



System Design Document for R-ICMS: Regional Integrated Corridor Management System

Version: 4.0

Approval date: 8/15/2020



System Design Document for Regional Integrated Corridor Management System (R-ICMS)

DOCUMENT CONTROL PANEL		
File Name & Location	C:\Users\cweston\Downloads\R-ICMS-SDD-4.0.docx	
Version Number:	4.0	
	Name	Date
Created By:	Clay Weston, SwRI	5/12/2020
Reviewed By:	John Horner and Kevin Miller	6/17/2020
	Clay Packard	6/19/2020
Modified By:	Clay Weston	7/29/2020
	Bill Kuhn	8/1/2020
	Jared Allen	8/1/2020
	Clay Weston	8/14/2020
Approved By:	Tushar Patel	8/15/2020

Table of Contents

1	Overview.....	1
1.1	<i>Identification</i>	<i>1</i>
1.2	<i>System Overview</i>	<i>1</i>
2	Reference Documents	1
3	Design	2
3.1	<i>System Context.....</i>	<i>2</i>
3.2	<i>Design Assumptions and Constraints [by Category]</i>	<i>3</i>
3.2.1	System Interfaces	4
3.2.2	Operational Assumptions.....	4
3.2.3	Non-operational Design	5
3.2.4	System Inputs and Outputs.....	5
3.2.5	Behavior and Performance	6
3.2.6	User View of Data	6
3.2.7	Safety, Security, and Privacy	7
3.2.8	Third Party Components	7
3.3	<i>System Architecture.....</i>	<i>13</i>
3.4	<i>System Components.....</i>	<i>16</i>
3.4.1	DFE Layers and Components.....	16
3.4.1.1	<i>Drivers.....</i>	<i>18</i>
3.4.1.2	<i>Pipelines.....</i>	<i>53</i>
3.4.1.3	<i>Data Store.....</i>	<i>72</i>
3.4.1.4	<i>Data Services</i>	<i>78</i>
3.4.2	DSS / Business Services	86
3.4.2.1	<i>Authentication (AuthN-BS).....</i>	<i>86</i>
3.4.2.2	<i>Authorization (AuthZ-BS).....</i>	<i>88</i>
3.4.2.3	<i>Notification (NOT-BS)</i>	<i>89</i>
3.4.2.4	<i>Response Plan Selection (RPS).....</i>	<i>91</i>
3.4.2.5	<i>Simulation Interfaces.....</i>	<i>96</i>

- 3.4.2.6 *Signal Optimization Timing (SOT-BS)* 102
- 3.4.2.7 *SunGuide (SG-BS)*..... 110
- 3.4.3 **User Interface** 110
 - 3.4.3.1 *Authentication (Authn-UI)*..... 112
 - 3.4.3.2 *Map (MAP-UI)* 113
 - 3.4.3.3 *Event List Interface* 124
 - 3.4.3.4 *Authorization (AuthZ-UI)* 132
 - 3.4.3.5 *Notifications (NOT-UI)* 139
 - 3.4.3.6 *Reporting (RP-UI)*..... 145
 - 3.4.3.7 *Dashboards (DB-UI)* 153
 - 3.4.3.8 *Traveler Advisory Messages (TAM-UI)* 154
 - 3.4.3.9 *DIVAS (DIV-UI)* 156
 - 3.4.3.10 *Response Plan Selection* 157
 - 3.4.3.11 *Signal Optimization Timing (SOT-UI)*..... 169
- 3.4.4 **Common Core** 192
 - 3.4.4.1 *Kubernetes*..... 192
 - 3.4.4.2 *Authorization (AuthZ-CC)* 193
 - 3.4.4.3 *Logging (Log-CC)*..... 193
 - 3.4.4.4 *Monitoring (Mon-CC)* 194
 - 3.4.4.5 *Notification (Not-CC)* 197
 - 3.4.4.6 *Gateway (GW-CC)*..... 197
- 3.5 System Resources** **198**
 - 3.5.1 **Hardware Configuration** 198
 - 3.5.2 **Physical Deployment: Servers and Network**..... 199
- 3.6 Concept of Execution**.....**200**
 - 3.6.1 **Security**..... 201
 - 3.6.1.1 *Login* 201
 - 3.6.1.2 *User Data Request*..... 203
 - 3.6.1.3 *User Authorization Update*..... 205
 - 3.6.1.4 *User Personalization*..... 207
 - 3.6.1.5 *GIS Use Cases*..... 209

3.6.2	Situational Awareness	214
3.6.2.1	<i>ITSIQA Ingestion</i>	214
3.6.2.2	<i>Map Update for Pre-Located Data</i>	216
3.6.2.3	<i>Map Update for Dynamically Located Data</i>	218
3.6.3	Event Diagrams.....	220
3.6.3.1	<i>SunGuide / R-ICMS Event</i>	220
3.6.3.2	<i>Response Plan Selection</i>	224
3.6.3.3	<i>Response Plan Approval</i>	226
3.6.3.4	<i>Response Plan Activation</i>	228
3.6.3.5	<i>Response Plan Monitor/Reevaluation</i>	230
3.6.4	Signal Optimization Timing Diagrams.....	232
3.6.4.1	<i>Periodic Optimization</i>	232
3.6.4.2	<i>On Demand Optimization</i>	234
3.6.4.3	<i>Modified Timing Plan</i>	236
3.6.4.4	<i>Detailed: Create Corridor</i>	238
3.6.4.5	<i>Detailed: Intersection Features</i>	240
3.6.4.6	<i>Detailed: Modify Corridor</i>	242
3.6.4.7	<i>Detailed: Corridor Optimization 1</i>	244
3.6.4.8	<i>Detailed: Corridor Optimization 2</i>	246
3.6.4.9	<i>Detailed: Deploy Corridor</i>	248
3.6.4.10	<i>Detailed: SOT Results Simulation Interface</i>	250
3.6.5	Other Use Case Diagrams	252
3.6.5.1	<i>Other: External Access – Authentication</i>	252
3.6.5.2	<i>Other: External Access – Data Request</i>	254
3.6.5.3	<i>Other: External Access – Streaming</i>	255
3.6.5.4	<i>Other: External Access – Metadata</i>	256
3.6.5.5	<i>Other: Parking</i>	256
3.6.5.6	<i>Other: Alerts, Information, and Notification</i>	258
3.6.5.10	<i>Other: Monitoring</i>	266
3.6.5.11	<i>Signal Controller Settings Data Service</i>	266
3.7	System Deployment	268

- 3.7.1 Service Host Deployment..... 268
- 3.7.2 Containerized Service Orchestration..... 269

List of Tables

Table 1 – Reference Documents	2
Table 2 – R-ICMS Fundamental Capabilities	5
Table 3 – Third Party Components	8
Table 4 – ITSQA Current Traffic Conditions Driver – Ingestion Parameters.....	26
Table 5 – ITSQA Archived Traffic Conditions Driver – Ingestion Parameters	29
Table 6 – SunGuide DMS Ingestion.....	33
Table 7 – SunGuide Events Ingestion.....	35
Table 8 – SunGuide CCTV Ingestion	37
Table 9 – SunGuide Truck Parking Ingestion	39
Table 10 – SunGuide Traffic Signals Ingestion	41
Table 11 – SunGuide Ramp Meters Ingestion	43
Table 12 - SunGuide Connected Vehicles Ingestion	44
Table 13 – Transit AVL Current Data Ingestion.....	45
Table 14 – Transit AVL Archive Data Ingestion	46
Table 15 – Transit Static Data Ingestion	47
Table 16 – Signal Controller Log Ingestion	48
Table 17 – Basemap Specification	48
Table 18 – School Zones Specification	49
Table 19 – School Locations Specification	50
Table 20 – Emergency Responder Locations Specification	51
Table 21 – Signalized Intersections Ingestion.....	52
Table 22 – Weather Alerts Ingestion	52
Table 23 – ITSQA Current Traffic Conditions Pipeline Specifications	55
Table 24 – ITSQA Archive Traffic Conditions Pipeline.....	56
Table 25 – SunGuide DMS Pipeline.....	58
Table 26 – SunGuide CCTV Pipeline	59
Table 27 – SunGuide Events Pipeline.....	60
Table 28 – SunGuide Truck Parking Pipeline	61
Table 29 – SunGuide Traffic Signals Pipeline	62
Table 30 – SunGuide Ramp Meters Pipeline	63

Table 31 - SunGuide Connected Vehicles Pipeline	65
Table 32 – Transit AVL Current Data Pipeline.....	66
Table 33 – Transit AVL Archive Data Pipeline.....	67
Table 34 – Transit Static Data Pipeline	68
Table 35 - Signal Controller Log Pipeline	69
Table 36 – Signalized Intersections Pipeline.....	71
Table 37 – Weather Alerts Pipeline	72
Table 38 – Generic Data Services.....	79
Table 39 – GIS Data Services.....	80
Table 40 – Stream Service Details.....	82
Table 41 – ArcGIS Feature Service API Details.....	82
Table 42 – ArcGIS Vector Layer Server API	85
Table 43 – R-ICMS Data Services	85
Table 44 - SOT real-time data sources.....	103
Table 45 – SOT non-standard intersection conditions	104
Table 46 – Response Plan User Interface Actions	158
Table 47 – Response Plan Selection Applicability Fields	168
Table 48 - SOT Intersection Restrictions.....	176
Table 49 – SOT Intersection Traffic Volume Defaults.....	178
Table 50 – Monitoring Alerts	195
Table 51 – Hardware Configuration.....	198
Table 52 – Use Case View Map Details.....	210
Table 53 – View GIS Layers Use Case Details.....	211
Table 54 – View Details Use Case Details	212
Table 55 – Receive Static Data Use Case Details	213
Table 56 – Receive Dynamic Data Use Case Details	214
Table 57 – Data Sources Table.....	272

List of Figures

Figure 1 – R-ICMS Context Diagram.....	3
Figure 2 – System Architecture Layer Color Legend.....	14
Figure 3 – R-ICMS High-Level Diagram	15

System Design Document for Regional Integrated Corridor Management System (R-ICMS)

Figure 4 – R-ICMS Data Streaming Workflow 17

Figure 5 – Ingestion Flow 19

Figure 6 – SunGuide Data Ingestion Process 22

Figure 7 – SunGuide Data Flow..... 22

Figure 8 – ITSQA Data Flow..... 24

Figure 9 – R-ICMS Pipeline 53

Figure 10 – TRF-CTC-PL Pipeline 54

Figure 11 – TRF-ATC-PL Diagram 56

Figure 12 – SunGuide Generalized Pipeline Diagram 57

Figure 13 – Transit AVL Current Data Pipeline Diagram 66

Figure 14 – Transit AVL Archive Data Pipeline Diagram 67

Figure 15 – Transit Static Data Pipeline Diagram 68

Figure 16 – Signal Controller Log Pipeline Diagram..... 69

Figure 17 – Signalized Intersections Pipeline Diagram 71

Figure 18 – Weather Alerts Pipeline Diagram 72

Figure 19 – GIS Data Store 76

Figure 20 – Geodata Shape File Data Flow 77

Figure 21 – Map Layer Shape File 78

Figure 22 – Subscription Service 79

Figure 23 – GeoEvent Server Flow 81

Figure 24 – Real-time GeoEvent Data Flow 82

Figure 25 – Response plan selection sequence diagram 94

Figure 26 : Simulation Interface – Timing Plans API classes 98

Figure 27 : Simulation Interface – Volume over capacity classes..... 98

Figure 28 : Simulation Interface – Travel Times classes 98

Figure 29 : Simulation Interface – Response Plan Simulation classes 99

Figure 30 : Simulation Interface – Historical Norm classes 99

Figure 31 : Simulation Callback Interface – Timing Plan Score classes..... 101

Figure 32 : Simulation Callback Interface – Response Plan Scores..... 102

Figure 33 – SOT Operational Database Diagram 107

Figure 34 – Angular MVC 111

System Design Document for Regional Integrated Corridor Management System (R-ICMS)

Figure 35 – User Interface Navigation Hierarchy..... 112

Figure 36 – Login Screen 113

Figure 37 – Map Data Flow 114

Figure 38 – Main Screen 115

Figure 39 – Search Screen..... 115

Figure 40 – Zoom features 116

Figure 41 – Home button..... 116

Figure 42 – Feature Details 117

Figure 43 – Layer Selection Screen 118

Figure 44 – Traffic Condition Layer Selection 118

Figure 45 – Basemap Selection 119

Figure 46 – Legend Selection 119

Figure 47 – Navigation 120

Figure 48 – Total Storm Precipitation..... 121

Figure 49 – Storm Watches & Warnings..... 122

Figure 50 – Tropical Cyclone Watches & Warnings..... 123

Figure 51 – Tropical Cyclone Watches & Warnings..... 123

Figure 52 – Main Menu Event List 124

Figure 53 – Map Page Event List..... 125

Figure 54 –Main Menu Event Details..... 125

Figure 55 – Map Page Event Details 126

Figure 56 – Email Event..... 132

Figure 57 – Map Page Email Event..... 132

Figure 58 – Device Group List Screen 133

Figure 59 – Add Device Group Screen 134

Figure 60 – Device Group Detail Screen 135

Figure 61 – Device Assignment Screen 136

Figure 62 – Device List Screen 136

Figure 63 – Device Approval Profile Screen..... 137

Figure 64 – Role List Screen 137

Figure 65 – Add Role Screen 138

Figure 66 – Role Detail Screen	139
Figure 67 – Notification Popup Dismissal	140
Figure 68 – Notification Popup Snoozing.....	141
Figure 69 – Notification Indicator	141
Figure 70 – Notification Feed.....	142
Figure 71 – Notification Feed with New Notification	143
Figure 72 – Alert Resolution	144
Figure 73 – Notification Feed with Resolved Alert	145
Figure 74 – Dashboard Display	154
Figure 75 – TAM Display	155
Figure 76 – TAM List Selection.....	Error! Bookmark not defined.
Figure 77 – Event List with Diversion Route Status	160
Figure 78 - Response Plan Selection: Awaiting Manager Selection.....	161
Figure 79 - Response Plan Selection: Awaiting Device Approval.....	162
Figure 80 - Response Plan Selection: Devices Overridden	163
Figure 81 - Response Plan Selection: Awaiting Approval	164
Figure 82 - Response Plan Selection: Devices Approved.....	165
Figure 83 - Response Plan Selection: Awaiting Activation	166
Figure 84 – Response Plan Selection: Settings List.....	167
Figure 85 – Response Plan Selection: Setting Creation/Modification.....	168
Figure 86 – SOT corridor list.....	171
Figure 87 – SOT Step 1: Corridor.....	172
Figure 88 – SOT Step 2: Schedule	173
Figure 89 - SOT Step 2: Schedule – Cluster Settings	173
Figure 90 – SOT Step 3: Intersections	174
Figure 91 – SOT Step 3: Intersections Geometry Details.....	174
Figure 92 – SOT Step 3: Intersections Overlap Phases	175
Figure 93 – SOT Step 3: Intersections (D – restriction).....	176
Figure 94 – SOT Step 3: Intersection (E - traffic volume).....	179
Figure 95 - SOT All Clusters – Simulation Region & Corridor statistics.....	180
Figure 96 - SOT All Clusters – Simulation Signals statistics.....	181

