleUpdate dialog to an External Center in accordance with current subscriptions. SRS127.1.1: If the current subscription is for event-driven updates, the C2C RI shall initiate the publication within 1 minute of a change in HAR Control Schedule (See Requirement SRS95) when the changed data item(s) are included within the original subscription request. SRS127.2: The C2C RI shall publish a hARInventoryMsg Message as part of a dlHARInventoryUpdate dialog to an External Center in accordance with current subscriptions. SRS127.2.1: If the current subscription is for event-driven updates, the C2C RI shall initiate the publication within 1 minute of a change in HAR Inventory (See Requirement SRS95) when the changed data item(s) are included within the original subscription request. SRS127.3: The C2C RI shall publish a hARMessageInventoryMsg Message as part of a dlHARMessageInventoryUpdate dialog to an External Center with a subscription for the changed data. SRS127.3.1: If the current subscription is for event-driven updates, the C2C RI shall initiate the publication within 1 minute of a	3.2.2.4.7	Highway Advisory Radio (HAR)	EventHandler(5.4.1.1.2.10) EventsActionLog(5.4.1.2.2.5.2) EventsIndex(5.4.1.2.2.5.3) EventUpdates(5.4.1.2.2.5.1) HARHandler(5.4.1.1.2.12) PublicationOCTag(5.3.1.2.33) RIEmulation(5.4.1.1.2.5) ApplicationLayerStandard(5.3.2.2.4) Entity Emulation Overview(6.2) Section 6.2 includes a description on how the messages (and data) in the dialogs are processed. The content is based on the message definitions included in TMDD Volume II. Unless the values are modified as part of the control request processing or a script initiated value change, the parameter values will be as specified in the Test Configuration at the start of a test. Note that the C2C RI's ability to satisfy this performance requirement will be impacted by the performance of the underlying selected application layer standard.

Requirement	SRS Section	Title	Design
change in HAR Message Inventory (See Requirement SRS95) when the changed data item(s) are included within the original subscription request. SRS127.4: The C2C RI shall publish a hARStatusMsg Message as part of a dlHARStatusUpdate dialog to an External Center in accordance with current subscriptions. SRS127.4.1: If the current subscription is for event-driven updates, the C2C RI shall initiate the publication within 1 minute of a change in HAR Status (See Requirement SRS95) when the changed data item(s) are included within the original subscription request.			
SRS128: When acting as an Owner Center and TMDD Entity Emulation is selected, the C2C RI shall begin responding to requests for Lane Control Signal information and control, per TMDD Volume 1 Section 3.4.2, after receipt from the External Center. SRS128.1: When the C2C RI receives a valid deviceCancelControlRequestMsg Message for a Lane Control Signal device from an External Center as part of a dlDeviceCancelControlRequest dialog the C2C RI shall send a deviceControlResponseMsg response Message to the External Center. SRS128.2: When the C2C RI receives a valid deviceControlStatusRequestMsg Message for a Lane Control Signal device from an External Center as part of a dlDeviceControlStatusRequest dialog the C2C RI shall send a deviceControlResponseMsg response Message to the External Center. SRS128.3: When the C2C RI receives a valid lCSControlRequestMsg Message for a Lane Control Signal device from an External Center as part of a	3.2.2.4.8	Lane Control Signal	EventHandler(5.4.1.1.2.10) EventUpdates(5.4.1.2.2.5.1) GateControlSchedule(5.4.1.2.2.6.1) GateHandler(5.4.1.1.2.11) GateInventory(5.4.1.2.2.6.2) GateStatus(5.4.1.2.2.6.3) LaneControlSignalHandler(5.4.1.1.2.13) RequestResponseOCReceiveTag(5.3.1.2.43) RequestResponseOCReplyTag(5.3.1.2.44) RIEmulation(5.4.1.1.2.5) ApplicationLayerStandard(5.3.2.2.4) Entity Emulation Overview(6.2) Section 6.2 includes a description on how the messages (and data) in the dialogs are processed. The content is based on the message definitions included in TMDD Volume II. Unless the values are modified as part of the control request processing or a script initiated value change, the parameter values will be as specified in the Test Configuration at the start of a test. Note that the C2C RI's ability to satisfy this performance requirement will be impacted by the performance of the underlying selected application layer standard.

Requirement	SRS Section	Title	Design
dlLCSControlRequest dialog the C2C RI shall send a deviceControlResponseMsg response Message to the External Center. SRS128.3.1: The C2C RI shall update the Lane Control Signal Status information (See SRS95 and SRS96) when a valid control command has been completed. SRS128.4: When the C2C RI receives a valid deviceInformationRequestMsg Message for a Lane Control Signal device from an External Center as part of a dlLCSControlScheduleRequest dialog the C2C RI shall send a lCSControlScheduleMsg response Message to the External Center. SRS128.5: When the C2C RI receives a valid deviceInformationRequestMsg Message for a Lane Control Signal device from an External Center as part of a dlLCSInventoryRequest dialog the C2C RI shall send a lCSInventoryMsg response Message to the External Center. SRS128.6: When the C2C RI receives a valid deviceInformationRequestMsg Message for a Lane Control Signal device from an External Center as part of a dlLCSStatusRequest dialog the C2C RI shall send a lCSStatusMsg response Message to the External Center.			
SRS129: When acting as an Owner Center and TMDD Entity Emulation is selected, the C2C RI shall begin responding to a subscription request for Lane Control Signal information, per TMDD Volume 1 Section 3.4.2, after receipt from the External Center. SRS129.1: When the C2C RI receives a valid deviceInformationRequestMsg Message for a Lane Control Signal device from an External Center as part of a dlDeviceInformationSubscription dialog the C2C RI shall send a	3.2.2.4.8	Lane Control Signal	LaneControlSignalHandler(5.4.1.1.2.13) RIEmulation(5.4.1.1.2.5) SubscriptionOCReceiveTag(5.3.1.2.46) SubscriptionOCReplyTag(5.3.1.2.47) ApplicationLayerStandard(5.3.2.2.4) Entity Emulation Overview(6.2) Section 6.2 includes a description on how the messages (and data) in the dialogs are processed. The content is based on the message definitions included in TMDD Volume II.