Figure 97 - SOT All Clusters – Simulation Delay % Improvement heatmap.....	182
Figure 98 - SOT Time Cluster Tabs– HCS7 Segments & Signal statistics	183
Figure 99 – SOT Time Cluster 2 – Signal Offsets	184
Figure 100 - SOT Time Cluster Tabs – Signal Splits	185
Figure 101 - SOT All Clusters – Reviews & Approvals	186
Figure 102 - SOT All Clusters – Timing Plans.....	187
Figure 103 - SOT All Clusters – Deploy Corridor	188
Figure 104 - SOT alert.....	190
Figure 105 - SOT alert and information message flow-chart.....	190
Figure 106 - SOT information feed –review requested	191
Figure 107 - SOT information feed – review denied.....	191
Figure 108 - SOT information feed – re-evaluation approved and deployed.....	192
Figure 109 – Unresolvable Alert Monitoring Sequence Diagram.....	196
Figure 110 – Resolvable Alert Monitoring Sequence Diagram.....	197
Figure 111 – Physical Deployment: Servers and Network Diagram	200
Figure 112 – Login Sequence Diagram.....	202
Figure 113 – User Data Request Sequence Diagram	204
Figure 114 – User Authorization Update Sequence Diagram.....	206
Figure 115 – User Personalization Sequence Diagram	208
Figure 116 – GIS Use Cases	209
Figure 117 – ITSQA Ingestion Sequence Diagram.....	215
Figure 118 – Map Update for Pre-Located Data Sequence Diagram	217
Figure 119 – Map Update for Dynamically Located Data Sequence Diagram	219
Figure 120 – SunGuide / R-ICMS Event Flow Diagram	221
Figure 121 – SunGuide / R-ICMS Event Flow Sequence Diagram.....	223
Figure 122 – Response Plan Selection Sequence Diagram	225
Figure 123 – Response Plan Approval Sequence Diagram	227
Figure 124 – Response Plan Activation Sequence Diagram	229
Figure 125 – Response Plan Monitor/Reevaluation Sequence Diagram.....	231
Figure 126 – SOT - Periodic Optimization Sequence Diagram.....	233
Figure 127 – SOT On Demand Optimization Sequence Diagram.....	235

Figure 128 – SOT Modified Timing Plan Sequence Diagram 237

Figure 129 – SOT Detailed Create Corridor Sequence Diagram 239

Figure 130 – SOT Detailed Intersection Features Diagram..... 241

Figure 131 – SOT Detailed Modify Corridor Sequence Diagram 243

Figure 132 – SOT Detailed Corridor Optimization 1 245

Figure 133– SOT Detailed Corridor Optimization 2 247

Figure 134 – SOT Detailed Deploy Corridor 249

Figure 135 – SOT Detailed Results Simulation Interface 251

Figure 136– External Access - Authentication Sequence Diagram..... 253

Figure 137 – External Access – Data Request Sequence Diagram..... 254

Figure 138 – External Access – Streaming Sequence Diagram 255

Figure 139 – External Access – Metadata Sequence Diagram 256

Figure 140– Parking Sequence Diagram 257

Figure 141– Alerts, Information, and Notification Flow Chart Diagram..... 259

Figure 142– Response Plan Notification Sequence Diagram..... 261

Figure 143 – Data Source Outage Alert Sequence Diagram 263

Figure 144– Logging Sequence Diagram..... 265

Figure 145– Monitoring Sequence Diagram 266

Figure 146– Service Host Deployment Diagram 268

Figure 147 – Containerized Service Orchestration Diagram 269

List of Acronyms and Abbreviations

AAM	Active Arterial Management
AM	Ante Meridiem
API	Application Program Interface
AST	Agency for State Technology
ATMS	Advanced Traffic Management System
AVL	Automatic Vehicle Location
AWS	Amazon Web Services
CCTV	Closed Circuit Television
CLI	Command Line Interface
COTS	Commercial Off the Shelf
CSV	Comma Separated Variable
DFE	Data Fusion Environment
DMS	Dynamic Message Signs
DOT	Department of Transportation
DSS	Decision Support System
ERD	Entity Relationship Diagram
ETL	Extract, Transform, Load
FCS	Florida Cybersecurity Standards
FDOT	Florida Department of Transportation
FTP/SFTP	File Transport Protocol / Secure File Transport Protocol
GIS	Geographic Information System
GTFS	General Transit Feed Specification
GTFS-RT	General Transit Feed Specification Real Time
HDFS	Hadoop Distributed File System
HTTPS	Hyper Text Transfer Protocol Secure
ICD	Interface Control Document
ID	Identifier
IEN	Information Exchange Network
IMC	Intersection Movement Counts
IT	Information Technology
ITS	Intelligent Transportation System
ITSIQA	Intelligent Transportation System Input Quality Assurance
JSON	JavaScript Object Notation
JWT	JSON Web Tokens
LDAP	Lightweight Directory Access Protocol
ME	Modeling Engine
MOE	Measure of Effectiveness
MS SQL	Microsoft SQL
MVC	Model View Controller
OAS	OpenAPI Specification
PD	Preliminary Design
PDF	Portable Document Format
PDR	Preliminary Design Review
PM	Post Meridiem
RCI	Roadway Characteristics Inventory
RDBMS	Relational DataBase Management System
REST	Representational State Transfer
R-ICMS	Regional Integrated Corridor Management System
RP	Response Plan
RPE	Response Plan Element

System Design Document for Regional Integrated Corridor Management System (R-ICMS)

RTOR	Right Turn on Red
SDD	System Design Document
SHS	State Highway System
SLES	SUSE Linux Enterprise Server
SOT.....	Signal Optimization Tool
SQL	Structured Query Language
SSL	Secure Sockets Layer
TBD	To Be Determined
TGDC.....	Time Grouped Demand Cluster
TLS.....	Transport Layer Security
TSMO	Transportation Systems Management and Operations
UI	User Interface
UML	Unified Modeling Language
URL	Uniform Resource Locator
XML	Extensible Markup Language

Acronyms are used as Suffixes to Components

BS	Business Service
CC.....	Common Core
DR	Driver layer
DS.....	Data Service
GW	Gateway
PL	Pipeline layer
ST	Data Store
UI	User Interface

Component Abbreviations

ADM-BS.....	Admin – Business Service
ADM-UI	Admin – User Interface
AL-CC	Alerting – Common Core
ALT-BS	Alert – Business Service
ALT-UI	Alerts – User Interface
AUTH-CC	Authorization – Common Core
BM-DR.....	Basemap – Driver
BRE-BS.....	Business Rules Engine – Business Service
CT5-PL	Traffic Conditions 5-Minute Average – Pipeline
CTC-PL	Current Traffic Conditions – Pipeline
EM-BS	Event – Business Service
EM-DS	Event Management – Data Service
EM-UI	Event Management – User Interface
ERL-DR	Emergency Responder Locations (GIS) – Driver
EVE-BS.....	Evaluation Engine – Business Service
EVT-PL	Events – Pipeline
FS-ST	File Store – Data Store
GEO-DS	Geographic Data Service
GIS-ST.....	GIS Database – Data Store
GW-CC	Gateway – Common Core
HCS7.....	Highway Capacity Manual Version 7
IDS-PL.....	ITS Device Status – Pipeline
IMC-DR.....	Intersection Movement Counts – Driver
IMC-PL.....	Intersection Movement Counts – Pipeline

System Design Document for Regional Integrated Corridor Management System (R-ICMS)

ITC-PL	ITS Device Status – Pipeline
LOG-CC	Logging – Common Core
Login-UI	Login – User Interface
MAP-UI	Map – User Interface
MD-DS	Metadata – Data Service
MON-CC	Monitoring – Common Core
NOS-ST	NoSQL – Data Store
NOT-UI	Notifications – User Interface
ORC-BS	Orchestration – Business Service
PK-PL	Parking – Pipeline
PRE-BS	Predictive Engine – Business Service
RCI-DR	RCI Number of Lanes at Intersection (GIS) – Driver
RP-BS	Response Plan (RP-BS) – Business Service
RPM-UI	Response Plan Management – User Interface
RPT-BS	Reporting – Business Service
RP-UI	Reporting – User Interface
SA-BS	Session Authentication – Business Service
SCL-DR	Signal Controller Log – Driver
SCL-PL	Signal Controller Log – Pipeline
SCL-CSV-DS	Signal Controller Log CSV – Data Service
SCL-CSV-DR	Signal Controller Log CSV – Driver
SCL-CSV-PL	Signal Controller Log CSV – Pipeline
SCL-DAT-DR	Signal Controller Log DAT – Driver
SE-DR	Special Event – Driver
SG-DR	SunGuide CCTV –Data Service
SG-CCTV-DS	SunGuide CCTV -Driver
SG-CCTV-DR	SunGuide CCTV -Driver
SG-CCTV-PL	SunGuide CCTV -Pipeline
SG-DMS-DS	SunGuide DMS –Data Service
SG-DMS-DR	SunGuide DMS -Driver
SG-DMS-PL	SunGuide DMS -Pipeline
SG-EV-DS	SunGuide Event –Data Service
SG-EV-DR	SunGuide Event -Driver
SG-EV-PL	SunGuide Event -Pipeline
SL-DR	School Locations (GIS) – Driver
SPM-DR	Signal Performance Measures – Driver
SPM-DAT-DR	Signal Performance Measures – Driver
SOT-BS	Signal Optimization Timing – Business Service
SOT-DS	Signal Optimization Timing – Data Service
SOT-UI	Signal Optimization Timing – User Interface
SQL-ST	SQL Database – Data Store
ST-PL	Signal Timing – Pipeline
SZ-DR	School Zones (GIS) – Driver
TA-DR	Transit AVL – Driver
TSD-DR	Transit Static Data – Driver
TAVL-PL	Transit AVL – Pipeline
TL-DR	Transit Locations – Driver
TRF-DR	Traffic – Driver
TRF-PL	Traffic – Pipeline
TRF-ATC-DR	ITS IQA Archive Traffic Conditions – Driver
TRF-ATC-PL	ITS IQA Archive Traffic Conditions – Pipeline
TRF-CTC-DR	ITS IQA Current Traffic Conditions – Driver
TRF-CTC-PL	ITS IQA Current Traffic Conditions – Pipeline

System Design Document for Regional Integrated Corridor Management System (R-ICMS)

UP-BS User Personalization – Business Service
VID-DR Video – Driver
WEA-DR Weather – Driver
YAML..... Yet Another Markup Language

1 Overview

This System Design Document (SDD) describes the detailed design for the Regional – Integrated Corridor Management System (R-ICMS). Version 4 of the document includes the preliminary design elements as well as the Critical Design elements through Iteration 4; there will be a total of 4 Iterations.

This document enables a system developer to understand how to further design and implement the R-ICMS, and this document is a communication tool for stakeholder concurrence and a record of how the system will meet system requirements.

1.1 Identification

The project for which this document was created is identified by the following:

Project Name: Central Florida Regional Integrated Corridor Management System

Agreement Number: BE521

Financial Project Identification: 436328-1-82-01

Federal Aid Project Number: Not Applicable

1.2 System Overview

The R-ICMS is intended to be an initial implementation of a multi-modal regional transportation management system. The R-ICMS will integrate freeway, arterial, transit, and rail transportation management for the I4 corridor, including management of transportation system components owned and operated by the state, as well as the county, city, and regional transportation agencies.

The R-ICMS will consist of, but not be limited to; commercial off-the-shelf (COTS) modeling software, a custom-built Decision Support System (DSS), a custom-built Information Exchange Network (IEN) subsystem that includes dashboards and other user interfaces to the system, and a Data Fusion Environment (DFE) to host data sources for both the R-ICMS and other external users and applications.

This project is funded and managed by District 5 of the Florida Department of Transportation (FDOT). It is intended for the use of District personnel, as well as personnel from the cities, counties, and transportation agencies located within the District. The initial deployment of the R-ICMS will be to the Transportation Management Center being built in District 5 by the FDOT.

2 Reference Documents

The following documents, of the exact issue shown, form a part of this document to the extent specified herein. In the event of a conflict between the documents referenced herein and the contents of this document, this document shall be considered the superseding requirement.

Table 1 – Reference Documents

System and Subsystem Requirements Specification for R-ICMS for: Regional Integrated Corridor Management System: R-ICMS-REQ-0.2	Southwest Research Institute FDOT R-ICMS Project SharePoint Site
BE521 - Executed Contract	Florida Department of Transportation D5prcustodian@dot.state.fl.us
SunGuide ICDs	Southwest Research Institute FDOT R-ICMS Project SharePoint Site http://www.SunGuidesoftware.com/document-library
Database Model Reference Document for RICMS R-ICMS-DM-1.1.docx	Southwest Research Institute FDOT R-ICMS Project SharePoint Site
Data Service Application Programmer Interface Reference Document R-ICMS-DSAPI-1.1.docx	Southwest Research Institute FDOT R-ICMS Project SharePoint Site
Kafka Documentation	Apache Software Foundation http://kafka.apache.org
ITSIQA ICD	Florida Department of Transportation http://www.cflsmartroads.com/projects/smartsignals/ITSIQA%20Design.pdf
Kubernetes Documentation	Cloud Native Computing Foundation https://kubernetes.io/docs/home/
TMDD v3.03d	https://www.ite.org/technical-resources/standards/tmdd/3-03/
SIIA Design Document	Florida Department of Transportation http://www.cflsmartroads.com/projects/smartsignals/Signal%20Inventory%20Tool%20Application%20Design%20Draft.pdf
HCS7 Documentation Set	https://mctrans.ce.ufl.edu/mct/index.php/hcs/hcs-downloads/ http://www.trb.org/Main/Blurbs/175169.aspx
Aimsun Documentation	Florida Department of Transportation

3 Design

This section documents the system design.

3.1 System Context

The R-ICMS will be integrated into the FDOT District 5 Transportation Systems Management and Operations (TSMO) systems. It will collect data from FDOT and other agency systems. It will store the data in a DFE, which will also provide the data to other components of the R-ICMS and to external users. Via a user interface referred to as the IEN, the R-ICMS will support FDOT and agency users in conducting traffic management operations. The system will support the

development, storage, and use of response plans to be used to manage the transportation infrastructure of the corridor in response to periodic and non-recurring congestion. The R-ICMS will support the development, storage and use of the set of business rules that determine when response plans should be suggested to users. The R-ICMS will include a DSS which will support the business functions of the users. The system will execute the business rules and determine when a set of response plans should be recommended to the users. The R-ICMS will support the selection of a response plan, its approval by affected agency users, and the implementation of the response plan. The R-ICMS will support the management of parking resources in the corridor and the optimization of signal timing plans for linear groups of signals. The R-ICMS will also support basic access to the data stored in the DFE both by other R-ICMS subsystems and by external users.

Figure 1 – R-ICMS Context Diagram provides an overview of the TSMO environment and the place of the R-ICMS and its DFE in that environment.