Requirement	SRS Section	Title	Design
c2cMessageReceipt response Message to the External Center.			Unless the values are modified as part of the control request processing or a script initiated value change, the parameter values will be as specified in the Test Configuration at the start of a test. Note that the C2C RI's ability to satisfy this performance requirement will be impacted by the performance of the underlying selected application layer standard.
SRS130: When acting as an Owner Center and TMDD Entity Emulation is selected, the C2C RI shall publish Lane Control Signal information updates to a subscribing External Center. SRS130.1: The C2C RI shall publish a lCSControlScheduleMsg Message as part of a dlLCSControlScheduleUpdate dialog to an External Center in accordance with current subscriptions. SRS130.1.1: If the current subscription is for event-driven updates, the C2C RI shall initiate the publication within 1 minute of a change in Lane Control Signal Control Schedule (See Requirement SRS95) when the changed data item(s) are included within the original subscription request. SRS130.2: The C2C RI shall publish a lCSInventoryMsg Message as part of a dlLCSInventoryUpdate dialog to an External Center in accordance with current subscriptions. SRS130.2.1: If the current subscription is for event-driven updates, the C2C RI shall initiate the publication within 1 minute of a change in Lane Control Signal Inventory (See Requirement SRS95) when the changed data item(s) are included within the original subscription request. SRS130.3: The C2C RI shall publish a lCSStatusMsg Message as part of a dlLCSStatusUpdate dialog to an External Center in accordance with current subscriptions. SRS130.3.1: If the current subscription is for event-driven	3.2.2.4.8	Lane Control Signal	GateControlSchedule(5.4.1.2.2.6.1) GateHandler(5.4.1.1.2.11) GateInventory(5.4.1.2.2.6.2) GateStatus(5.4.1.2.2.6.3) LaneControlSignalHandler(5.4.1.1.2.13) PublicationOCTag(5.3.1.2.33) RIEmulation(5.4.1.1.2.5) ApplicationLayerStandard(5.3.2.2.4) Entity Emulation Overview(6.2) Section 6.2 includes a description on how the messages (and data) in the dialogs are processed. The content is based on the message definitions included in TMDD Volume II. Unless the values are modified as part of the control request processing or a script initiated value change, the parameter values will be as specified in the Test Configuration at the start of a test. Note that the C2C RI's ability to satisfy this performance requirement will be impacted by the performance of the underlying selected application layer standard.

Requirement	SRS Section	Title	Design
updates, the C2C RI shall initiate the publication within 1 minute of a change in Lane Control Signal Status (See Requirement SRS95) when the changed data item(s) are included within the original subscription request.			
SRS131: When acting as an Owner Center and TMDD Entity Emulation is selected, the C2C RI shall begin responding to requests for Ramp Meter information and control, per TMDD Volume 1 Section 3.4.2, after receipt from the External Center. SRS131.1: When the C2C RI receives a valid deviceCancelControlRequestMsg Message for a Ramp Meter device from an External Center as part of a dlDeviceCancelControlRequest dialog the C2C RI shall send a deviceControlResponseMsg response Message to the External Center. SRS131.2: When the C2C RI receives a valid deviceControlStatusRequestMsg Message for a Ramp Meter device from an External Center as part of a dlDeviceControlStatusRequest dialog the C2C RI shall send a deviceControlResponseMsg response Message to the External Center. SRS131.3: When the C2C RI receives a valid rampMeterControlRequestMsg Message for a Ramp Meter device from an External Center as part of a dlRampMeterControlRequest dialog the C2C RI shall send a deviceControlResponseMsg response Message to the External Center. SRS131.3.1: The C2C RI shall update the Ramp Meter Status information (See SRS95 and SRS96) when a valid control command has been completed. SRS131.4: When the C2C RI receives a valid	3.2.2.4.9	Ramp Meter	GateHandler(5.4.1.1.2.11) GateStatus(5.4.1.2.2.6.3) HARControlSchedule(5.4.1.2.2.7.1) HARHandler(5.4.1.1.2.12) HARInventory(5.4.1.2.2.7.2) HARMessageInventory(5.4.1.2.2.7.3) HARStatus(5.4.1.2.2.7.4) RampMeterHandler(5.4.1.1.2.14) RequestResponseOCReceiveTag(5.3.1.2.43) RequestResponseOCReplyTag(5.3.1.2.44) RIEmulation(5.4.1.1.2.5) ApplicationLayerStandard(5.3.2.2.4) Entity Emulation Overview(6.2) Section 6.2 includes a description on how the messages (and data) in the dialogs are processed. The content is based on the message definitions included in TMDD Volume II. Unless the values are modified as part of the control request processing or a script initiated value change, the parameter values will be as specified in the Test Configuration at the start of a test. Note that the C2C RI's ability to satisfy this performance requirement will be impacted by the performance of the underlying selected application layer standard.

Requirement	SRS Section	Title	Design
deviceInformationRequestMsg Message for a Ramp Meter device from an External Center as part of a dlRampMeterControlScheduleRequest dialog the C2C RI shall send a rampMeterControlScheduleMsg response Message to the External Center. SRS131.5: When the C2C RI receives a valid deviceInformationRequestMsg Message for a Ramp Meter device from an External Center as part of a dlRampMeterInventoryRequest dialog the C2C RI shall send a rampMeterInventoryMsg response Message to the External Center. SRS131.6: When the C2C RI receives a valid deviceInformationRequestMsg Message for a Ramp Meter device from an External Center as part of a dlRampMeterPlanInventoryRequest dialog the C2C RI shall send a rampMeterPlanInventoryMsg response Message to the External Center. SRS131.7: When the C2C RI receives a valid devicePriorityQueueRequestMsg Message for a Ramp Meter device from an External Center as part of a dlRampMeterPriorityQueueRequest dialog the C2C RI shall send a rampMeterPriorityQueueMsg response Message to the External Center. SRS131.8: When the C2C RI receives a valid deviceInformationRequestMsg Message for a Ramp Meter device from an External Center as part of a dlRampMeterStatusRequest dialog the C2C RI shall send a rampMeterStatusMsg response Message to the External Center.			

Requirement	SRS Section	Title	Design
SRS132: When acting as an Owner Center and TMDD Entity Emulation is selected, the C2C RI shall begin responding to a subscription request for Ramp Meter information, per TMDD Volume 1 Section 3.4.2, after receipt from the External Center. SRS132.1: When the C2C RI receives a valid deviceInformationRequestMsg Message for a Ramp Meter device from an External Center as part of a dlDeviceInformationSubscription dialog the C2C RI shall send a c2cMessageReceipt response Message to the External Center. SRS132.2: When the C2C RI receives a valid deviceInformationRequestMsg Message for a Ramp Meter device from an External Center as part of a dlRampMeterPlanInventorySubscription dialog the C2C RI shall send a c2cMessageReceipt response Message to the External Center.	3.2.2.4.9	Ramp Meter	RampMeterHandler(5.4.1.1.2.14) RIEmulation(5.4.1.1.2.5) SubscriptionOCReceiveTag(5.3.1.2.46) SubscriptionOCReplyTag(5.3.1.2.47) ApplicationLayerStandard(5.3.2.2.4) Entity Emulation Overview(6.2) Section 6.2 includes a description on how the messages (and data) in the dialogs are processed. The content is based on the message definitions included in TMDD Volume II. Unless the values are modified as part of the control request processing or a script initiated value change, the parameter values will be as specified in the Test Configuration at the start of a test. Note that the C2C RI's ability to satisfy this performance requirement will be impacted by the performance of the underlying selected application layer standard.
SRS133: When acting as an Owner Center and TMDD Entity Emulation is selected, the C2C RI shall publish Ramp Meter information updates to a subscribing External Center. SRS133.1: The C2C RI shall publish a rampMeterControlScheduleMsg Message as part of a dlRampMeterControlScheduleUpdate dialog to an External Center in accordance with current subscriptions. SRS133.1.1: If the current subscription is for event-driven updates, the C2C RI shall initiate the publication within 1 minute of a change in Ramp Meter Control Schedule (See Requirement SRS95) when the changed data item(s) are included within the original subscription request. SRS133.2: The C2C RI shall publish a rampMeterInventoryMsg Message as part of a dlRampMeterInventoryUpdate dialog to an External Center in accordance with current subscriptions.	3.2.2.4.9	Ramp Meter	HARControlSchedule(5.4.1.2.2.7.1) HARHandler(5.4.1.1.2.12) HARInventory(5.4.1.2.2.7.2) HARMessageInventory(5.4.1.2.2.7.3) HARStatus(5.4.1.2.2.7.4) PublicationOCTag(5.3.1.2.33) RampMeterHandler(5.4.1.1.2.14) RIEmulation(5.4.1.1.2.5) ApplicationLayerStandard(5.3.2.2.4) Entity Emulation Overview(6.2) Section 6.2 includes a description on how the messages (and data) in the dialogs are processed. The content is based on the message definitions included in TMDD Volume II. Unless the values are modified as part of the control request processing or a script initiated value change, the parameter values will be as specified in the Test Configuration at the start of a test. Note that the C2C RI's ability to satisfy this performance requirement will be impacted by the performance of the underlying selected

Requirement	SRS Section	Title	Design
SRS133.2.1: If the current subscription is for event-driven updates, the C2C RI shall initiate the publication within 1 minute of a change in Ramp Meter Inventory (See Requirement SRS95) when the changed data item(s) are included within the original subscription request. SRS133.3: The C2C RI shall publish a rampMeterPlanInventoryMsg Message as part of a dlRampMeterPlanInventoryUpdate dialog to an External Center in accordance with current subscriptions. SRS133.3.1: If the current subscription is for event-driven updates, the C2C RI shall initiate the publication within 1 minute of a change in Ramp Meter Plan Inventory (See Requirement SRS95) when the changed data item(s) are included within the original subscription request. SRS133.4: The C2C RI shall publish a rampMeterStatusMsg Message as part of a dlRampMeterStatusUpdate dialog to an External Center in accordance with current subscriptions. SRS133.4.1: If the current subscription is for event-driven updates, the C2C RI shall initiate the publication within 1 minute of a change in Ramp Meter Status (See Requirement SRS95) when the changed data item(s) are included within the original subscription request.			application layer standard.
SRS134: When acting as an Owner Center and TMDD Entity Emulation is selected, the C2C RI shall begin responding to requests for Signal Section information and control, per TMDD Volume 1 Section 3.4.2, after receipt from the External Center. SRS134.1: When the C2C RI receives a valid deviceCancelControlRequestMsg Message for a Signal Section device from an External Center as part of a dlDeviceCancelControlRequest	3.2.2.4.10	Signal Section	HARHandler(5.4.1.1.2.12) HARStatus(5.4.1.2.2.7.4) LaneControlSignalControlSchedule(5.4.1.2.2.8.1) LaneControlSignalHandler(5.4.1.1.2.13) LaneControlSignalInventory(5.4.1.2.2.8.2) LaneControlSignalStatus(5.4.1.2.2.8.3) RequestResponseECTag(5.3.1.2.42) RequestResponseOCReceiveTag(5.3.1.2.43) RIEmulation(5.4.1.1.2.5)

Requirement	SRS Section	Title	Design
dialog the C2C RI shall send a deviceControlResponseMsg response Message to the External Center. SRS134.2: When the C2C RI receives a valid sectionControlRequestMsg Message for a Signal Section device from an External Center as part of a dlSectionControlRequest dialog the C2C RI shall send a sectionControlResponseMsg response Message to the External Center. SRS134.2.1: The C2C RI shall update the Signal Section Status information (See SRS95 and SRS96) when a valid control command has been completed. SRS134.3: When the C2C RI receives a valid deviceInformationRequestMsg Message for a Signal Section device from an External Center as part of a dlSectionControlScheduleRequest dialog the C2C RI shall send a sectionControlScheduleMsg response Message to the External Center SRS134.4: When the C2C RI receives a valid sectionControlStatusRequestMsg Message for a Signal Section device from an External Center as part of a dlSectionControlStatusRequest dialog the C2C RI shall send a sectionControlResponseMsg response Message to the External Center. SRS134.5: When the C2C RI receives a valid devicePriorityQueueRequestMsg Message for a Signal Section device from an External Center as part of a dlSectionPriorityQueueRequest dialog the C2C RI shall send a sectionPriorityQueueMsg response Message to the External Center. SRS134.6: When the C2C RI receives a valid SectionSignalTimingPatternInventoryRequestMsg Message for a Signal Section device from an External Center as part of a			SignalSectionHandler(5.4.1.1.2.15) ApplicationLayerStandard(5.3.2.2.4) Entity Emulation Overview(6.2) Section 6.2 includes a description on how the messages (and data) in the dialogs are processed. The content is based on the message definitions included in TMDD Volume II. Unless the values are modified as part of the control request processing or a script initiated value change, the parameter values will be as specified in the Test Configuration at the start of a test. Note that the C2C RI's ability to satisfy this performance requirement will be impacted by the performance of the underlying selected application layer standard.

Requirement	SRS Section	Title	Design
dlSectionSignalTimingPatternInventoryRequest dialog the C2C RI shall send a SectionSignalTimingPatternInventoryMsg response Message to the External Center. SRS134.7: When the C2C RI receives a valid deviceInformationRequestMsg Message for a Signal Section device from an External Center as part of a dlSectionStatusRequest dialog the C2C RI shall send a sectionStatusMsg response Message to the External Center.			
SRS135: When acting as an Owner Center and TMDD Entity Emulation is selected, the C2C RI shall begin responding to a subscription request for Signal Section information, per TMDD Volume 1 Section 3.4.2, after receipt from the External Center. SRS135.1: When the C2C RI receives a valid deviceInformationRequestMsg Message for a Signal Section device from an External Center as part of a dlDeviceInformationSubscription dialog the C2C RI shall send a c2cMessageReceipt response Message to the External Center. SRS135.2: When the C2C RI receives a valid SectionSignalTimingPatternInventoryRequestMsg Message for a Signal Section device from an External Center as part of a dlSectionSignalTimingPatternInventorySubscription dialog the C2C RI shall send a c2cMessageReceipt response Message to the External Center.	3.2.2.4.10	Signal Section	LaneControlSignalControlSchedule(5.4.1.2.2.8.1) LaneControlSignalHandler(5.4.1.1.2.13) LaneControlSignalInventory(5.4.1.2.2.8.2) LaneControlSignalStatus(5.4.1.2.2.8.3) RIEmulation(5.4.1.1.2.5) SignalSectionHandler(5.4.1.1.2.15) SubscriptionOCReceiveTag(5.3.1.2.46) SubscriptionOCReplyTag(5.3.1.2.47) ApplicationLayerStandard(5.3.2.2.4) Entity Emulation Overview(6.2) Section 6.2 includes a description on how the messages (and data) in the dialogs are processed. The content is based on the message definitions included in TMDD Volume II. Unless the values are modified as part of the control request processing or a script initiated value change, the parameter values will be as specified in the Test Configuration at the start of a test. Note that the C2C RI's ability to satisfy this performance requirement will be impacted by the performance of the underlying selected application layer standard.