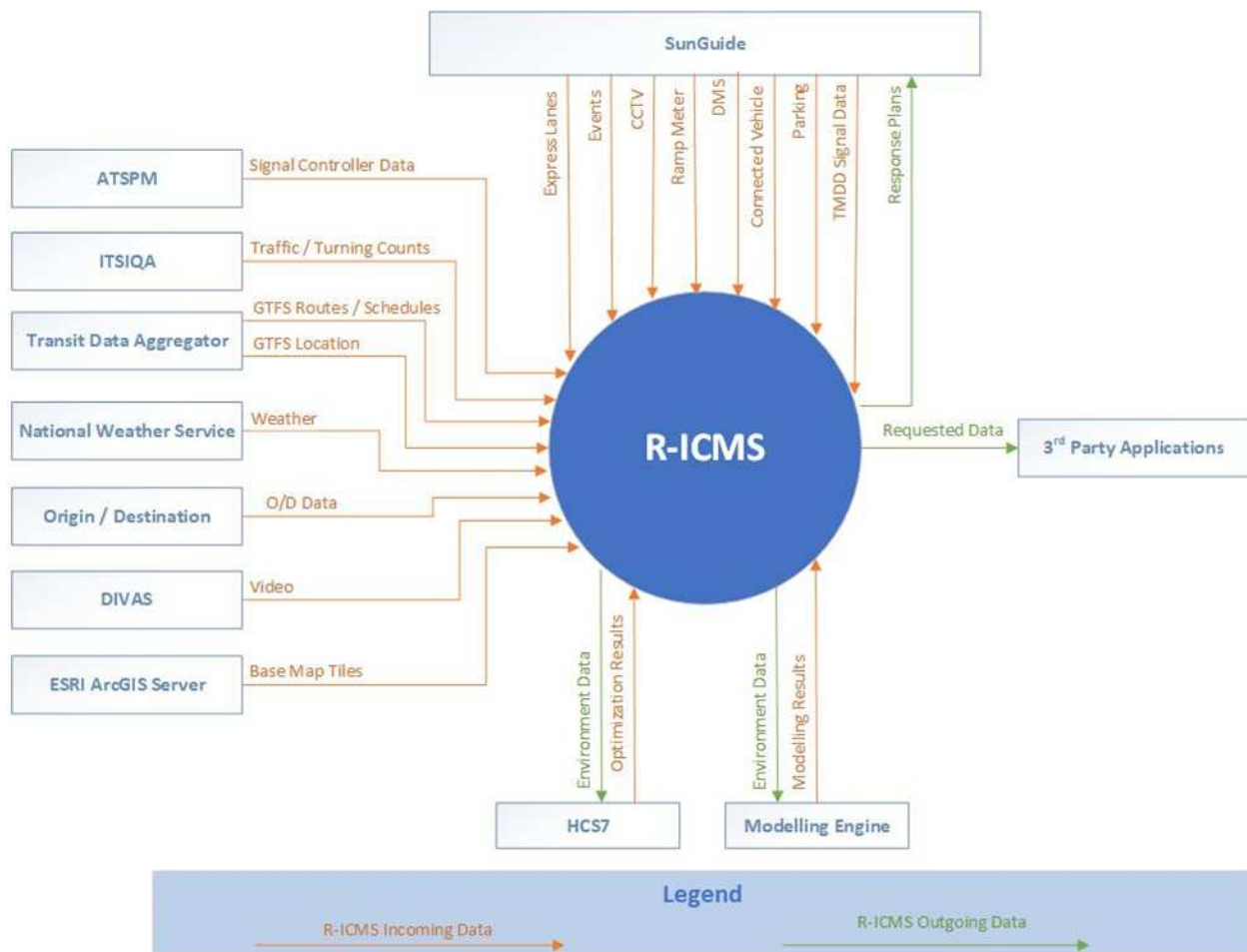


Figure 1 – R-ICMS Context Diagram

3.2 Design Assumptions and Constraints [by Category]

The R-ICMS design philosophy is centered around the use of architecturally significant use cases to drive an architecture intended to satisfy the R-ICMS requirements. Architecturally significant

use cases are those which have structural or performance impacts on the architecture needed to satisfy the users’ needs. This document, in addition to the high-level design information, contains more detailed design for those use cases selected as being architecturally significant and those use cases intended to be implemented in Iteration 1. Use cases not detailed in this document are not less important and are not being ignored; they were just not the specific ones selected to drive the architecture. Once the architecture has been designed, it is anticipated that there should be relatively few changes to the high-level design as the other use cases are added. In general, UML sequence diagrams have been used to illustrate the interactions between architectural components. Occasionally, other diagram types have been used to illustrate specific portions of the architecture or the design.

The following paragraphs describe high-level design assumptions and constraints considered in developing the architecture.

3.2.1 System Interfaces

The R-ICMS will interface with multiple different systems to consume/aggregate data, provide response plan actions, and provide data to external systems. The following subsections describe the different interfaces

Name	Description
Data Source Drivers	The system will utilize drivers to connect to various data sources. The data sources provide the input data that the system uses to produce operation results. The system will handle reconnect attempts as necessary and will alert when data is unavailable to be ingested.
SunGuide Driver	The system will communicate with SunGuide in order to ingest SunGuide Data. Additionally, the system will communicate with SunGuide in order to add comments, event histories, and will pass response plan data to SunGuide for implementation.
Third Party Access	Data ingested into the Data Fusion Environment and data created from the operational use of the system will be made available via subscription service and REST APIs.
Modelling Engine	The system will interface with the Aimsun modelling engine to request travel times, historical norm data, SOT simulations, and response plan simulations. Simulation Interface API definitions can be found in section 3.4.2.5
Signal Timing Engine	The signal optimization tool communicates with the HCS7 streets tool to produce optimization results. Data will be combined from ITSQA, TMDD, and SIIA to produce HCS7 input data. This data will be used as input to HCS7 which will then produce optimization result data that, when combined with simulation results from the Aimsun modelling engine, will produce

3.2.2 Operational Assumptions

A number of design decisions were made based on operational assumptions which include:

- The R-ICMS will consist of a combination of open source, commercial and custom software

- The DFE will make use of Structured Query Language (SQL) databases, noSQL databases, Geospatial Information System (GIS) data, File Stores, and streaming data pipelines
- The R-ICMS will operate in a web services environment, and the IEN will operate within a browser on the user’s system
- In the future, the R-ICMS will provide access to transportation-related data for external systems developed or purchased by the District or other agencies
- In the future, the DFE will be integrated into the Smart Cities infrastructure of the region and provide data and analytic access to applications used to integrate Corridor systems
- R-ICMS will utilize external systems to implement some elements of response plans, e.g. R-ICMS will request SunGuide modify DMS messages, modify signal timing plans, etc.
- FDOT will provide hosting and operational environment (Traffic Management Center & Data Center)
- The Aimsun traffic simulation tool will be used by the Predictive Engine to simulate proposed signal timing plans and proposed response plans
- Authentication will be based on FDOT’s Active Directory, but fine-grained authorization will be based on a combination of Active Directory roles and an internal data store of permissions associated with those roles.

3.2.3 Non-operational Design

A number of design decisions were made based on Non-operational design assumptions and constraints which include:

- The R-ICMS will operate within the FDOT D5 TSMO Intelligent Transportation Systems (ITS) network.

3.2.4 System Inputs and Outputs

Selection of the set of inputs to be addressed during the initial phase of the implementation of the R-ICMS was based on several fundamental capabilities needed by the initial phase of the system. They are listed and described in Table 2 – R-ICMS Fundamental Capabilities.

Table 2 – R-ICMS Fundamental Capabilities

Name	Description
Situational Awareness	The capability of providing users with an overall understanding of the current traffic and parking conditions within the core traffic network covered by the R-ICMS
Response Plan Execution	The ability of users to respond to non-recurring events generated by SunGuide by providing suggested response plans and allowing users to select and approve the response plans for implementation
Response Plan Monitoring	The capability of monitoring the traffic network while response plans are active to determine if the response plan should be replaced or deactivated
Signal Timing Plan Optimization	The capability of providing authorized users the ability to develop signal timing plan sets based on collected traffic data

The Data Sources table (see Appendix A) provides the specification of the set of inputs to be received. The contract provides notional representations of the user interaction screens for some of the user interactions. Additional details of the interactions are being provided as a part of the detailed design. Additional information specifying the formats and protocols of the system interactions are being provided on an ongoing basis by FDOT.

3.2.5 Behavior and Performance

This section provides the behavior and performance assumptions and constraints.

For performance and architectural clarity, a layered architecture has been defined for the system. The set of layers and their functionality is provided in Section 3.3 System Architecture.

Since one of the primary capabilities to be provided by the R-ICMS is an understanding of the current traffic network situation, a map-based interface to the real-time (i.e., 1-minute updates) data from the traffic network sensors is a major focus of the system, and therefore, a major influence on the architectural decisions. As proposed, ArcGIS is being used as the basis of the map user interface. The use of ArcGIS has significant impact on the overall architecture of the system.

The contract provides a base set of behavioral and performance requirements; this document reflects approved updates to those requirements. One of the architecturally significant performance requirements is to update users' screens within three (3) seconds of receiving data. To meet this performance requirement, ArcGIS functionality that allows the addition of attributes (such as colors of polygons and appearance of icons) will be separated from the generation image tiles displayed on the user interface to represent a map. The ArcGIS Software Development Kit will be used to display the image tiles as a map, and overlay attributes derived from user layer selection and real-time data updates.

A pipeline architecture has been selected to ensure that real-time updates are consistently available to both the business services and the data services that provide updates to the user interfaces.

To facilitate performance, especially in data updates, it is anticipated that the data services will provide a "publish-subscribe" mechanism for access to streaming data.

3.2.6 User View of Data

The user interface for the R-ICMS will be accessed via a browser. Angular has been chosen as the framework with which the user interface will be implemented. The R-ICMS user interface will be:

- **Interactive:** Facilitate user understanding and actions needed to accomplish the primary purposes of the initial version of the R-ICMS
- **Map-based:** Facilitate understanding of the current traffic network situation

The R-ICMS will provide several types of user interfaces including:

- **Traffic Network Conditions:** User interfaces will be provided to facilitate understanding of specific aspects of the traffic network conditions.
- **Response Plan Management:** User Interfaces will be provided to allow users to evaluate, select, approve, activate, and terminate response plans,
- **User & System Configuration:** User Interfaces will be provided to configure the system and users.
- **Signal Time Updates:** User interfaces will be provided to request periodic and on-demand generation, modification, evaluation, and signing of signal timing plans

3.2.7 Safety, Security, and Privacy

The R-ICMS will be integrated into the security environment of FDOT D5. R-ICMS, so users will be required to have active accounts in the FDOT D5 domain or in a domain trusted by the FDOT D5 domain. Those accounts will be created, managed, and terminated by standard FDOT D5 processes.

Fine-grained authorization of user capabilities will be controlled within the R-ICMS. Administrative R-ICMS users will be provided the capability to control which data sources users can access, which actions users can perform, and which response plan elements users can approve based on the devices contained in the response plan element. Permissions for access to data sources and system capabilities can be provided to users by assigning users to specific roles. Response plan element approval capability can be provided to users by assigning users to agencies which have the relevant element assignments.

Access to the R-ICMS from outside the FDOT network will be provided through secured communications channels (System Requirement 3.1). It is anticipated that these secure channels will be based on Secure Sockets Layer (SSL) to the browser.

3.2.8 Third Party Components

On the following pages, Table 3 – Third Party Components contains the initial list of known third-party components intended for use as part of the R-ICMS. Other libraries and third-party components, identified later, will be addressed in the appropriate 90% design documentation.

Table 3 – Third Party Components

Component	License	Type	Features	Reason Chosen
HCS7 (7.8.5)	Proprietary	SOT	Optimization of corridors	A decision was made during the contract negotiations to utilize the capabilities of HCS7, a signal timing plan optimization tool-set developed by the University of Florida, as the basis of the Signal Optimization Tool (SOT) that is an integral part of the initial version of the R-ICMS. HCS7 generates proposed signal timing plans for intersections and for corridors. The optimization is based on using intersection and corridor parameters combined with calculations based on the federal Highway Capacity Manual and a genetic approach to optimization. Investigation of the match between user needs, the capabilities of HCS7, and the ability of HCS7 to be integrated into the R-ICMS is still being conducted. However, it is anticipated that much of the base capabilities of the SOT will be provided by the HCS7 tool-set. Additional user needs not met by the basic HCS7 tool-set will be provided by the portions of the R-ICMS wrapping the HCS7 tool-set.
Aimsun	Proprietary	Traffic Simulation	Online, real-time, mesoscopic traffic simulation	A traffic simulation engine is selected by FDOT.
ElastAlert	Apache 2.0	Alerting	Monitoring and alerting based on data in Elasticsearch	ElastAlert is used to watch for various conditions in Elasticsearch that are of interest for monitoring the health of the R-ICMS system and notifying an operator if something appears out of the ordinary. This helps catch any anomalies in the system early so they can be addressed.
NSwag	<u>MIT</u>	Code generator / Library	OpenAPI code generator for ASP.NET Core and TypeScript	Specifying internal and external Application Program Interfaces (API) in terms of specifications and then using code generators to create clients and server stubs greatly reduces the time spent implementing and debugging wire protocols. NSwag supports generating both ASP.NET Core and TypeScript bindings from a common specification, allowing the front-end and back-end to easily communicate with each other.
Logstash	<u>Apache 2.0 / Elastic</u>	Data flow	ingestor for Elasticsearch	Logstash can take data from many sources, transform and filter it, and pass it on to many other services - the primary one being Elasticsearch. Logstash was chosen for its integration with Elasticsearch.
Beats	<u>Apache 2.0 / Elastic</u>	Data flow	Data and metrics shipper	Beats are small applications that collect and send data over the network to various services. It's lighter-weight than Logstash and they are often used in conjunction. Beats would allow us to get logs and basic system metrics from systems without the memory and processor overhead of Logstash on machines that don't require the full capabilities of Logstash.

System Design Document for Regional Integrated Corridor Management System (R-ICMS)

Component	License	Type	Features	Reason Chosen
Kafka	Apache 2.0	Data flow	Distributed, persistent, streaming message queue	Kafka is a messaging system that works to ensure every message sent to it is received by its intended recipients by saving events to disk and distributing them among multiple instances for high availability. It is used for fanning out events to multiple services and will be the primary carrier of data from the drivers to the rest of the system. The intent is to use the Cloudera distribution of Kafka.
Redis	3-Clause BSD	Data store	In-memory key-value data structure store	Redis is a widely used, stable, fast key-value store often used to cache shared application state. In this case, the intent is to use it to store authorization data. Storing this data in Redis allows all requests for authorization to use an always up service with a well-defined API to do so instead of relying on a hand-rolled service with a custom API that will likely be updated more frequently. Additionally, Redis supports replication for high availability.
MongoDB	AGPL3 or proprietary	Data store	NoSQL database	A popular data store for document-oriented, semi-structured information. It is a high-performance data store that supports sharing and replication.
ElasticSearch	Apache 2.0 / Elastic	Data store	search engine	ElasticSearch is a tool for indexing and querying text-heavy information. It was selected it because it supports semi-structured data such as logs, high availability with automatic recovery, and has a suite of tools for ingestion and visualization of data.
HDFS (part of Hadoop)	Apache 2.0	Data store	Distributed File System	The Hadoop Distributed File System (HDFS) is a scalable, distributed file system. All input data and some generated data, such as signal optimization runs, will be stored in HDFS. This will give future users of the system all the raw data that the system used to perform analytics and big data analyses. HDFS allows adding nodes to a running cluster, allowing the storage to scale without requiring the cluster to be restarted. The Cloudera distribution of Hadoop will be used, which includes HDFS.
Microsoft SQL Server	proprietary	Data store	Relational Database System	Microsoft SQL (MS SQL) Server will be used for smaller volume, highly relational data. MS SQL Server will be used for SOT operation and inputs, permissions, agency and device configuration and defaults, a possible back-end for the ArcGIS database, managed events, and response plans. FDOT will provide a Microsoft SQL Server instance.

System Design Document for Regional Integrated Corridor Management System (R-ICMS)

Component	License	Type	Features	Reason Chosen
<u>ESRI ArcGIS Enterprise Advanced xxx.xxx.xxx.xxx and ESRI GeoEventServer xxx.xxx.xxx.xxx</u>	<u>proprietary, one time, or subscription</u>	Data store	Geographic data store	Much of the data for the R-ICMS project has locations and geographical shapes associated with it, so the data will be stored in a data store with mapping and first-class support for queries on these properties. ESRO ArcGIS Enterprise Advanced xxx.xxx.xxx.xxx and ESRI GeoEvent Server xxx.xxx.xxx.xxx was chosen to enable the use of the GeoEvent server for real-time streaming geographic information.
<u>Kubernetes</u>	<u>Apache 2.0</u>	Deployment	Container orchestration for deployment, load balancing, hot failover, canary and rolling deployments	A container orchestrator handles scheduling and running containers across multiple computers and routing traffic among them. This simplifies management of applications running on clusters of computers. It works to ensure that applications stay running and will automatically restart them (on different machines if needed). It also supports rolling deployments to ensure uptime, rolling back deployments to previous states, and partial deployments to test stability of new code before doing a full roll out (canary deployments). These are all desirable features in a system that values high uptime. Kubernetes is a community developed system created by former Google engineers. Additionally, major cloud providers (Amazon Web Services (AWS), Google Cloud, Azure) all support Kubernetes clusters as a paid service.
<u>ASP.NET Core</u>	<u>Apache 2.0</u>	Framework	Web framework for .NET Core	ASP.NET Core is the next version of the ASP.NET MVC framework. It runs on both .NET Core and .NET Framework. It boasts a more modular design than its predecessors and is actively developed.
<u>Angular</u>	<u>MIT</u>	Framework	Front-end development framework	Angular provides ways to build modular interfaces in the browser and encourages efficient and extensible data flow patterns within an application. It is popular and maintained by Google.
<u>System.IdentityModel.Tokens.Jwt</u>	<u>MIT</u>	Library	.NET library for creating and validating JavaScript Object Notation (JSON) web tokens	JSON Web Tokens (JWTs) will be used to store user authorizations. JWTs support storing data and are cryptographically signed, making them tamper-proof and safe to send to a user's device. JWTs also have standard fields for usernames and expiration dates.
<u>Serilog</u>	<u>Apache 2.0</u>	Library	Structured logging library for .NET Core	Logging is essential for debugging and metrics. Serilog supports multiple log levels and data sinks including console and files. It supports logging in JSON formats (that's the "structured" in "structured logging"), which simplifies ingestion via tools like Logstash.
<u>StackExchange.Redis</u>	<u>MIT</u>	Library	.NET library for interacting with Redis	This library is used to access Redis from a .NET application. It is maintained by Stack Exchange, the company that runs many large question and answer sites, including Stack Overflow, a well-known developer resource.