Requirement	SRS Section	Title	Design
SRS136: When acting as an Owner Center and TMDD Entity Emulation is selected, the C2C RI shall publish Signal Section information updates to a subscribing External Center. SRS136.1: The C2C RI shall publish a sectionControlScheduleMsg Message as part of a dlSectionControlScheduleUpdate dialog to an External Center in accordance with current subscriptions. SRS136.1.1: If the current subscription is for event-driven updates, the C2C RI shall initiate the publication within 1 minute of a change in Section Control Schedule (See Requirement SRS95) when the changed data item(s) are included within the original subscription request. SRS136.2: The C2C RI shall publish a SectionSignalTimingPatternInventoryMsg Message as part of a dlSectionSignalTimingPatternInventoryUpdate dialog to an External Center in accordance with current subscriptions. SRS136.2.1: If the current subscription is for event-driven updates, the C2C RI shall initiate the publication within 1 minute of a change in Signal Section Timing Pattern Inventory (See Requirement SRS95) when the changed data item(s) are included within the original subscription request. SRS136.3: The C2C RI shall publish a sectionStatusMsg Message as part of a dlSectionStatusUpdate dialog to an External Center in accordance with current subscriptions. SRS136.3.1: If the current subscription is for event-driven updates, the C2C RI shall initiate the publication within 1 minute of a change in Signal Section Status (See Requirement SRS95) when the changed data item(s) are included within the original subscription request.	3.2.2.4.10	Signal Section	LaneControlSignalControlSchedule(5.4.1.2.2.8.1) LaneControlSignalHandler(5.4.1.1.2.13) LaneControlSignalInventory(5.4.1.2.2.8.2) LaneControlSignalStatus(5.4.1.2.2.8.3) PublicationOCTag(5.3.1.2.33) RIEmulation(5.4.1.1.2.5) SignalSectionHandler(5.4.1.1.2.15) TrafficControllerHandler(5.4.1.1.2.16) IntersectionSignalControllerControlSchedule(5.4.1.2.2.11.1) IntersectionSignalControllerInventory(5.4.1.2.2.11.2) IntersectionSignalControllerStatus(5.4.1.2.2.11.3) IntersectionSignalControllerTimingPatternInventory(5.4.1.2.2.11.4) ApplicationLayerStandard(5.3.2.2.4) Entity Emulation Overview(6.2) Section 6.2 includes a description on how the messages (and data) in the dialogs are processed. The content is based on the message definitions included in TMDD Volume II. Unless the values are modified as part of the control request processing or a script initiated value change, the parameter values will be as specified in the Test Configuration at the start of a test. Note that the C2C RI's ability to satisfy this performance requirement will be impacted by the performance of the underlying selected application layer standard.

Requirement	SRS Section	Title	Design
SRS137: When acting as an Owner Center and TMDD Entity Emulation is selected, the C2C RI shall begin responding to requests for Intersection Signal Controller information and control, per TMDD Volume 1 Section 3.4.2, after receipt from the External Center. SRS137.1: When the C2C RI receives a valid deviceCancelControlRequestMsg Message for an Intersection Signal Controller device from an External Center as part of a dlDeviceCancelControlRequest dialog the C2C RI shall send a deviceControlResponseMsg response Message to the External Center. SRS137.2: When the C2C RI receives a valid deviceControlStatusRequestMsg Message for an Intersection Signal Controller device from an External Center as part of a dlDeviceControlStatusRequest dialog the C2C RI shall send a deviceControlResponseMsg response Message to the External Center. SRS137.3: When the C2C RI receives a valid intersectionSignalControlRequestMsg Message for an Intersection Signal Controller device from an External Center as part of a dlIntersectionSignalControlRequest dialog the C2C RI shall send a intersectionSignalControlResponse Msg response Message to the External Center. SRS137.3.1: The C2C RI shall update the Intersection Signal Controller Status information (See SRS95 and SRS96) when a valid control command has been completed. SRS137.4: When the C2C RI receives a valid deviceInformationRequestMsg Message for an Intersection Signal Controller device from an External Center as part of a dlIntersectionSignalControlSchedule	3.2.2.4.11	Intersection Signal Controller	RequestResponseOCReceiveTag(5.3.1.2.43) RequestResponseOCReplyTag(5.3.1.2.44) RIEmulation(5.4.1.1.2.5) TrafficControllerHandler(5.4.1.1.2.16) IntersectionSignalControllerControlSchedule(5.4.1.2.2.11.1) IntersectionSignalControllerInventory(5.4.1.2.2.11.2) IntersectionSignalControllerStatus(5.4.1.2.2.11.3) IntersectionSignalControllerTimingPatternInventory(5.4.1.2.2.11.4) VideoSwitchHandler(5.4.1.1.2.17) VideoSwitchInventory(5.4.1.2.2.12.1) VideoSwitchStatus(5.4.1.2.2.12.2) ApplicationLayerStandard(5.3.2.2.4) Entity Emulation Overview(6.2) Section 6.2 includes a description on how the messages (and data) in the dialogs are processed. The content is based on the message definitions included in TMDD Volume II. Unless the values are modified as part of the control request processing or a script initiated value change, the parameter values will be as specified in the Test Configuration at the start of a test. Note that the C2C RI's ability to satisfy this performance requirement will be impacted by the performance of the underlying selected application layer standard.

Requirement	SRS Section	Title	Design
Request dialog the C2C RI shall send a intersectionSignalControlScheduleMsg response Message to the External Center. SRS137.5: When the C2C RI receives a valid deviceInformationRequestMsg Message for an Intersection Signal Controller device from an External Center as part of a dlIntersectionSignalInventoryRequest dialog the C2C RI shall send a intersectionSignalInventoryMsg response Message to the External Center. SRS137.6: When the C2C RI receives a valid devicePriorityQueueRequestMsg Message for an Intersection Signal Controller device from an External Center as part of a dlIntersectionSignalPriorityQueueRequest dialog the C2C RI shall send a intersectionSignalPriorityQueueMsg response Message to the External Center. SRS137.7: When the C2C RI receives a valid deviceInformationRequestMsg Message for an Intersection Signal Controller device from an External Center as part of a dlIntersectionSignalStatusRequest dialog the C2C RI shall send a intersectionSignalStatusMsg response Message to the External Center. SRS137.8: When the C2C RI receives a valid intersectionSignalTimingPatternInventoryRequestMsg Message for an Intersection Signal Controller device from an External Center as part of a dlIntersectionSignalTimingPatternInventoryRequest dialog the C2C RI shall send a intersectionSignalTimingPatternInventoryMsg response Message to the External Center.			