System Design Document for Regional Integrated Corridor Management System (R-ICMS)

Component	License	Type	Features	Reason Chosen
<u>MongoDB C# driver</u>	<u>Apache 2.0</u>	Library	.NET Core library for interacting with MongoDB	This library is used to access MongoDB from .NET Framework and .NET Core applications. We chose this driver because it is officially maintained and supported by MongoDB.
<u>confluent-kafka-dotnet</u>	<u>Apache 2.0</u>	Library	.NET library for interacting with Kafka	This library is for interfacing with Kafka from .NET applications and supports all basic Kafka operations. This was selected because it's the most up to date and maintained library for interfacing Kafka with .NET.
<u>SUSE Linux Enterprise Server (SLES)</u>	GNU General Public License and various	Operating system	Linux-based Operating System	SLES is one of the supported operating systems of Cloudera that has an option for paid support.
<u>Windows Server</u>	<u>proprietary</u>	Operating system	Windows operating system	Some of the software selected only runs on Windows. The primary one is HCS7. In order to run this software, a Windows server is required.
<u>Docker</u>	<u>Apache 2.0</u>	Packaging / Deployment	Container runtime for bundling and codifying application dependencies and running them in repeatable, isolated environments	Using containers: <ul style="list-style-type: none"> • simplifies dependency management and deployment, as only the Docker runtime needs to be installed to run the applications • System setup and configuration management are easier which helps to feel confident that the application will behave the same in multiple environments (development, testing, and production) • Ensures that there aren't missing dependencies required to run applications and documents the creation of these environments in a source control friendly way • Containers are also used industry-wide; as an example, the major public clouds - Amazon Web Services (AWS), Google Cloud, and Microsoft Azure - all support running containers as a paid service.
<u>Cloudera</u>	<u>proprietary subscription</u>	Platform	Data warehousing software and support	Cloudera provides a managed and supported data warehousing and analytics platform based on Kafka and Hadoop with additional management features. Cloudera was chosen for its ease of administration and support.
<u>.NET Core</u>	<u>MIT</u>	Runtime	.NET runtime	.NET Core is an open source runtime from Microsoft for .NET applications that runs on Windows, Linux, and MacOS. It is different than .NET Framework, which is Windows only. (Both implement the .NET Standard; Framework includes additional Windows-specific libraries.) .NET Core was chosen because it is stable, developed by Microsoft, is open source and cross-platform.
<u>Kibana</u>	<u>Apache 2.0 / Elastic</u>	Visualization	dashboard and analytics on top of ElasticSearch	Kibana is a tool for querying and visualizing aggregations of data from ElasticSearch. Kibana was selected it for its integration with ElasticSearch.

System Design Document for Regional Integrated Corridor Management System (R-ICMS)

Component	License	Type	Features	Reason Chosen
<u>Nginx</u>	<u>2-Clause BSD</u>	Web server	Web server for gateway for Transport Layer Security (TLS) termination and request routing/proxying	Nginx is a widely deployed web server that supports TLS and reverse proxying of applications behind different routes. This will be the gateway through which all requests into the system pass.
<u>Java</u>	<u>Java 8</u>	Runtime	Java runtime	Java is a general-purpose, concurrent, object-oriented, class-based, and the runtime environment (JRE) which consists of JVM which is the cornerstone of the Java platform.
<u>Ubuntu 18.04</u>	GNU General Public License and various	Operating system	Linux-based Operating System	Wide community support. Options for paid support. Long term service security releases. Supported by Cloudera.
<u>Power BI Report Server</u>	Premium	Report Server	On-Premises or hosted Cloud	Included in Power BI Premium, On-premise source data, ability to embed reports into the application, lack of data refresh constraints

3.3 System Architecture

As described in the original invitation to negotiate, the R-ICMS consists of three subsystems:

- The DFE provides a “big data” environment for the ingestion, collection transformation, storage and publishing of the data
- The DSS provides the business layer for the R-ICMS
- The IEN provides the user interface for the R-ICMS

The design architecture of the R-ICMS has been broken down into seven layers: drivers, pipelines, data stores, data services, business services, and user interfaces.

- Data Fusion Engine is comprised of four layers
 - Driver layer component names end in “-DR”. The Driver layer implements interfaces to external systems. These interfaces may be inputs or outputs.
 - Pipeline layer component names end in “-PL”. The Pipeline layer receives data from a producer and distributes the data to consumers. The producers are often drivers and the consumers are often persistent storage, applications that require real time data, and less commonly external systems that require real time data.
 - Data Store layer components end in “-ST”. The Data Store layer implements data persistence. The current design uses HDFS, MongoDB, MS SQL Server and Elastic to implement persistence storage.
 - Data Service layer components end in “-DS”. The Data Service Layer provides data to other applications usually from the Data Store layer. This differs from the Pipeline layer that distributes real time data.
- Decision Support System Component is comprised of a single layer
 - Business Service Layer components end in “-BS”. Business services are components that provide functionality to users. These components allow a user (human or system) to act on the data and change the state or content of the system.
- Information Exchange Network is comprised of a single layer
 - User Interface Layer components end in “-UI”. The User Interface layer implements a map for display of traffic conditions within the system, dialogs for interacting with traffic events especially response plans, dialogs for interaction with the Signal Optimization Tool (SOT), and configuration of the system (e.g. users, response plans, etc.)
- Common Core is an additional collection of components that are utilized throughout the architecture.
 - Common Core components end in “-CC”.

Figure 2 provides a color legend that is used throughout this Software Design Document. It is followed by a textual description of those colors. Figure 3 illustrates the overall system architecture at a high level using the colors of Figure 2.



Figure 2 – System Architecture Layer Color Legend

- The layers and components shown in green, i.e., the Drivers, Pipelines, and Data Stores, are expected to reside in the “Big Data” environment.
- The layers and components shown in purple, i.e., the Data Services and the Business Services, are expected to reside in the standard FDOT services physical servers.
- The common components shown in blue are also expected to primarily reside in the standard FDOT services physical servers.
- The components shown in brown are expected to reside in the user’s systems.

The following subsections list and describe the components identified for inclusion in the R-ICMS.

System Design Document for Regional Integrated Corridor Management System (R-ICMS)

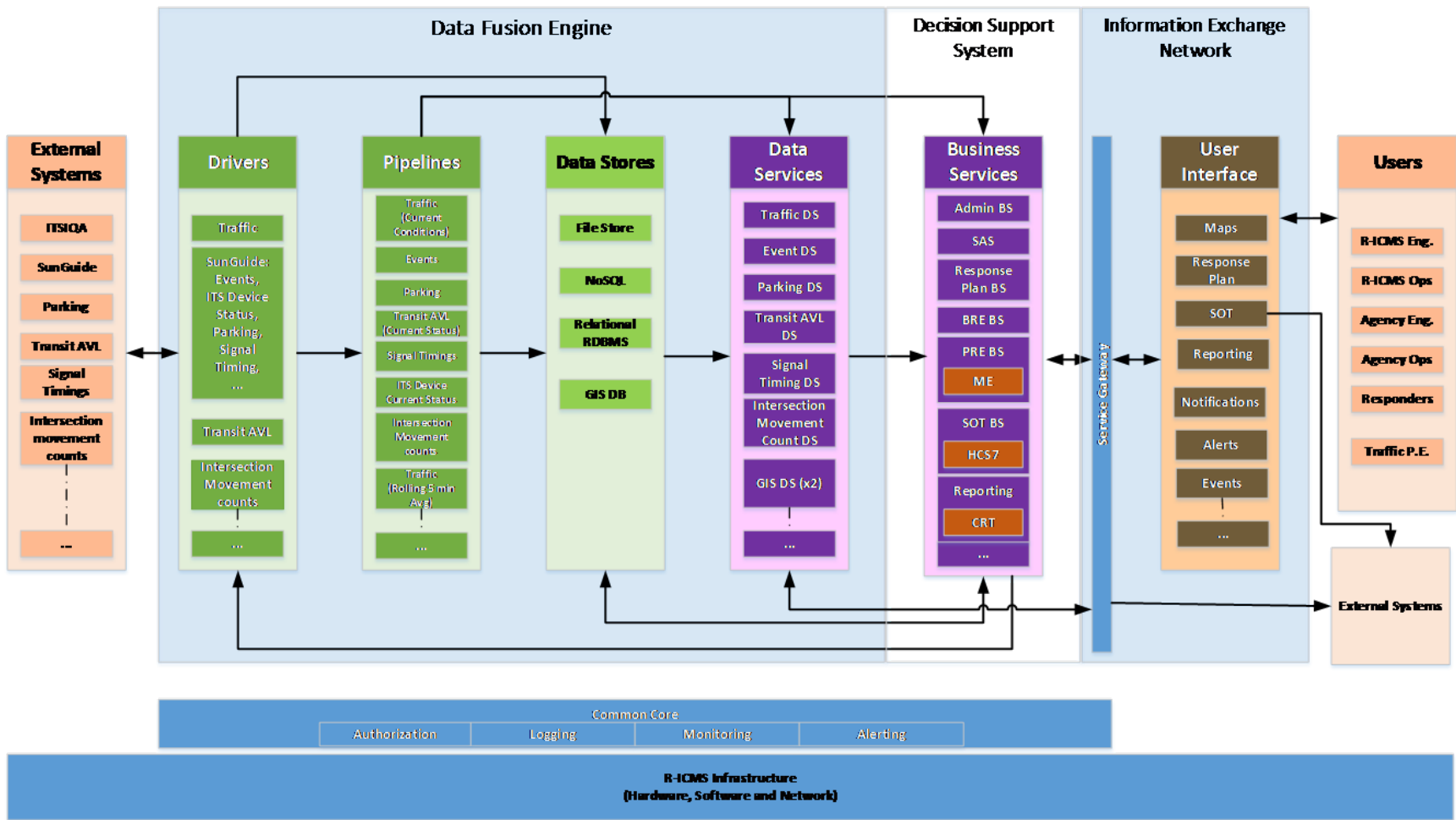


Figure 3 – R-ICMS High-Level Diagram

Source systems are shown on the left. Updates to input data streams will be ingested through a set of drivers within the DFE. Those drivers will be responsible for maintaining and reporting the status of the link with the external system. R-ICMS will provide a common set of functions to support monitoring and logging of status information throughout the system. The drivers will also be used by the R-ICMS to send messages to SunGuide to request modifications to external systems as a part of the activation of response plans.

Streaming data will pass from the drivers into the pipeline layer of the architecture. The pipeline layer will be used to facilitate rapid throughput of streaming updates throughout the system. Streaming data pipelines will be consumed by data services, as well as the business services. The data services will publish the data to consumers who have been authenticated, authorized, and who have subscribed to the data. Business services will consume the data from the pipelines to make the real-time decisions needed to respond to events and evaluate the status of the traffic network. The streaming data pipelines will also be consumed by the data stores, so the incoming data is stored for later use and analysis.

Non-streaming data, otherwise known as batch, static or, more appropriately, slowly changing data, will not be entered through the pipelines. Non-streaming data will be entered into the system using processes executed by a system administrator. Where the data is available in files with standard formats and available via a standard protocol such as File Transport Protocol / Secure File Transport Protocol (FTP/SFTP), a standardized generated script process will be used to retrieve the data. Where the data can be acquired through standardized interfaces, those interfaces will be used. However, non-streaming data sources that do not have standard interfaces for collecting the data will require additional customization. Once the data has been collected, it will be placed into the data stores using scripts. The data services will access the data stores to provide data to business services.

Data will be stored in different types of data stores depending on its user, volume, age, and nature. Response plan elements, response plans, user actions, system recommendations, signal timing plans, and other transactional data will be stored in a MS SQL database. Streaming data, modeling results, and other large, complex sets of data will be stored in NoSQL databases such as HDFS, Mongo, or ElasticSearch. Data to be used to display maps and the data on maps will be stored in an ArcGIS Database.

3.4 System Components

The following sections describe the design of the system organized by subsystems.

3.4.1 DFE Layers and Components

Four of the R-ICMS architecture layers consist primarily of functionality attributed to the DFE. The DFE layers include:

- **Drivers** handle the connections to external systems. They collect or receive data from external systems and send data to external systems.

System Design Document for Regional Integrated Corridor Management System (R-ICMS)

- **Pipelines** provides access to and transformation of streaming data being ingested from external systems
- **Data Stores** provides persistence of data from both internal and external sources
- **Data Services** provides authorized users filtered access to historical data and real-time updates by subscription

The following is the typical data flow per the R-ICMS system architecture:

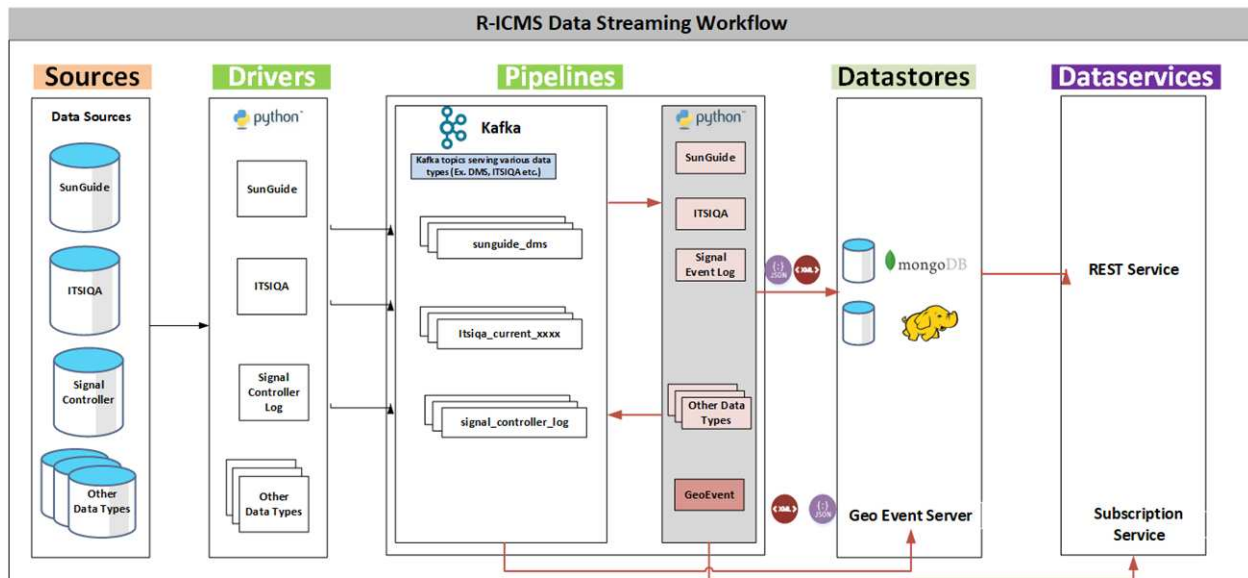


Figure 4 – R-ICMS Data Streaming Workflow

Several specific, common big data design patterns will be used in the design and implementation of the DFE including the following:

- **Multisource Extractor Pattern:** for extracting multiple data source types in an efficient manner
- **Multi-destination Pattern:** to transport data to multiple components like HDFS, GIS, IN-memory data service (for real-time), NoSQL
- **Real-Time Streaming Pattern:** for real-time streaming of data to the web user interface
- **Just-in-Time Transformation Pattern:** to upload large files (GIS Shape files)/large quantities of data in batch mode or using traditional Extract, Transform, Load (ETL) methods only when required to save compute time¹

¹ These patterns are common in big-data environments; they are described in the article that may be found at http://guttitech.com/phpfusion/faq.php?cat_id=7

3.4.1.1 Drivers

The following components are assigned to the Driver layer of the architecture. Drivers are responsible for initiating, monitoring, and maintaining links to external systems. The drivers will be responsible for adding the “Received Time” to data received by the R-ICMS. If a connection to an external system is lost, the driver is responsible for reconnecting to the external system and, where feasible, requesting data missed while the connection was down. The drivers are also responsible for informing the monitoring service of connection status.

3.4.1.1.1 Driver Structure

Data ingestion is the process of extracting data from external systems and loading the data into a data warehouse environment (aka ETL). The Driver performs the first step in ingestion. The most common steps involved in the ingestion phase are:

1. Establish a connection to the external data source (web service, FTP site, SQL database etc.)
2. Once the connection is successful, the process will make a request for the intended data. The method will vary depending on the source type (API Call, FTP get request, SQL query etc.)
3. The data will get transferred to the processing system where the process will make additional inspection/adjustments/processing to make it more meaningful based on the intended target system usage.
 - a. **Identification:** Identification of various data source formats (identified during the design phase) will be handled at this step.
 - b. **Filtration:** Remove any data that is not required. Sometimes, the sources provide more data than is required for the application and the extra information needs to be filtered at the very first level.
 - c. **Validation:** Identification and removal of any ill-formatted data which will cause exceptions downstream for the applications will be handled in this step. This will be done by validating the data values and data types coming from the source.
 - d. **Transformation:** This step will be used when the external source data format is not the intended format for the destination. Transformation will happen at the entry level to suit the intended destinations. Also, same data will be structured in different ways depending on the needs of the R-ICMS application.

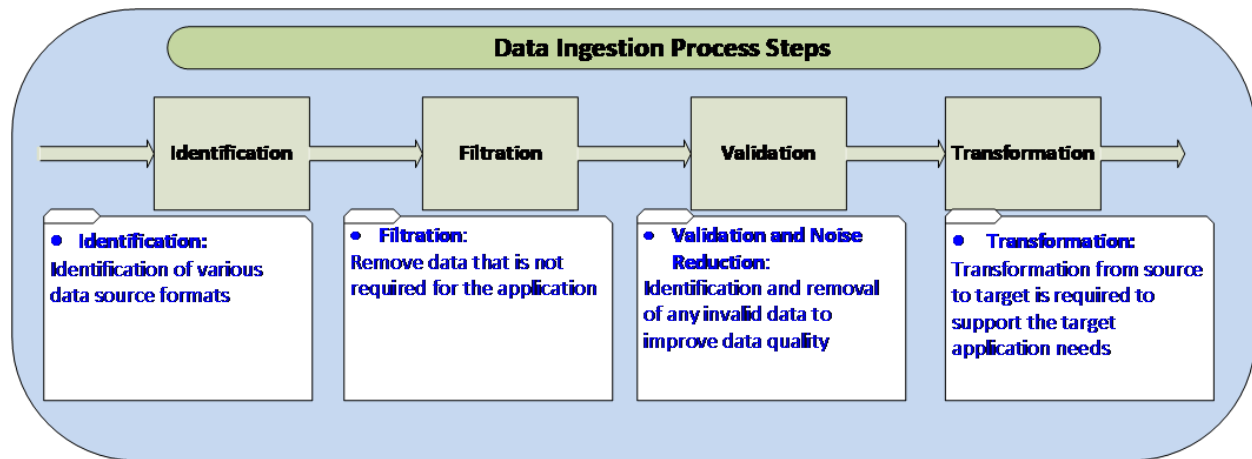


Figure 5 – Ingestion Flow

4. When data is received/retrieved and when it is passed to other layers, a log entry is generated using the common logging service including begin/end timestamps, elapsed time, status of the process, and data size/volume. This information is needed for picking up from where the previous processes left off, as well as providing vital statistics about the data processed and resource usage.
5. The process will also incorporate retry/recovery options when exceptions are encountered and will log such incidents through the logging mechanism.
6. All these process steps will be encapsulated into script(s), and the script execution will be automated using a job scheduler at pre-defined intervals depending on the data availability frequency.

In addition to the above steps, a separate process will be established to send notifications to the configured users about job failure. The process will be driven by a configuration mechanism mapping the data load jobs and the users to be notified. Metadata with the job information will be stored in the NoSQL to help map notifications with the intended users.

3.4.1.1.1.1 Configuration Based Design

The R-ICMS data ingestion process will adopt the configuration-based approach which will facilitate much simpler, reusable, efficient and better maintainable code which follows the suite of best practices methodology. This approach encapsulates defining the required configuration parameters to accomplish the entire process of fetching data, storing and distributing to other processes. Each individual data source will have a manually entered and maintained configuration file consisting of parameters for various functions (e.g., source location, availability method, and credentials required, HDFS storage location, NoSQL storage location, etc.). The main advantages of this approach is minimal code changes while parameters can be altered without impacting the code, which also results in less code builds and deployments. This approach is dependent on the following items:

- Define one configuration per individual data source with all necessary parameters required.
- Store the configuration in files.

- Build a common utility script to fetch the configuration for the data source.
- Create initial configuration
- Build scripts (enter them into source code control system).

3.4.1.1.1.2 Ingestion Methods

R-ICMS ingestion methods can be classified as either API, TCP socket (e.g. SunGuide Databus), or SMB. The process for ingesting data for these methods follows the following steps:

1. The driver will run in a Docker container deployed using Kubernetes
2. The driver will use the configuration information provided in the Kubernetes manifest file during the Kubernetes deployment to access the data source
3. The driver will retrieve the data from the data source
4. The driver will either load the data into the data store or will publish it to the pipeline
5. The driver will call the Common Core log function to log relevant information
6. The driver will call the Common Core notification function to send error notifications

The driver will generate a log each time a data ingestion process is executed. Log entries will contain the following parameters:

- Data source and data type/sub-set (build a list of all the data sources and sub-set data types and make it as part of metadata)
- Process name (reference the process name as part of the metadata referenced above)
- Process execution date and timestamp
- Process begin timestamp, end timestamp and time elapsed
- Size of the data transferred
- Number of records processed where applicable
- Process status (success/failure)
- Errors encountered during execution (process exceptions)
- Errors received from the source when requesting data

These parameters shall provide the base for system performance analytics.

3.4.1.1.1.2.1 API / Web Service Calls

Another common type of data ingestion process is through API calls. When the data is made available through web services/API calls, the ingestion will adopt this method to fetch data from the external systems. This section describes the common applicable methodology, but the process will be different for each individual data source depending on the type/format of the data. A common utility will be used for making API calls and will wrap the functionality for each specific data source to process and distribute the data accordingly. In addition, more common functions can be defined depending on the context and need.

Design and create a script that can fetch configuration parameters, call the API, and return the content of the data received. The script will be a common function and can be used anywhere when data needs to be brought in through API calls.

Once the data is fetched from the source, the following steps will take place:

1. Validate the data for any missing data elements and add them if necessary (ex. update date and timestamp)
2. Publish the data to Kafka topic
3. When there are exceptions that the data doesn't need to go through the pipeline, the process for that individual dataset will extend to handle further processing and storage
4. Call the Common Core logging function to log the process

3.4.1.1.1.2.2 SunGuide Databus

The SunGuide Databus drivers will communicate with the SunGuide Databus process to request and receive data related to DMS, Events, CCTV, etc. Communication will occur over a single TCP socket connection with the SunGuide Databus subsystem. Configuration data for specific data types (such as DMS) will be requested and subscribed from the representative subsystem through communication with Databus. Status data will be collected directly from Databus via status update messages. To receive data from the Databus, the R-ICMS is responsible for executing the following series of steps:

1. Initiate a socket connection to SunGuide Databus
2. Authenticate to the appropriate subsystems (DMS, CCTV, etc.)
3. Subscribe to the sub-system to receive configuration data
4. Retrieve data from appropriate subsystems to get current configuration and status information
5. Subscribe to Databus to receive status updates for appropriate subsystems
6. From that point onwards, any updates from SunGuide will be pushed to the client which will then publish the updates to a Kafka topic
7. Detect and log when connection persists but no data is received
8. SunGuide sends a notification when a subsystem disconnects. Subsequently, it sends a notification when a subsystem re-connects

Below are the steps describing the ingestion process implementation in general. There might be additional steps specific to individual data types and are documented with the respective data type ingestion process where applicable.

1. The driver handles all connectivity to the SunGuide Databus. This includes TCP/IP socket connectivity, authentication, subsystem disconnects and re-connect
2. The driver sends requests to SunGuide and receives responses
3. The driver publishes the response data to the appropriate Kafka topics
4. The driver uses Common Core to log to a centralized log database

When the external system makes files available via a shared folder on Microsoft Windows platform the SMB protocol must be used to ingest the files. The following describes the common steps for any driver utilizing the SMB protocol.

- Initiate connection to the remote share using an SMB client API/library
- Identify the files to be retrieved. Each driver may use different criteria used to identify the files.
- Retrieve the files
- Perform any post-processing of the retrieved files as needed before publishing to Kafka.
- Publish the post-processed files to a Kafka topic with any added metadata (e.g., received date/time, compression type) as message headers
- Call the Common Core logging function to log the process

3.4.1.1.2 ITSQA Traffic Conditions Drivers

The ITSQA Traffic Conditions Drivers ingest data from ITSQA. The following ITSQA data sets are available through HTTP GET calls to the ITSQA system:

- **LinkConfig:** Configuration information for ITSQA's master link configuration, including a list of links and their locations
- **TrafficData:** Primary link data output, providing speed, volume, occupancy, travel time, and quality values of each link
- **LaneTrafficData:** Detailed link data output, providing speed, volume, occupancy, travel time, and quality values of each link. Also includes speed, volume, occupancy, and travel time information for each lane for each link that contains lane-level data
- **ClassData:** Vehicle classification data reported on a per link basis
- **TMConfig:** Configuration information for ITSQA's master intersection configuration, including a list of intersections, approaches, and lanes
- **TMData:** Intersection movement count data reported on a per lane per approach per intersection basis

ITSQA produces two versions of the data feed; one version contains only data that is publicly releasable excluding 3rd party data that is restricted by license agreement for public release and another version that includes all data for use by FDOT internally. RICMS will ingest both versions of the data. Furthermore, ITSQA provides two states, one that provides current data and the other that provides access to a data archive which contains data for a configurable duration (at this time one week).

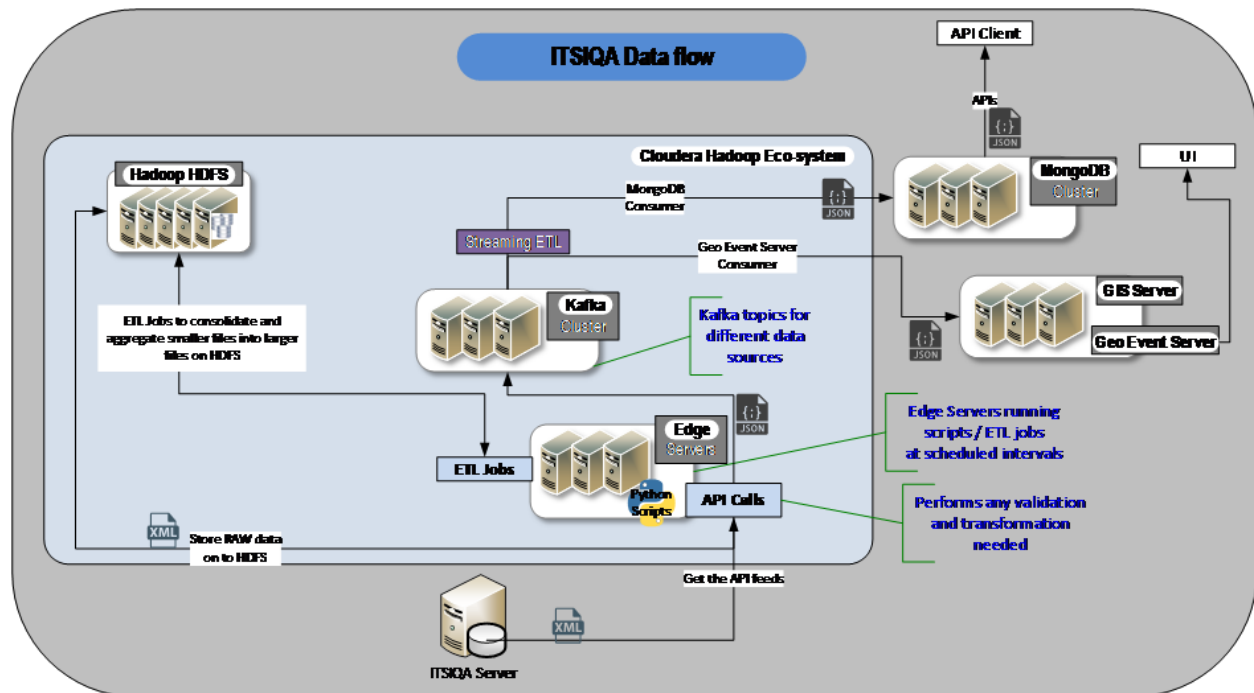


Figure 8 – ITSIQA Data Flow

3.4.1.1.2.1 ITSIQA Current Traffic Conditions Driver (TRF-CTC-DR)

The ITSIQA current traffic conditions drivers (TRF-CTC-DR) will retrieve ITSIQA current traffic conditions data using ITSIQA’s SMB interface and publish it to a pipeline (Kafka topic). ITSIQA provides two versions of six datasets and each version will have its own driver.

- There will be two sets of drivers deployed; one that includes only un-restricted use data, and one for that contains all data.
- Each set of drivers will include a driver for each of the different files in the ITSIQA file set.

The drivers will poll for new data at a frequency that is greater than the frequency at which the source data is being published². The typical process steps to get the data from ITSIQA real-time stream into R-ICMS are as follows:

1. The driver will read the required configuration from the runtime configuration file. The configuration file includes the source ITSIQA SMB connection details and the Kafka topic name
2. It will make the SMB retrieve file call and will receive the file
3. The driver will parse the XML file to extract the TimeStamp attribute from the root element and compare the TimeStamp value to the last processed file’s timestamp. The driver will only publish the retrieved file to Kafka if the timestamp has changed.

² The frequency will be configurable and will be adjusted in production to account for the drift and jitter of the clocks on two disparate systems.