Requirement	SRS Section	Title	Design
SRS138: When acting as an Owner Center and TMDD Entity Emulation is selected, the C2C RI shall begin responding to a subscription request for Intersection Signal Controller information, per TMDD Volume 1 Section 3.4.2, after receipt from the External Center. SRS138.1: When the C2C RI receives a valid deviceInformationRequestMsg Message for an Intersection Signal Controller device from an External Center as part of a dlDeviceInformationSubscription dialog the C2C RI shall send a c2cMessageReceipt response Message to the External Center. SRS138.2: When the C2C RI receives a valid intersectionSignalTimingPatternInventoryRequestMsg Message for an Intersection Signal Controller device from an External Center as part of a dlIntersectionSignalTimingPatternInventorySubscription dialog the C2C RI shall send a c2cMessageReceipt response Message to the External Center.	3.2.2.4.12	Video Switch	RIEmulation(5.4.1.1.2.5) SubscriptionOCReceiveTag(5.3.1.2.46) SubscriptionOCReplyTag(5.3.1.2.47) TrafficControllerHandler(5.4.1.1.2.16) VideoSwitchHandler(5.4.1.1.2.17) VideoSwitchStatus(5.4.1.2.2.12.2) ApplicationLayerStandard(5.3.2.2.4) Entity Emulation Overview(6.2) Section 6.2 includes a description on how the messages (and data) in the dialogs are processed. The content is based on the message definitions included in TMDD Volume II. Unless the values are modified as part of the control request processing or a script initiated value change, the parameter values will be as specified in the Test Configuration at the start of a test. Note that the C2C RI's ability to satisfy this performance requirement will be impacted by the performance of the underlying selected application layer standard.
SRS139: When acting as an Owner Center and TMDD Entity Emulation is selected, the C2C RI shall publish Intersection Signal Controller information updates to a subscribing External Center. SRS139.1: The C2C RI shall publish a intersectionSignalControlScheduleMsg Message as part of a dlIntersectionSignalControlScheduleUpdate dialog to an External Center in accordance with current subscriptions. SRS139.1.1: If the current subscription is for event-driven updates, the C2C RI shall initiate the publication within 1 minute of a change in Intersection Signal Controller Control Schedule (See Requirement SRS95) when the changed data item(s) are included within the original subscription request. SRS139.2: The C2C RI shall publish	3.2.2.4.11	Intersection Signal Controller	PublicationOCTag(5.3.1.2.33) RIEmulation(5.4.1.1.2.5) TrafficControllerHandler(5.4.1.1.2.16) VideoSwitchHandler(5.4.1.1.2.17) VideoSwitchInventory(5.4.1.2.2.12.1) VideoSwitchStatus(5.4.1.2.2.12.2) ApplicationLayerStandard(5.3.2.2.4) Entity Emulation Overview(6.2) Section 6.2 includes a description on how the messages (and data) in the dialogs are processed. The content is based on the message definitions included in TMDD Volume II. Unless the values are modified as part of the control request processing or a script initiated value change, the parameter values will be as specified in the Test Configuration at the start of a test. Note that the C2C RI's ability to satisfy this performance requirement

Requirement	SRS Section	Title	Design
a intersectionSignalInventoryMsg Message as part of a dlIntersectionSignalInventoryUpdate dialog to an External Center in accordance with current subscriptions. SRS139.2.1: If the current subscription is for event-driven updates, the C2C RI shall initiate the publication within 1 minute of a change in Intersection Signal Controller Inventory (See Requirement SRS95) when the changed data item(s) are included within the original subscription request. SRS139.3: The C2C RI shall publish a intersectionSignalStatusMsg Message as part of a dlIntersectionSignalStatusUpdate dialog to an External Center in accordance with current subscriptions. SRS139.3.1: If the current subscription is for event-driven updates, the C2C RI shall initiate the publication within 1 minute of a change in Intersection Signal Controller Status (See Requirement SRS95) when the changed data item(s) are included within the original subscription request. SRS139.4: The C2C RI shall publish a intersectionSignalTimingPatternInventoryMsg Message as part of a dlIntersectionSignalTimingPatternInventoryUpdate dialog to an External Center in accordance with current subscriptions. SRS139.4.1: If the current subscription is for event-driven updates, the C2C RI shall initiate the publication within 1 minute of a change in Intersection Signal Controller Timing Pattern Inventory (See Requirement SRS95) when the changed data item(s) are included within the original subscription request.			will be impacted by the performance of the underlying selected application layer standard.

Requirement	SRS Section	Title	Design
SRS140: When acting as an Owner Center and TMDD Entity Emulation is selected, the C2C RI shall begin responding to requests for Video Switch information and control, per TMDD Volume 1 Section 3.4.2, after receipt from the External Center. SRS140.1: When the C2C RI receives a valid deviceCancelControlRequestMsg Message for a Video Switch device from an External Center as part of a dlDeviceCancelControlRequest dialog the C2C RI shall send a deviceControlResponseMsg response Message to the External Center. SRS140.2: When the C2C RI receives a valid deviceControlStatusRequestMsg Message for a Video Switch device from an External Center as part of a dlDeviceControlStatusRequest dialog the C2C RI shall send a deviceControlResponseMsg response Message to the External Center. SRS140.3: When the C2C RI receives a valid videoSwitchControlRequestMsg Message for a Video Switch device from an External Center as part of a dlVideoSwitchControlRequest dialog the C2C RI shall send a deviceControlResponseMsg response Message to the External Center. SRS140.3.1: The C2C RI shall update the Video Switch Status information (See SRS95 and SRS96) when a valid control command has been completed. SRS140.4: When the C2C RI receives a valid deviceInformationRequestMsg Message for a Video Switch device from an External Center as part of a dlVideoSwitchInventoryRequest dialog the C2C RI shall send a videoSwitchInventoryMsg response Message to the External Center. SRS140.5: When the C2C RI receives a valid	3.2.2.4.12	Video Switch	DMSHandler(5.4.1.1.2.8) DMSInventory(5.4.1.2.2.3.2) DMSMessageInventory(5.4.1.2.2.3.4) DMSStatus(5.4.1.2.2.3.5) RequestResponseOCReceiveTag(5.3.1.2.43) RequestResponseOCReplyTag(5.3.1.2.44) RIEmulation(5.4.1.1.2.5) VideoSwitchHandler(5.4.1.1.2.17) ApplicationLayerStandard(5.3.2.2.4) Entity Emulation Overview(6.2) Section 6.2 includes a description on how the messages (and data) in the dialogs are processed. The content is based on the message definitions included in TMDD Volume II. Unless the values are modified as part of the control request processing or a script initiated value change, the parameter values will be as specified in the Test Configuration at the start of a test. Note that the C2C RI's ability to satisfy this performance requirement will be impacted by the performance of the underlying selected application layer standard.