System Design Document for Regional Integrated Corridor Management System (R-ICMS)

4. Publish the received data to appropriate pipeline (i.e. Kafka topic) and include a message header that indicates the date/time when the data was ingested
5. A common notification library will interact with the NOT-BS, allowing a service to create both alerts and notifications.
6. All data receiving events will be logged into the Common Core Log System with details about the time, and data being received

Table 4 – ITSQA Current Traffic Conditions Driver – Ingestion Parameters

Data Set Name	Attribute	Value
LinkConfig	UNC Path	\\xxx.xxx.xxx.xxx\itsqa\Web\LinkConfig-ITSQA-AllSources.xml \\xxx.xxx.xxx.xxx\itsqa\Web\LinkConfig-ITSQA-NoHERE.xml
	Target Location	Kafka topics: itsqa_all_sources_link_config_current itsqa_no_here_link_config_current
	Frequency	Daily
TrafficData	UNC Path	\\xxx.xxx.xxx.xxx\itsqa\Web\TrafficData-ITSQA-AllSources.xml \\xxx.xxx.xxx.xxx\itsqa\Web\TrafficData-ITSQA-NoHERE.xml
	Target Location	Kafka topics: itsqa_all_sources_traffic_data_current itsqa_no_here_link_traffic_data_current
	Frequency	60 seconds
ClassData	UNC Path	\\xxx.xxx.xxx.xxx\itsqa\Web\ClassData-ITSQA-AllSources.xml \\xxx.xxx.xxx.xxx\itsqa\Web\ClassData-ITSQA-NoHERE.xml
	Target Location	Kafka topics: itsqa_all_sources_class_data_current itsqa_no_here_class_data_current
	Frequency	60 seconds
LaneTrafficData	UNC Path	\\xxx.xxx.xxx.xxx\itsqa\Web\LaneTrafficData-ITSQA-AllSources.xml \\xxx.xxx.xxx.xxx\itsqa\Web\LaneTrafficData-ITSQA-NoHERE.xml
	Target Location	Kafka topics: itsqa_all_sources_lane_traffic_data_current itsqa_no_here_lane_traffic_data_current
	Frequency	60 seconds
TMCCConfig	UNC Path	\\xxx.xxx.xxx.xxx\itsqa\Web\TMCCConfig-ITSQA-AllSources.xml \\xxx.xxx.xxx.xxx\itsqa\Web\TMCCConfig-ITSQA-NoHERE.xml
	Target Location	Kafka topics: itsqa_all_sources_tmc_config_current itsqa_no_here_tmc_config_current
	Frequency	Daily
TMCDData	UNC Path	\\xxx.xxx.xxx.xxx\itsqa\Web\TMCDData-ITSQA-AllSources.xml \\xxx.xxx.xxx.xxx\itsqa\Web\TMCDData-ITSQA-NoHERE.xml
	Target Location	Kafka topics: itsqa_all_sources_tmc_data_current itsqa_no_here_tmc_data_current

System Design Document for Regional Integrated Corridor Management System (R-ICMS)

Data Set Name	Attribute	Value
	Frequency	60 seconds
Common Attributes	Source Data Format	XML
	ICD Location	TBD (pending URL from FDOT)
	Target Data Format	XML (gzip compressed)
	Data Load Mechanism	Server Message Block (SMB)

3.4.1.1.2.2 ITSQA Archive Traffic Conditions (TRF-ATC-DR)

The ITSQA archive data drivers will retrieve data using SMB and publish it to a Kafka topic. The benefit of using an archive data set as the source for loading into data at rest targets is that it will cover any missing gaps of the data and there will be no need for a second process to find data gaps and get the missing data. ITSQA provides two versions of six data sets and each version will have its own driver. A separate process is required for fetching each type of data set at an interval that is smaller than the Source Data Update Frequency. The process steps to get the data from ITSQA archive data stream into R-ICMS are as follows:

1. The driver will read the required configuration from the runtime configuration file. The configuration file includes the source ITSQA SMB connection details and Kafka topic name
2. The driver will read the reference of the last file processed from (evaluated in the following order) the Kafka topic or the MongoDB collection and will make SMB calls to retrieve the list of files published to the archive since the last file processed
3. The ITSQA archive traffic conditions driver will retrieve all files that have been written to the archive since the last time the driver had run.
4. Publish the received data to a Kafka topic and include a message header that indicates the date/time when the data was ingested
5. A common notification library will interact with the NOT-BS, allowing a service to create both alerts and notifications.
6. All data receiving events will be logged into the Common Core Log System with details about the time, and data being received

Table 5 – ITSQA Archived Traffic Conditions Driver – Ingestion Parameters

Data Set Name	Attribute	Value
LinkConfig	UNC Path	\\xxx.xxx.xxx.xxx\itsqa\Web\Archive\LinkConfig\LinkConfig-ITSQA-AllSources-<YYYY-MM-DD>.xml \\xxx.xxx.xxx.xxx\itsqa\Web\Archive\LinkConfig\LinkConfig-ITSQA-NoHERE-<YYYY-MM-DD>.xml
	Target Location	Kafka topics: itsqa_all_sources_link_config_archive itsqa_no_here_link_config_archive
	Frequency	Daily
TrafficData	UNC Path	\\xxx.xxx.xxx.xxx\itsqa\Web\Archive\TrafficData\ITSQA-AllSources\<YYYY-MM-DD>\TrafficData-ITSQA-AllSources-<YYYY-MM-DD>-<hhmm>.xml \\xxx.xxx.xxx.xxx\itsqa\Web\Archive\TrafficData\ITSQA-NoHERE\<YYYY-MM-DD>\TrafficData-ITSQA-NoHERE-<YYYY-MM-DD>-<hhmm>.xml
	Target Location	Kafka topics: itsqa_all_sources_traffic_data_archive itsqa_no_here_link_traffic_data_archive
	Frequency	60 seconds
ClassData	UNC Path	\\xxx.xxx.xxx.xxx\itsqa\Web\Archive\ClassificationData\ITSQA-AllSources\<YYYY-MM-DD>\ClassData-ITSQA-AllSources-<YYYY-MM-DD>-<hhmm>.xml \\xxx.xxx.xxx.xxx\itsqa\Web\Archive\ClassificationData\ITSQA-NoHERE\<YYYY-MM-DD>\ClassData-ITSQA-NoHERE-<YYYY-MM-DD>-<hhmm>.xml
	Target Location	Kafka topics: itsqa_all_sources_class_data_archive itsqa_no_here_class_data_archive
	Frequency	60 seconds
LaneTrafficData	UNC Path	\\xxx.xxx.xxx.xxx\itsqa\Web\Archive\LaneTrafficData\ITSQA-AllSources\<YYYY-MM-DD>\LaneTrafficData-ITSQA-AllSources-<YYYY-MM-DD>-<hhmm>.xml \\xxx.xxx.xxx.xxx\itsqa\Web\Archive\LaneTrafficData\ITSQA-NoHERE\<YYYY-MM-DD>\LaneTrafficData-ITSQA-NoHERE-<YYYY-MM-DD>-<hhmm>.xml
	Target Location	Kafka topics: itsqa_all_sources_lane_traffic_data_archive itsqa_no_here_lane_traffic_data_archive
	Frequency	60 seconds
TMConfig	UNC Path	\\xxx.xxx.xxx.xxx\itsqa\Web\Archive\TMConfig\TMConfig-ITSQA-AllSources-<YYYY-MM-DD>.xml \\xxx.xxx.xxx.xxx\itsqa\Web\Archive\TMConfig\TMConfig-ITSQA-NoHERE-<YYYY-MM-DD>.xml
	Target Location	Kafka topics:

System Design Document for Regional Integrated Corridor Management System (R-ICMS)

Data Set Name	Attribute	Value
		itsiq_a_all_sources_tmc_config_archive itsiq_a_no_here_tmc_config_archive
	Frequency	Daily
TMCDData	UNC Path	\\xxx.xxx.xxx.xxx\itsiqa\Web\Archive\TMCDData\ITSIQA-AllSources\<YYYY-MM-DD>\TMCDData-ITSIQA-AllSources-<YYYY-MM-DD>-<hhmm>.xml \\xxx.xxx.xxx.xxx\itsiqa\Web\Archive\TMCDData\ITSIQA-NoHERE\<YYYY-MM-DD>\TMCDData-ITSIQA-NoHERE-<YYYY-MM-DD>-<hhmm>.xml
	Target Location	Kafka topics: itsiq_a_all_sources_tmc_data_archive itsiq_a_no_here_tmc_data_archive
	Frequency	60 seconds
Common Attributes	Source Data Format	XML
	ICD Location	TBD (pending URL from FDOT)
	Target Data Format	XML (gzip compressed)
	Data Load Mechanism	Server Message Block (SMB)

3.4.1.1.3 SunGuide (SG-DR)

A SunGuide (see the SunGuide reference documents in Section 2 Reference Documents) driver will be configured to provide access to each streaming data set sourced from SunGuide. The streaming data sources received from SunGuide include:

- Events
- Cameras
- Ramp Meters
- Dynamic Message Signs
- Connected Vehicle Roadside Units
- Origin/Destination
- Traffic Signal Status
- Express Lanes
- Parking Data

The SunGuide driver will be responsible for maintaining the connection with SunGuide as well as managing the connections to each of the subsystems within SunGuide from which data will be requested. The driver will be responsible for reconnecting, authenticating, and subscribing to each subsystem when the overall SunGuide connection is lost or when any individual subsystem is started/restarted.

Multiple instances of the SunGuide driver may be used to interact with multiple SunGuide software instances. This allows real data to be extracted from the production SunGuide software system while performing development and test of data extraction from a test SunGuide software system to avoid impacting operations.

Interface Control Documents (ICD) for SunGuide drivers and other SunGuide documents are available at

<http://SunGuidesoftware.com/document-library/ICD>.

The general ICD for interacting with SunGuide is found at

http://SunGuidesoftware.com/SunGuidesoftware/document-library/ICD/6_2/SunGUIDe-General-ICD-6.2.pdf

For all SunGuide Drivers:

1. Data Load Mechanism:
 - a. The data load process will make use of a Python script implementing TCP Sockets interface with SunGuide Databus.
 - b. The client process/driver will establish the connection to the server, authenticate, subscribe to the data feed, receive initial data and subsequent updates.
 - c. The client process will also redirect the incoming data to Kafka message queue.
 - d. The data load scripts will be deployed to a Kubernetes/Docker environment (as described in the Deployment and Orchestration section) for execution.
 - e. A downstream process will read the data from Kafka topics and distribute to the designated destinations.

2. Frequency
 - a. Initial data subscription request causes the SunGuide software system to send current data for all devices or objects of the requested type to the subscriber.
 - b. Updates to the data are pushed to the subscriber when status changes or when signs are polled by SunGuide Databus.
3. Process Notifications The notification service will send the required notifications using the data captured in the logs provided by the logging service.
4. Logging All data receiving events will be logged into the NoSQL database with details about the time, and data being received.

3.4.1.1.3.1 SunGuide DMS (SG-DMS-DR)

SG-DMS-DR is the R-ICMS driver that retrieves DMS configuration and status information from SunGuide. The configuration details are used for building a map layer, the status data for those DMS devices will show the current message and status displayed at that time on the map.

- The generalized process details for the SunGuide data ingestion are documented under section 3.4.1.1.2.2 SunGuide Databus.
- When a request is made to SunGuide for the data stream for the first time, it will provide the complete current state of the DMS devices including the configuration and current status.
- The subsequent updates are pushed to the client process within the subscription mechanism.
- The following describe the sequence of steps involved to get DMS data feed from SunGuide system:
 1. Request connection
 2. Authenticate request
 3. Retrieve data request from DMS subsystem (initial set of device data)
 4. Subscribe to databus – receive updates
 5. Subscribe to DMS subsystem – receive device changes

Table 6 – SunGuide DMS Ingestion

Data Set Name	Attribute	Value				
SunGuide DMS	Connection	SunGuide Server xxx.xxx.xxx.xxx Port 8089				
	Source Format	XML				
	Source ICD Locations	Source ICD	Schemas URL	~/7_1/SunGuide%207.1%20Schemas.zip		
		Schema	Folder within zip			
		common - authenticateReq	common/requests			
		common - authenticateResp	common/responses			
		dms - subscribeReq	dms/requests			
		dms – subscribeResp	dms/responses			
		dms - retrieveDataReq	dms/requests			
		dms - retrieveDataResp	dms/responses			
		databus – subscribeReq	databus/requests			
		databus – subscribeResp	databus/responses			
	databus – statusUpdateMsg	databus/messages				
	SunGuide Request / Responses	Subsystem	Request	Response	Parameters	
		dms	authenticateReq	authenticateResp	refId = refId1 icdVersion = 1.0 providerName = dms username = <username> password = <password>	
		dms	subscribeReq	subscribeResp	refId = refId1 icdVersion = 1.0 providerName = dms username = <username> securityToken = <securityToken> dmsData = true	
		dms	retrieveDataReq	retrieveDataResp		
databus		subscribeReq	subscribeResp statusUpdateMsg	refId = refId1 icdVersion = 1.0 providerName = databus dataReq = dms		
Target Format	XML (gzip compressed)					
Target Location	Kafka Topic: sunguide_dms					

3.4.1.1.3.2 SunGuide Events (SG-EV-DR)

SG-EV-DR is the R-ICMS driver that retrieves SunGuide traffic event / incident information. The generalized process details for the SunGuide data ingestion are documented under section 3.4.1.1.1.2.2 SunGuide Databus. The steps specific to SunGuide Events are documented in this section.

The source data attributes are available as part of SunGuide ICD documentation which are located at <http://SunGuidesoftware.com/document-library>

To bring data from SunGuide to the R-ICMS system, the system will deploy a SunGuide Databus driver/client process with the help of a script, which will accomplish the process referenced in section 3.4.1.1.1.2.2.

When a request is made to SunGuide for the data stream for the first time, it will provide the complete set of active events. The subsequent updates are available through the subscription mechanism from the Databus.

The following describe the sequence of steps involved to get Event data feed from SunGuide system:

1. Request connection
2. Send a status request to databus – receive status response which provides the events at that time
3. Subscribe to databus – receive updates

Table 7 – SunGuide Events Ingestion

Data Set Name	Attribute	Value			
SunGuide Events	Connection	SunGuide Server xxx.xxx.xxx.xxx Port 8089			
	Source Format	XML			
	Source ICD Locations	Schemas URL	~/7_1/SunGuide%207.1%20Schemas.zip		
		Schema	Folder within zip		
		databus - subscribeReq databus - statusReq	databus/requests		
		databus - subscribeResp databus - statusResp	databus/responses		
		databus - statusUpdateMsg	databus/messages		
	SunGuide Request / Responses	Subsystem	Request	Response	Parameters
		databus	subscribeReq statusReq	subscribeResp statusResp statusUpdateMsg	refId = refId1 icdVersion = 1.0 providerName = databus dataReq = event
	Target Format	XML (gzip compressed)			
Target Location	Kafka Topic: sunguide_event				

3.4.1.1.3.3 SunGuide Camera (SG-CCTV-DR)

Closed Circuit Television (CCTV) cameras are deployed along the roadways within District 5 and SunGuide provides an interface to maintain and interact with them. These cameras provide live video feeds to the traffic operators assisting with traffic management activities. The SunGuide has an interface to store and maintain the camera device configuration including location, network details and video feed info. In addition, SunGuide periodically polls the cameras to get updated status. This CCTV device information along with status activity is available through SunGuide's CCTV subsystem and databus provider mechanism. R-ICMS will ingest, store and make it available CCTV info to R-ICMS subsystems and external users through APIs.

The source data attributes are available as part of SunGuide ICD documentation which are located at <http://SunGuidesoftware.com/document-library>

The generalized process details for the SunGuide data ingestion are documented under section 3.4.1.1.1.2.2 SunGuide Databus. The steps specific to SunGuide CCTV are documented in this section.

The following describe the sequence of steps involved to get CCTV data feed from SunGuide system:

1. Request connection
2. Authenticate request
3. Retrieve data request from CCTV subsystem (initial set of device data)
4. Subscribe to databus – receive updates
5. Subscribe to CCTV subsystem – receive device modifications

6.