Requirement	SRS Section	Title	Design
deviceInformationRequestMsg Message for a Video Switch device from an External Center as part of a dlVideoSwitchStatusRequest dialog the C2C RI shall send a videoSwitchStatusMsg response Message to the External Center.			
SRS141: When acting as an Owner Center and TMDD Entity Emulation is selected, the C2C RI shall begin responding to a subscription request for Video Switch information, per TMDD Volume 1 Section 3.4.2, after receipt from the External Center. SRS141.1: When the C2C RI receives a valid deviceInformationRequestMsg Message for a Video Switch device from an External Center as part of a dlDeviceInformationSubscription dialog the C2C RI shall send a c2cMessageReceipt response Message to the External Center.	3.2.2.4.12	Video Switch	DMSHandler(5.4.1.1.2.8) DMSMessageInventory(5.4.1.2.2.3.4) RIEmulation(5.4.1.1.2.5) SubscriptionOCReceiveTag(5.3.1.2.46) SubscriptionOCReplyTag(5.3.1.2.47) VideoSwitchHandler(5.4.1.1.2.17) ApplicationLayerStandard(5.3.2.2.4) Entity Emulation Overview(6.2) Section 6.2 includes a description on how the messages (and data) in the dialogs are processed. The content is based on the message definitions included in TMDD Volume II. Unless the values are modified as part of the control request processing or a script initiated value change, the parameter values will be as specified in the Test Configuration at the start of a test. Note that the C2C RI's ability to satisfy this performance requirement will be impacted by the performance of the underlying selected application layer standard.
SRS142: When acting as an Owner Center and TMDD Entity Emulation is selected, the C2C RI shall publish Video Switch information updates to a subscribing External Center. SRS142.1: The C2C RI shall publish a videoSwitchInventoryMsg Message as part of a dlVideoSwitchInventoryUpdate dialog to an External Center in accordance with current subscriptions. SRS142.1.1: If the current subscription is for event-driven updates, the C2C RI shall initiate the publication within 1 minute of a	3.2.2.4.12	Video Switch	DMSHandler(5.4.1.1.2.8) DMSStatus(5.4.1.2.2.3.5) ESSHandler(5.4.1.1.2.9) ESSInventory(5.4.1.2.2.4.1) ESSStatus(5.4.1.2.2.4.4) PublicationOCTag(5.3.1.2.33) RIEmulation(5.4.1.1.2.5) VideoSwitchHandler(5.4.1.1.2.17) ApplicationLayerStandard(5.3.2.2.4) Entity Emulation Overview(6.2) Section 6.2 includes a description on how the messages (and data) in the dialogs are processed. The content is based on the message definitions

Requirement	SRS Section	Title	Design
change in Video Switch Inventory (See Requirement SRS95) when the changed data item(s) are included within the original subscription request. SRS142.2: The C2C RI shall publish a videoSwitchStatusMsg Message as part of a dlVideoSwitchStatusUpdate dialog to an External Center in accordance with current subscriptions. SRS142.2.1: If the current subscription is for event-driven updates, the C2C RI shall initiate the publication within 1 minute of a change in Video Switch Status (See Requirement SRS95) when the changed data item(s) are included within the original subscription request.			included in TMDD Volume II. Unless the values are modified as part of the control request processing or a script initiated value change, the parameter values will be as specified in the Test Configuration at the start of a test. Note that the C2C RI's ability to satisfy this performance requirement will be impacted by the performance of the underlying selected application layer standard.
SRS143: The RI shall provide the Test Developer with an option to select parameters that will be used for Entity Emulation during a test. SRS143.1: The RI shall provide an option for the Test Developer to choose whether the Entity Emulation data will be initialized prior to the execution of each selected Test Case. SRS143.2: The RI shall provide an option for the Test Developer to specify Entity Emulation data values.	3.2.1.1.2	Edit and Store a Test Configuration	Section 6.2 includes a description on how the messages (and data) in the dialogs are processed. The content is based on the message definitions included in TMDD Volume II. Unless the values are modified as part of the control request processing or a script initiated value change, the parameter values will be as specified in the Test Configuration at the start of a test. Note that the C2C RI's ability to satisfy this performance requirement will be impacted by the performance of the underlying selected application layer standard.
SRS144: When acting as an Owner Center and TMDD Entity Emulation is selected, the C2C RI shall send an error response message, per TMDD Volume 1 Section 3.4.2, after receipt of an invalid message from the External Center.	3.2.2.4	Entity Emulation Actions	RIEmulation(5.4.1.1.2.5) Entity Emulation Overview(6.2) Section 6.2 includes a description of dialog handling. As part of this description, it indicates the conditions when an error response message will be generated.

Requirement	SRS Section	Title	Design
SRS145: The C2C RI TMDD entity emulation shall support one-time, periodic and on-change publications as specified in NTCIP 2306 v1.69.	3.2.2.4	Entity Emulation Actions	Publication(5.8.1.2.5) PublicationOCTag(5.3.1.2.33) RIEmulation(5.4.1.1.2.5) SubPubController(5.8.1.2.6) Subscription(5.8.1.2.7) SubscriptionOCReceiveTag(5.3.1.2.46) SubscriptionOCReplyTag(5.3.1.2.47) TMDDv303SubPubMapping(5.8.1.2.12) Entity Emulation Overview(6.2) Section 6.2 includes an additional description of subscription and publication handling.
SRS146: When acting as an Owner Center and TMDD Entity Emulation is selected, the C2C RI shall publish DMS information updates to a subscribing External Center. SRS146.1: The C2C RI shall publish a dMSInventoryMsg Message as part of a dlDMSInventoryUpdate dialog to an External Center in accordance with current subscriptions. SRS146.1.1: If the current subscription is for event-driven updates, the C2C RI shall initiate the publication within 1 minute of a change in DMS Inventory (See Requirement SRS95) when the changed data item(s) are included within the original subscription request. SRS146.2: The C2C RI shall publish a dMSMessageInventoryMsg Message as part of a dlDMSMessageInventoryUpdate dialog to an External Center in accordance with current subscriptions. SRS146.2.1: If the current subscription is for event-driven updates, the C2C RI shall initiate the publication within 1 minute of a change in DMS Message Inventory (See Requirement SRS95) when the changed data item(s) are included within the original subscription request. SRS146.3: The C2C RI shall publish a dMSStatusMsg Message as part of	3.2.2.4.3	Dynamic Message Sign (DMS)	DMSHandler(5.4.1.1.2.8) RIEmulation(5.4.1.1.2.5) Entity Emulation Overview(6.2) Section 6.2 includes a description on how the messages (and data) in the dialogs are processed. The content is based on the message definitions included in TMDD Volume II. Unless the values are modified as part of the control request processing or a script initiated value change, the parameter values will be as specified in the Test Configuration at the start of a test. Note that the C2C RI's ability to satisfy this performance requirement will be impacted by the performance of the underlying selected application layer standard.