Table 8 – SunGuide CCTV Ingestion

Data Set Name	Attribute	Value			
SunGuide CCTV	Connection	SunGuide Server xxx.xxx.xxx.xxx Port 8089			
	Source Format	XML			
	Source ICD Locations	Schemas URL	~/7_1/SunGuide%207.1%20Schemas.zip		
		Schema	Folder within zip		
		common - authenticateReq	common/requests		
		common - authenticateResp	common/responses		
		cctv - subscribeReq	cctv/requests		
		cctv - subscribeResp	cctv/responses		
		cctv - retrieveDataReq	cctv/requests		
		cctv - retrieveDataResp	cctv/responses		
		databus – subscribeReq	databus/requests		
	databus – subscribeResp databus – statusUpdateMsg	databus/responses			
	SunGuide Request / Responses	Subsystem	Request	Response	Parameters
		cctv	authenticateReq	authenticateResp	refId = refId1 icdVersion = 1.0 providerName = cctv username = <username> password = <password>
		cctv	subscribeReq	subscribeResp	refId = refId1 icdVersion = 1.0 providerName = cctv username = <username> securityToken = <securityToken> cameraData = true cameraStatus = true
		databus	retrieveDataReq	retrieveDataResp	refId = refId1 icdVersion = 1.0 username = <username> securityToken = <securityToken>
	databus	subscribeReq	subscribeResp statusUpdateMsg	refId = refId1 icdVersion = 1.0 providerName = databus dataReq = cctv	
Target Format	XML (identical to source)				
Target Location	Kafka Topics: sunguide_camera				

3.4.1.1.3.4 SunGuide Truck Parking (SG-TP-DR)

SG-TP-DR is the R-ICMS driver that retrieves Truck Parking Facility configuration and status information from SunGuide. The configuration details are used for building a map layer, the status data for those Truck Parking Facilities will show the current message and status displayed at that time on the map.

- The generalized process details for the SunGuide data ingestion are documented under section 3.4.1.1.1.2.2 SunGuide Databus.
- When a request is made to SunGuide for the data stream for the first time, it will provide the complete current state of the Truck Parking Facilities including the configuration and current status.
- The subsequent updates are pushed to the client process within the subscription mechanism.
- The following describe the sequence of steps involved to get Truck Parking Facility data feed from SunGuide system:
 1. Request connection
 2. Authenticate request
 3. Retrieve data request from TPS subsystem (initial set of truck parking facility data)
 4. Subscribe to databus – receive updates
 5. Subscribe to TPS subsystem – receive facility changes

Table 9 – SunGuide Truck Parking Ingestion

Data Set Name	Attribute	Value			
SunGuide Truck Parking	Connection	SunGuide Server xxx.xxx.xxx.xxx Port 8089			
	Source Format	XML			
	Source ICD Locations	Source URL	~ /7_1/SunGuide%207.1%20Schemas.zip		
		Schema	Folder within zip		
		common - authenticateReq	common/requests		
		common - authenticateResp	common/responses		
		tps - subscribeReq	tps/requests		
		tps - subscribeResp	tps/responses		
		tps - retrieveDataReq	tps/requests		
		tps - retrieveDataResp	tps/responses		
		databus – subscribeReq	databus/requests		
		databus – subscribeResp	databus/responses		
	databus – statusUpdateMsg	databus/messages			
	SunGuide Request / Responses	Subsystem	Request	Response	Parameters
		tps	authenticateReq	authenticateResp	refId = refId1 icdVersion = 1.0 providerName = tps username = <username> password = <password>
		tps	subscribeReq	subscribeResp addTruckParkingFacilityResp modifyTruckParkingFacilityResp deleteTruckParkingFacilityResp	refId = refId1 icdVersion = 1.0 providerName = tps username = <username> securityToken = <securityToken> truckParkingFacilityData = true
		tps	retrieveDataReq	retrieveDataResp	
	databus	subscribeReq	subscribeResp statusUpdateMsg	refId = refId1 icdVersion = 1.0 providerName = databus dataReq = truckParkingFacility	
Target Format	XML (gzip compressed)				
Target Location	Kafka Topic: sunguide_truck_parking				

3.4.1.1.3.5 SunGuide Traffic Signals (SG-TS-DR)

SG-TS-DR is the R-ICMS driver that retrieves Traffic Signal configuration and status information from SunGuide. The configuration details are used for building a map layer, the status data for those Traffic Signals will be displayed on the map.

- The generalized process details for the SunGuide data ingestion are documented under section 3.4.1.1.2.2 SunGuide Databus.
- When a request is made to SunGuide for the data stream for the first time, it will provide the complete current state of the Traffic Signals including the configuration and current status.
- The subsequent updates are pushed to the client process within the subscription mechanism.
- The following describe the sequence of steps involved to get Traffic Signal data feed from SunGuide system:
 1. Request connection
 2. Authenticate request
 3. Retrieve data requests from TCS subsystem (initial set of Traffic Signal Status, Traffic Signal Inventory, Timing Plans, and Schedules)
 4. Subscribe to TCS subsystem – receive changes to Traffic Signal Status, Traffic Signal Inventory, Timing Plans, and Schedules

Table 10 – SunGuide Traffic Signals Ingestion

Data Set Name	Attribute	Value			
SunGuide Traffic Signals	Connection	SunGuide Server xxx.xxx.xxx.xxx Port 8089			
	Source Format	XML			
	Source ICD Locations	Schemas URL	~/7_1/SunGuide%207.1%20Schemas.zip		
		Schema	Folder within zip		
		tcs - authenticateReq	common/requests		
		tcs - authenticateResp	common/responses		
		tcs - subscribeReq	tcs/requests		
		tcs - subscribeResp	tcs/responses		
		tcs - retrieveDataReq	tcs/requests		
		tcs - retrieveDataResp	tcs/responses		
	SunGuide Request / Responses	Subsystem	Request	Response	Parameters
		tcs	authenticateReq	authenticateResp	refId = refId1 icdVersion = 1.0 providerName = tcs username = <username> password = <password>
		tcs	subscribeReq	subscribeResp addTrafficSignalMsg modifyTrafficSignalMsg removeTrafficSignalMsg addTimingPlanMsg modifyTimingPlanMsg removeTimingPlanMsg addScheduleMsg modifyScheduleMsg removeScheduleMsg	refId = refId1 icdVersion = 1.0 providerName = tcs username = <username> securityToken = <securityToken> trafficSignalData = true timingPlanData = true scheduleData = true
		tcs	retrieveDataReq	retrieveDataResp	refId = refId1 icdVersion = 1.0 providerName = tcs username = <username> securityToken = <securityToken> trafficSignalStatus = true timingPlans = true schedules = true
	Target Format	XML (gzip compressed)			
Target Location	Kafka Topic: sunguide_traffic_signal				

3.4.1.1.3.6 SunGuide Ramp Meters (SG-RM-DR)

SG-RM-DR is the R-ICMS driver that retrieves Ramp Meter configuration and status information from SunGuide. The configuration details are used for building a map layer, the status data for those Ramp Meters will show the current message and status displayed at that time on the map.

- The generalized process details for the SunGuide data ingestion are documented under section 3.4.1.1.1.2.2 SunGuide Databus.
- When a request is made to SunGuide for the data stream for the first time, it will provide the complete current state of the Ramp Meters including the configuration and current status.
- The subsequent updates are pushed to the client process within the subscription mechanism.
- The following describe the sequence of steps involved to get Ramp Meter data feed from SunGuide system:
 1. Request connection
 2. Authenticate request
 3. Retrieve data requests from RMS subsystem (initial set of Ramp Meter configuration and status information)
 4. Subscribe to RMS subsystem – receive changes to Ramp Meter configuration and status information

Table 11 – SunGuide Ramp Meters Ingestion

Data Set Name	Attribute	Value			
SunGuide Ramp Meters	Connection	SunGuide Server xxx.xxx.xxx.xxx Port 8089			
	Source Format	XML			
	Source ICD Locations	Schemas URL	~/7_1/SunGuide%207.1%20Schemas.zip		
		Schema	Folder within zip		
		rms - authenticateReq	common/requests		
		rms - authenticateResp	common/responses		
		rms - subscribeReq	rmc/requests		
		rms - subscribeResp	rmc/responses		
		rms - retrieveDataReq	rmc/requests		
	rms - retrieveDataResp	rmc/responses			
	SunGuide Request / Responses	Subsystem	Request	Response	Parameters
		rms	authenticateReq	authenticateResp	refId = refId1 icdVersion = 1.0 provider username = <username> password = <
		rms	subscribeReq	subscribeResp addRmcUpdateMsg rmcUpdateMsg	refId = refId1 icdVersion = 1.0 provider username = <username> securityToken rmcData = true statusUpdate = true
		rms	retrieveDataReq	retrieveDataResp	refId = refId1 icdVersion = 1.0 provider username = <username> securityToken rmcData = true statusList = true
Target Format	XML (gzip compressed)				
Target Location	Kafka Topic: sunguide_ramp_meter				

3.4.1.1.3.7 SunGuide Connected Vehicles (SG-CV-DR)

SG-CV-DR is the R-ICMS driver that retrieves Roadside Equipment (RSE) and Travel Advisory Message (TAM) information from SunGuide. The configuration details are used for building a map layer and the current status data for RSEs and TAMs are presented on that map layer.

- The generalized process details for the SunGuide data ingestion are documented under section 3.4.1.1.1.2.2 SunGuide Databus.
- When a request is made to SunGuide for the data stream for the first time, it will provide the complete current state of the RSEs and TAMs including the configuration and current status.
- The subsequent updates are pushed to the client process within the subscription mechanism.
- The following describe the sequence of steps involved to get Ramp Meter data feed from SunGuide system:
 1. Request connection
 2. Authenticate request
 3. Retrieve data requests from CVS subsystem (initial set of RSE and TAM configuration and status information)
 4. Subscribe to CVS subsystem – receive changes to RSE and TAM configuration and status information

Table 12 - SunGuide Connected Vehicles Ingestion

Data Set Name	Attribute	Value				
SunGuide Ramp Meters	Connection	SunGuide Server xxx.xxx.xxx.xxx Port 8089				
	Source Format	XML				
	Source ICD Locations	Source ICD	Schemas URL	~/7_1/SunGuide%207.1%20Schemas.zip		
		Schema	Folder within zip			
		cvs - authenticateReq	common/requests			
		cvs - authenticateResp	common/responses			
		cvs - subscribeReq	cvs/requests			
		cvs - subscribeResp	cvs/responses			
		cvs - retrieveDataReq	csv/requests			
		cvs - retrieveDataResp	csv/responses			
		databus – subscribeReq	databus/requests			
		databus – subscribeResp	databus/responses			
		databus – statusUpdateMsg	databus/messages			
	SunGuide Request / Responses	Subsystem	Request	Response	Parameters	
		cvs	authenticateReq	authenticateResp	refId = refId1 icdVersion = 1.0 providerName = cvs username = <username> password = <password>	
		cvs	subscribeReq	subscribeResp addRseResp modifyRseResp deleteRseResp addTamResp modifyTamResp deleteTamResp	refId = refId1 icdVersion = 1.0 providerName = cvs username = <username> securityToken = <securityToken> rseConfig = true tamData = true	
		cvs	retrieveDataReq	retrieveDataResp	refId = refId1 icdVersion = 1.0 providerName = cvs username = <username> securityToken = <securityToken> rseConfigData = true statusList = true	
		databus	subscribeReq	subscribeReq statusUpdateMsg	refId = refId1 icdVersion = 1.0 providerName = databus dataReq = rse	
	Target Format	XML (gzip compressed)				
	Target Location	Kafka Topic: sunguide_cvs				

3.4.1.1.4 Transit AVL Drivers

The Transit AVL Drivers will retrieve trip updates and vehicle positions from a REST API hosted by FDOT that provides access to the data collected and stored by their GTFS Aggregator application. Since the ultimate source of this data is a GTFS-RT feed file from each participating transit agency it shares the same data as the feed file but is delivered by the GTFS Aggregator REST API in JSON format.

The RICMS will have separate drivers for ingesting current data and archive data.

3.4.1.1.4.1 Transit AVL Current Data Driver (TAVL-CU-DR)

The Transit AVL Current Data Driver (TAVL-CU-DR) will retrieve current transit AVL feed data using the GTFS Aggregator REST API and publish it to a pipeline (Kafka topic). Multiple instances of the driver will be deployed. The deployed drivers will share a common codebase but will be configured to either current retrieve trip updates or vehicle positions feed data.

1. Read the required configuration from the runtime configuration file. The configuration file includes the GTFS Aggregator API routes, API key, and Kafka topic.
2. Issue an HTTP request to the configured agencies route to get a list of all supported agencies.
3. Iterate over the list of agencies in the response and issue an HTTP request to the configured transit AVL feed data route.
4. Publish the received data to configured Kafka topic and include a message header that indicates the date/time when the data was ingested
5. A common notification library will interact with the NOT-BS, allowing a service to create both alerts and notifications.
6. All data receiving events will be logged into the Common Core Log System with details about the time, and data being received

Table 13 – Transit AVL Current Data Ingestion

Data Set Name	Attribute	Value
Transit AVL Current Trip Updates	API URL	http://itssd5gtfsda01.d5-its.tsmo.dot.state.fl.us/developer/api/v2
	Source Format	JSON
	Routes	/agencies /gtfsrttripupdates
	Target Format	JSON (gzip compressed)
	Target Location	Kafka Topic: transit_avl_trip_updates_current
Transit AVL Current Vehicle Positions	API URL	http://xxx.xxx.xxx.xxx/developer/api/v2
	Source Format	JSON
	Routes	/agencies /gtfsrtvehiclepositions
	Target Format	JSON (gzip compressed)
	Target Location	Kafka Topic: transit_avl_vehicle_positions_current

3.4.1.1.4.2 Transit AVL Archive Driver (TAVL-AR-DR)

The Transit AVL Archive Data Driver (TAVL-AR-DR) will retrieve archived transit AVL feed data using the GTFS Aggregator REST API and publish it to a pipeline (Kafka topic). Multiple instances of the driver will be deployed. The deployed drivers will share a common codebase but will be configured to either archived retrieve trip updates or vehicle positions feed data.

1. Read the required configuration from the runtime configuration file. The configuration file includes the GTFS Aggregator API routes, API key, Kafka topic, and MongoDB collection.
2. Issue an HTTP request to the configured agencies route to get a list of all supported agencies.
3. Iterate over the list of agencies in the response and perform the following steps
 - a. Perform a lookup in MongoDB (if no Kafka topic) to determine the most recent archive file that was successfully ingested and saved to MongoDB.
 - b. If the lookup is successful then use the datetime of that file for the subsequent step; otherwise calculate a datetime for a configured number of days in the past from which to get archived data
 - c. Issue an HTTP request to the configured API route to retrieve the list of files since the datetime determined in the previous step.
 - d. Iterate over the list of files and issue an HTTP request to the configured API route to retrieve the file.
 - e. Publish the received file data to the configured Kafka topic and include a message header that indicates the date/time when the data was ingested
4. A common notification library will interact with the NOT-BS, allowing a service to create both alerts and notifications.
5. All data receiving events will be logged into the Common Core Log System with details about the time, and data being received

Table 14 – Transit AVL Archive Data Ingestion

Data Set Name	Attribute	Value
Transit AVL Archive Trip Updates	API URL	http://itssd5gtfsda01.d5-its.tsmo.dot.state.fl.us/developer/api/v2
	Source Format	JSON
	Routes	/agencies
		/gtfsrttripupdatesfiles /gtfsrttripupdatesfromfile
	Target Format	JSON (gzip compressed)
	Target Location	Kafka Topic: transit_avl_trip_updates_archive
Transit AVL Archive Vehicle Positions	API URL	http://xxx.xxx.xxx.xxx/developer/api/v2
	Source Format	JSON
	Routes	/agencies
		/gtfsrtvehiclepositionsfiles

		/gtfsrtvehiclepositionsfromfile
	Target Format	JSON (gzip compressed)
	Target Location	Kafka Topic: transit_avl_vehicle_positions_archive

3.4.1.1.5 Transit Static Data (TSD-DR)

The Transit Static Data Driver (TSD-DR) will retrieve transit static data from a REST API hosted by FDOT that provides access to the data collected and stored by their GTFS Aggregator application. Since the ultimate source of this data is a GTFS feed file from each participating transit agency it shares the same data as the feed file but is delivered by the GTFS Aggregator REST API in JSON format.