Requirement	SRS Section	Title	Design
a dlDMSStatusUpdate dialog to an External Center in accordance with current subscriptions. SRS146.3.1: If the current subscription is for event-driven updates, the C2C RI shall initiate the publication within 1 minute of a change in DMS Status (See Requirement SRS95) when the changed data item(s) are included within the original subscription request.			
SRS147: When acting as an Owner Center and TMDD Entity Emulation is selected, the C2C RI shall publish ESS information updates to a subscribing External Center. SRS147.1: When the C2C RI receives a valid deviceInformationRequestMsg Message for a ESS device from an External Center as part of a dlDeviceInformationSubscription dialog the C2C RI shall send a c2cMessageReceipt response Message to the External Center.	3.2.2.4.4	Environmental Sensor Station (ESS)	ESSHandler(5.4.1.1.2.9) RIEmulation(5.4.1.1.2.5) Entity Emulation Overview(6.2) Section 6.2 includes a description on how the messages (and data) in the dialogs are processed. The content is based on the message definitions included in TMDD Volume II. Unless the values are modified as part of the control request processing or a script initiated value change, the parameter values will be as specified in the Test Configuration at the start of a test. Note that the C2C RI's ability to satisfy this performance requirement will be impacted by the performance of the underlying selected application layer standard.
SRS148: The C2C RI TMDD Entity Emulation shall support the publication (one-time, periodic and event-driven) of entity information for subscriptions as specified in TMDDv303c Volume II Section 2.3.4.	3.2.2.4	Entity Emulation Actions	Publication(5.8.1.2.5) PublicationOCTag(5.3.1.2.33) RIEmulation(5.4.1.1.2.5) SubPubController(5.8.1.2.6) Subscription(5.8.1.2.7) TMDDv303SubPubMapping(5.8.1.2.12) Entity Emulation Overview(6.2) Section 6.2 includes an additional description of subscription and publication handling.
SRS149: When acting as an owner center and TMDD entity emulation is selected, the C2C RI shall begin responding to requests for Center Active Verification, per TMDD Volume 1 Section 3.4.2, after receipt from the external center. SRS149.1: When the C2C RI receives a valid	3.2.2.4.13	Center Active Verification	RequestResponseOCReceiveTag(5.3.1.2.43) RequestResponseOCReplyTag(5.3.1.2.44) RIEmulation(5.4.1.1.2.5) CenterActiveVerificationHandler(5.4.1.1.2.1.2.1) CenterActiveVerification(5.4.1.1.2.1

Requirement	SRS Section	Title	Design
centerActiveVerificationRequestMsg message for a Center Active Verification from an external center as part of a dlCenterActiveVerification dialog, the C2C RI shall send a centerActiveVerificationResponseMsg response message to the external center.			.2.2) ApplicationLayerStandard(5.3.2.2.4) Entity Emulation Overview(6.2) Section 6.2 includes a description on how the messages (and data) in the dialogs are processed. The content is based on the message definitions included in TMDD Volume II. Unless the values are modified as part of the control request processing or a script initiated value change, the parameter values will be as specified in the Test Configuration at the start of a test. Note that the C2C RI's ability to satisfy this performance requirement will be impacted by the performance of the underlying selected application layer standard.
SRS150: When acting as an owner center and TMDD entity emulation is selected, the C2C RI shall begin responding to a subscription request for Center Active Verification, per TMDD Volume 1 Section 3.4.2, after receipt from the external center. SRS150.1: When the C2C RI receives a valid centerActiveVerificationRequestMsg message for a Center Active Verification from an external center as part of a dlCenterActiveVerificationSubscription dialog, the C2C RI shall send a c2cMessageReceipt response message to the external center.	3.2.2.4.13	Center Active Verification	CenterActiveVerificationHandler(5.4.1.1.2.1.2.1) CenterActiveVerification(5.4.1.1.2.1.2.2) RIEmulation(5.4.1.1.2.5) SubscriptionOCReceiveTag(5.3.1.2.46) SubscriptionOCReplyTag(5.3.1.2.47) ApplicationLayerStandard(5.3.2.2.4) Entity Emulation Overview(6.2) Section 6.2 includes a description on how the messages (and data) in the dialogs are processed. The content is based on the message definitions included in TMDD Volume II. Unless the values are modified as part of the control request processing or a script initiated value change, the parameter values will be as specified in the Test Configuration at the start of a test. Note that the C2C RI's ability to satisfy this performance requirement will be impacted by the performance of the underlying selected application layer standard.

Requirement	SRS Section	Title	Design
SRS151: When acting as an owner center and TMDD entity emulation is selected, the C2C RI shall publish Center Active Verification updates to a subscribing external center. SRS151.1: The C2C RI shall publish a centerActiveVerificationMsg message as part of a dlCenterActiveVerificationUpdate dialog to an external center in accordance with current subscriptions. SRS151.1.1: If the current subscription is for event-driven updates, the C2C RI shall initiate the publication within 1 minute of a change in Center Active Verification status (see Requirement SRS95).	3.2.2.4.13	Center Active Verification	CenterActiveVerificationHandler(5.4.1.1.2.1.2.1) RIEmulation(5.4.1.1.2.5) Entity Emulation Overview(6.2) Section 6.2 includes a description on how the messages (and data) in the dialogs are processed. The content is based on the message definitions included in TMDD Volume II. Unless the values are modified as part of the control request processing or a script initiated value change, the parameter values will be as specified in the Test Configuration at the start of a test. Note that the C2C RI's ability to satisfy this performance requirement will be impacted by the performance of the underlying selected application layer standard.
SRS152: When acting as an owner center and TMDD entity emulation is selected, the C2C RI shall begin responding to requests for Link information, per TMDD Volume 1 Section 3.4.2, after receipt from the external center. SRS152.1: When the C2C RI receives a valid trafficNetworkInformationRequestMsg message for a Link from an external center as part of a dlLinkInventoryRequest dialog, the C2C RI shall send a linkInventoryMsg response message to the external center. SRS152.2: When the C2C RI receives a valid trafficNetworkInformationRequestMsg message for a Link from an external center as part of a dlLinkStatusRequest dialog, the C2C RI shall send a linkStatusMsg response message to the external center.	3.2.2.4.14	Link (inventory, status)	RequestResponseOCReceiveTag(5.3.1.2.43) RequestResponseOCReplyTag(5.3.1.2.44) RIEmulation(5.4.1.1.2.5) LinkHandler(5.4.1.1.2.3.2.1) LinkStatus(5.4.1.1.2.3.2.3) LinkInventory(5.4.1.1.2.3.2.2) ApplicationLayerStandard(5.3.2.2.4) Entity Emulation Overview(6.2) Section 6.2 includes a description on how the messages (and data) in the dialogs are processed. The content is based on the message definitions included in TMDD Volume II. Unless the values are modified as part of the control request processing or a script initiated value change, the parameter values will be as specified in the Test Configuration at the start of a test. Note that the C2C RI's ability to satisfy this performance requirement will be impacted by the performance of the underlying selected application layer standard.

Requirement	SRS Section	Title	Design
SRS153: When acting as an owner center and TMDD entity emulation is selected, the C2C RI shall begin responding to a subscription request for Link information, per TMDD Volume 1 Section 3.4.2, after receipt from the external center. SRS153.1: When the C2C RI receives a valid trafficNetworkInformationRequestMsg message for a Link from an external center as part of a dlTrafficNetworkInformationSubscription dialog, the C2C RI shall send a c2cMessageReceipt response message to the external center.	3.2.2.4.14	Link (inventory, status)	LinkHandler(5.4.1.1.2.3.2.1) RIEmulation(5.4.1.1.2.5) SubscriptionOCReceiveTag(5.3.1.2.46) SubscriptionOCReplyTag(5.3.1.2.47) ApplicationLayerStandard(5.3.2.2.4) Entity Emulation Overview(6.2) Section 6.2 includes a description on how the messages (and data) in the dialogs are processed. The content is based on the message definitions included in TMDD Volume II. Unless the values are modified as part of the control request processing or a script initiated value change, the parameter values will be as specified in the Test Configuration at the start of a test. Note that the C2C RI's ability to satisfy this performance requirement will be impacted by the performance of the underlying selected application layer standard.
SRS154: When acting as an owner center and TMDD entity emulation is selected, the C2C RI shall publish link information updates to a subscribing external center. SRS154.1: The C2C RI shall publish a linkInventoryMsg message as part of a dlLinkInventoryUpdate dialog to an external center in accordance with current subscriptions. SRS154.1.1: If the current subscription is for event-driven updates, the C2C RI shall initiate the publication within 1 minute of a change in Link Inventory (see Requirement SRS95) when the changed data item(s) are included within the original subscription request. SRS154.2: The C2C RI shall publish a linkStatusMsg message as part of a dlLinkStatusUpdate dialog to an external center in accordance with current subscriptions. SRS154.2.1: If the current subscription is for event-driven updates, the C2C RI shall initiate the	3.2.2.4.14	Link (inventory, status)	LinkHandler(5.4.1.1.2.3.2.1) LinkInventory(5.4.1.1.2.3.2.2) LinkStatus(5.4.1.1.2.3.2.3) RIEmulation(5.4.1.1.2.5) Entity Emulation Overview(6.2) Section 6.2 includes a description on how the messages (and data) in the dialogs are processed. The content is based on the message definitions included in TMDD Volume II. Unless the values are modified as part of the control request processing or a script initiated value change, the parameter values will be as specified in the Test Configuration at the start of a test. Note that the C2C RI's ability to satisfy this performance requirement will be impacted by the performance of the underlying selected application layer standard.

Requirement	SRS Section	Title	Design
publication within 1 minute of a change in Link Status (see Requirement SRS95) when the changed data item(s) are included within the original subscription request.			
SRS155: When acting as an owner center and TMDD entity emulation is selected, the C2C RI shall begin responding to requests for Node information, per TMDD Volume 1 Section 3.4.2, after receipt from the external center. SRS155.1: When the C2C RI receives a valid trafficNetworkInformationRequestMsg message for a Node from an external center as part of a dlNodeInventoryRequest dialog, the C2C RI shall send a nodeInventoryMsg response message to the external center. SRS155.2: When the C2C RI receives a valid trafficNetworkInformationRequestMsg message for a Node from an external center as part of a dlNodeStatusRequest dialog, the C2C RI shall send a nodeStatusMsg response message to the external center.	3.2.2.4.15	Node (inventory, status)	RequestResponseOCReceiveTag(5.3.1.2.43) RequestResponseOCReplyTag(5.3.1.2.44) RIEmulation(5.4.1.1.2.5) NodeHandler(5.4.1.1.2.2.2.1) NodeInventory(5.4.1.1.2.2.2.2) NodeStatus(5.4.1.1.2.2.2.3) ApplicationLayerStandard(5.3.2.2.4) Entity Emulation Overview(6.2) Section 6.2 includes a description on how the messages (and data) in the dialogs are processed. The content is based on the message definitions included in TMDD Volume II. Unless the values are modified as part of the control request processing or a script initiated value change, the parameter values will be as specified in the Test Configuration at the start of a test. Note that the C2C RI's ability to satisfy this performance requirement will be impacted by the performance of the underlying selected application layer standard.
SRS156: When acting as an owner center and TMDD entity emulation is selected, the C2C RI shall begin responding to a subscription request for Node information, per TMDD Volume 1 Section 3.4.2, after receipt from the external center. SRS156.1: When the C2C RI receives a valid trafficNetworkInformationRequestMsg message for a Node from an external center as part of a dlTrafficNetworkInformationSubscription dialog, the C2C RI shall send a c2cMessageReceipt response	3.2.2.4.15	Node (inventory, status)	NodeHandler(5.4.1.1.2.2.2.1) RIEmulation(5.4.1.1.2.5) SubscriptionOCReceiveTag(5.3.1.2.46) SubscriptionOCReplyTag(5.3.1.2.47) ApplicationLayerStandard(5.3.2.2.4) Entity Emulation Overview(6.2) Section 6.2 includes a description on how the messages (and data) in the dialogs are processed. The content is based on the message definitions

Requirement	SRS Section	Title	Design
message to the external center.			included in TMDD Volume II. Unless the values are modified as part of the control request processing or a script initiated value change, the parameter values will be as specified in the Test Configuration at the start of a test. Note that the C2C RI's ability to satisfy this performance requirement will be impacted by the performance of the underlying selected application layer standard.
SRS157: When acting as an owner center and TMDD entity emulation is selected, the C2C RI shall publish Node information updates to a subscribing external center. SRS157.1: The C2C RI shall publish a nodeInventoryMsg message as part of a dlNodeInventoryUpdate dialog to an external center in accordance with current subscriptions. SRS157.1.1: If the current subscription is for event-driven updates, the C2C RI shall initiate the publication within 1 minute of a change in Node Inventory (see Requirement SRS95) when the changed data item(s) are included within the original subscription request. SRS157.2: The C2C RI shall publish a nodeStatusMsg message as part of a dlNodeStatusUpdate dialog to an external center in accordance with current subscriptions. SRS157.2.1: If the current subscription is for event-driven updates, the C2C RI shall initiate the publication within 1 minute of a change in Node Status (see Requirement SRS95) when the changed data item(s) are included within the original subscription request.	3.2.2.4.15	Node (inventory, status)	LinkHandler(5.4.1.1.2.3.2.1) LinkInventory(5.4.1.1.2.3.2.2) LinkStatus(5.4.1.1.2.3.2.3) RIEmulation(5.4.1.1.2.5) Entity Emulation Overview(6.2) Section 6.2 includes a description on how the messages (and data) in the dialogs are processed. The content is based on the message definitions included in TMDD Volume II. Unless the values are modified as part of the control request processing or a script initiated value change, the parameter values will be as specified in the Test Configuration at the start of a test. Note that the C2C RI's ability to satisfy this performance requirement will be impacted by the performance of the underlying selected application layer standard.

Requirement	SRS Section	Title	Design
SRS158: The C2C RI shall provide the test developer with an interface to edit and save test case data files. SRS158.1: The interface will allow the user to create a custom test case data file for a test case. SRS158.2: The interface will allow the user to edit a custom test case data file for a test case. SRS158.3: The interface will show how the edits to the test case data will affect the messages that will be transmitted during testing.	3.2.1.1.3	Edit and Store User Defined Test Case Data	Test Case Editor (5.1.2.2.12)