Since the size of the transit feed data typically exceeds the maximum recommended message size for Kafka the transit static data driver will save the feed data to a file system and publish only the id that refers to the file to support downstream pipeline applications and subscription services.

1. At a configurable frequency the driver requests the list of agencies from the agencies API route.
2. The driver iterates over the list of agencies in the API response and compares the last updated timestamp in the response to the last updated timestamp of the previously saved response in MongoDB.
3. If the feed data for any agencies is determined to be changed or new the driver retrieves the feed data for those agencies using the gtfsagency route.
4. The driver saves each agency’s feed data to GridFS which is a file storage system that is part of MongoDB.
5. Upon saving the feed data for each agency, the driver receives the file id of the GridFS file for that feed data and publishes it to a kafka topic.

Table 15 – Transit Static Data Ingestion

Data Set Name	Attribute	Value
Transit Static	API URL	http://xxx.xxx.xxx.xxx/developer/api/v2
	Source Format	JSON
	Routes	/agencies
		/gtfsbyagency
	Target Format	BSON
Target Locations	MongoDB GridFS bucket: transit.static (file content) Kafka Topics: transit_static (file id)	

3.4.1.1.6 Signal Controller Log (SCL-DR)

The Signal Controller Log Driver (SCL-DR) will retrieve ZIP files containing one or more binary signal controller log DAT files from a drop folder. The SCL data is from four different signal

vendors: Trafficware, Econolite, and Intelight. The data from each of the vendors is in a proprietary format for which there is no ICD.

- The driver polls the drop folder looking for ZIP files
- When ZIP files are found, the driver retrieves each ZIP file, unzips it, and publishes each of the DAT files within the ZIP file to a Kafka topic

Table 16 – Signal Controller Log Ingestion

Data Set Name	Attribute	Value
Signal Controller Log	UNC Path	\\xxx.xxx.xxx.xxx\atspm_dat_drop\RICMS_PROD
	File Name	Snap_yyyyMMdd_HHmms_fff.zip
	Source Format	ZIP file containing a set of vendor proprietary binary DAT files
	Source ICD	http://www.cflsmartrroads.com/projects/smartsignals.html
	Target Format	Vendor proprietary binary DAT
	Target Location	Kafka Topic: signal_controller_log_dat

3.4.1.1.7 Basemap (GIS) (BM-DR)

The Basemap Driver (BM-DR) archives updated ARBM roadway network ESRI geodatabase to the HDFS file system. The load process combines manual steps with automated steps.

1. Get the ARBM roadway network geodatabase
2. Use ESRI ArcGIS to generate a new set of map tiles from the ARBM roadway network geodatabase, storing them in the ESRI Data Store
3. Move geodatabase to the staging area specified by the configuration file
4. BM-DR automatically detects the new geodatabase in the staging area
5. BM-DR writes the geodatabase files to HDFS
6. BM-DR will delete the geodatabase from the staging area
7. The driver will use the notification service to send notices the new roadway network has been saved in HDFS
8. The driver will log the update event in the system log.

Table 17 – Basemap Specification specifies attributes of the data for the driver.

Table 17 – Basemap Specification

Data Set Name	Basemap
Source location	https://ubr.fdot.gov/featured
File Name	ARBM2011_withAADT.zip ARBM2013Q1_07.zip ARBM2015Q1_v4.gdb.zip NavTeq2018Q1_RT.gdb.zip
Source Format	ESRI Shape Files in ZIP format
ICD Source Location	https://www.esri.com/library/whitepapers/pdfs/shapefile.pdf
Target Location	HDFS File store: /user/gis/basemap/<year>/<month>/<day>/basemap_<date>_<time>.zip

Target Format	Binary ZIP file in the HDFS
Frequency	As needed

3.4.1.1.8 School Zones (GIS) (SZ-DR)

The School Zones Driver (SZ-DR) archives updated school zone data as ESRI shape files in the HDFS files system. The load process combines manual steps with automated steps.

1. Download the School Zones GIS Shape file and upload to location designated by the configuration file.
2. SZ-DR scans the respective folder at specified intervals looking for new file. When a new file is copied into the folder, then the process reads and loads the data into the HDFS.
3. SZ-DR automatically detects the new shape files in the staging area
4. SZ-DR writes the shape files to HDFS
5. SZ-DR will delete the shape files from the staging area
6. SZ-DR will use the notification service to send notices the new roadway network has been saved in HDFS
7. The driver will log the update event in the system log.

Table 18 – School Zones Specification specifies attributes of the data for the driver.

Table 18 – School Zones Specification

Data Set Name	School Zones
Source location	Provided by FDOT
File Name	FDOT provided zip file.
Source Format	Binary GIS Shape File in ZIP format
ICD Source Location	https://www.esri.com/library/whitepapers/pdfs/shapefile.pdf
Target Location (HDFS)	HDFS file store location: /user/gis/school_zones/<year>/<month>/<day>/school_zones_<date>_<time>.zip
Target Format	Binary ZIP file in the HDFS

3.4.1.1.9 School Locations (GIS) (SL-DR)

The School Zones Driver (SL-DR) archives updated school zone data as ESRI shape files in the HDFS files system. The load process combines manual steps with automated steps.

1. Download the School Zones GIS Shape file and upload to a location designated by the config file.
2. SL-DR scans the respective folder at specified intervals looking for new file. When a new file is copied into the folder, then the process reads and loads the data into the HDFS.
3. SL-DR automatically detects the new shape files in the staging area
4. SL-DR writes the shape files to HDFS
5. SL-DR will delete the shape files from the staging area
6. SL-DR will use the notification service to send notices the new roadway network has been saved in HDFS
7. The driver will log the update event in the system log.

Table 19 – School Locations Specification specifies attributes of the data for the driver.

Table 19 – School Locations Specification

Data Set Name	School Locations
Source location	https://www.fgdl.org/metadataexplorer/explorer.jsp
File Name	SCHOOLS_D5_JAN07
Source Format	Binary GIS Shape File in ZIP format
ICD Source Location	https://www.esri.com/library/whitepapers/pdfs/shapefile.pdf
Metadata	https://www.fgdl.org/metadataexplorer/full_metadata.jsp?docId=%7B44BC265D-78B0-4DE0-B2FD-3191CE555231%7D&loggedIn=false
Target Location (HDFS)	/user/gis/school_locations/<year>/<month>/<day>/school_locations_<date>_<time>.zip
Target Format	Binary ZIP file in the HDFS

3.4.1.1.10 Emergency Responder Locations (GIS) (ERL-DR)

The Emergency Responder Locations Driver (ERL-DR) loads emergency responder locations into the HDFS files system; emergency responder locations include fire stations, hospitals and law enforcement offices.

The load process combines manual steps with automated steps.

1. Download the School Zones GIS Shape file and upload to location specified by the config file.
2. ERL-DR scans the respective folder at specified intervals looking for new file. When a new file is copied into the folder, then the process reads and loads the data into the HDFS.
3. ERL-DR automatically detects the new shape files in the staging area
4. ERL-DR writes the shape files to HDFS
5. ERL-DR will delete the shape files from the staging area
6. ERL-DR will use the notification service to send notices the new roadway network has been saved in HDFS
7. The driver will log the update event in the system log.

Data sources for the four counties is below.

- Seminole County:
 - <http://www.seminolecountyfl.gov/departments-services/information-services/gis-geographic-information-systems/gis-data.shtml>
 - [Facilities.gdb.zip](#)
- Orange County Interactive Map:
 - ftp://ftp.onetgov.net/divisions/Infomap/pub/GIS_Downloads/FTP_Shapefile.zip
 - Fire Stations Countywide.zip
 - Law Enforcement Agencies.zip
- Osceola County:
 - <https://fldot.sharepoint.com/sites/FDOT-EXT-TEO/D5TSMO/DataInitiatives/ICMS/Shared%20Documents/10%20Data%20Sources/Osceola%20County%20Facilities>
- Volusia County:
 - <http://maps.vcgov.org/gis/download/shpfiles/>
 - facility.zip

Table 20 – Emergency Responder Locations Specification specifies attributes of the data for the driver.

Table 20 – Emergency Responder Locations Specification

Data Set Name	Emergency Responder Locations
Source location	See above
File Name	See above
Source Format	Binary GIS Shape File in ZIP format
ICD Source Location	https://www.esri.com/library/whitepapers/pdfs/shapefile.pdf
Target Location (HDFS)	/user/gis/emergency_responder_locations/<county>/<year>/<month>/<day>/emergency_responder_locations<county>_<date>_<time>.zip
Target Format	Binary ZIP file in the HDFS

3.4.1.1.11 *Signalized Intersections (SI-DR)*

The Signalized Intersections driver retrieves intersection geometry and other data from the Signalized Intersection Inventory Application and stores the data in the HDFS. The REST API can be queried in two ways:

- Using multiple routes that return summary information for a set of intersections and
- Using single route that returns detail information for a single intersection whose ID was provided as a parameter.

Consequently, the Signalized Intersections driver will need to use multiple routes in order to retrieve the detail information for multiple intersections.

The typical process steps to get the data from the SIIA REST API into R-ICMS are as follows:

1. The driver will read the required configuration from the runtime configuration file. The configuration file includes the API connection details and required routes.
2. The driver will send an authentication request and receive an authorization token to be used when sending subsequent data requests.
3. The driver will determine whether it has run before by querying for the document with the maximum last modified date/time in the MongoDB collection. If no document is returned, then the current run is the first run.
4. The driver will send the appropriate data requests to get summary information
 - a. If first run, use the appropriate route to retrieve summary information for all intersections
 - b. If subsequent run, use the appropriate route to retrieve summary information for only those intersections that have been modified since the maximum last modified date/time determined in step 3
5. The driver will extract the list of intersection IDs from the summary response, loop over that list, use the appropriate route to retrieve intersection detail for each intersection
6. The driver will publish each intersection detail response to a Kafka topic and include a message header that indicates the date/time when the data was ingested
7. A common notification library will interact with the NOT-BS, allowing a service to create both alerts and notifications.

8. All data receiving events will be logged into the Common Core Log System with details about the time, and data being received

Table 21 – Signalized Intersections Ingestion

Data Set Name	Attribute	Value
Signalized Intersections	API URL	http://xxx.xxx.xxx.xxx:8092/siia.api/api
	Source Format	JSON
	Routes	/Token
		/allIntersections
		/intersectionByModifiedDateTime/{datetime}
		/intersection/{id}
	Target Format	JSON (identical to source)
Target Location	Kafka Topics: siia	

3.4.1.1.12 Weather Alerts (WEA-DR)

The Weather Alerts Driver (WEA-DR) retrieves active weather alert data from the National Weather Service (NWS) REST API for a configurable list of zones (counties) every 5 minutes by default but the frequency is configurable.

The typical process steps to get the data from the NWS REST API into R-ICMS are as follows:

1. The driver will read the required configuration from the runtime configuration file. The configuration file includes the API connection details, list of zones, and polling frequency.
2. The driver will send the API request to retrieve the weather alert data, formatting and passing the list of zones as query parameters within the URL.
3. The driver will publish the entire response to a Kafka topic and include a message header that indicates the date/time when the data was ingested
4. A common notification library will interact with the NOT-BS, allowing a service to create both alerts and notifications.
5. All data receiving events will be logged into the Common Core Log System with details about the time, and data being received

Table 22 – Weather Alerts Ingestion

Data Set Name	Attribute	Value
Weather Alerts	API URL	https://api.weather.gov
	Source Format	JSON
	Route	/alerts/active
	Headers	accept: application/ld+json
	Parameters	zone=<list_of_zone_ids>
	Target Format	JSON (gzip compressed)
	Target Location	Kafka Topics: nws_weather_alerts

3.4.1.2 Pipelines

The following sections describe the Pipeline layer of the architecture and each of the Pipelines being implemented. The R-ICMS application implements a Pipeline layer to distribute data from producers (e.g. drivers) to consumers (data stores, data services, etc.) in near real time. The R-ICMS Pipelines are designed using the Kafka open source software and its concepts of topics, connectors, stream processors, etc. Please see the Kafka reference materials for detail on Kafka. Kafka can be scaled into a cluster when workloads demand more streaming resources. In terms of more classical terminology, Pipelines implement the “TL” of “ETL” processing.

Figure 9 – R-ICMS Pipeline illustrates in a simple manner the construction of a Pipeline using the colors from Figure 2 – System Architecture Layer Color Legend.

The figure illustrates that

- Pipelines have a single-entry point for data records through a Kafka Topic
- Data may be pushed directly to a Data Service through a Consumer API (i.e. consumer)
- Data may be pushed through a Connector to transform the data for a Data Store (e.g., HDFS or MongoDB), a Data Service, or an External Application
- Data may be pushed through a Stream Processor transforming the data into a second Kafka Topic before being pushed into a Data Service, a Connector for a Data Store, or a 3rd Party Connector into an External System.
- The illustrated configuration is not exhaustive.

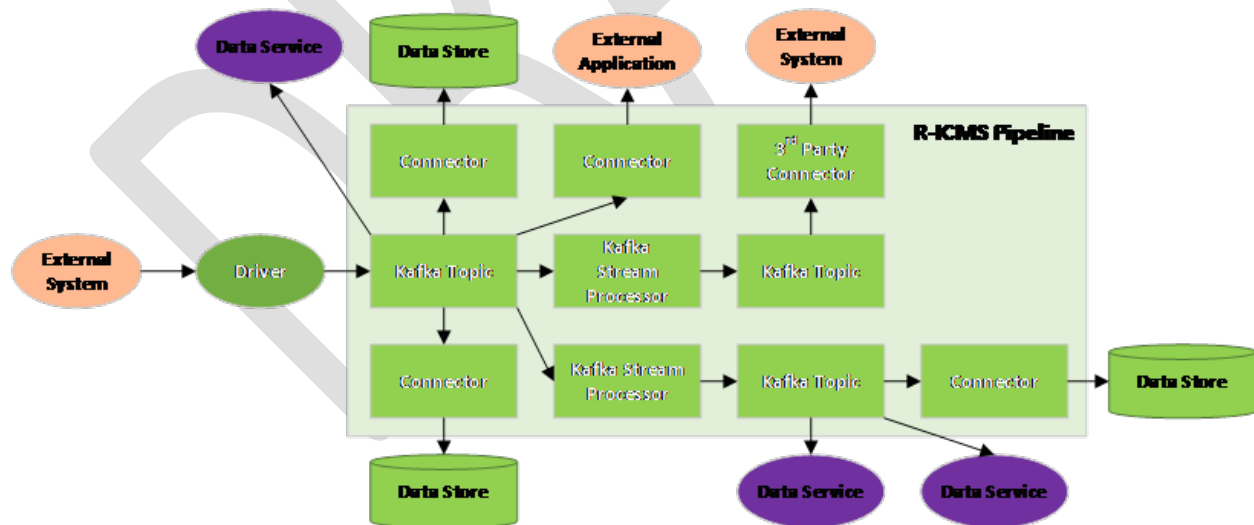


Figure 9 – R-ICMS Pipeline

3.4.1.2.1 Traffic Pipelines (TRF-PL)

Traffic Pipelines (TRF-PL) are pipelines that distribute traffic data ingested by the Driver layer throughout the remainder of the R-ICMS architecture.

3.4.1.2.1.1 ITSQA Current Traffic Conditions Pipeline (TRF-CTC-PL)

The ITSQA Current Traffic Conditions Pipeline (TRF-CTC-PL) represents Pipelines that receive data from the TRF-CTC-DR drivers. There are two sets of Pipelines, one for each of the two current traffic conditions feeds i.e. one that includes HERE data and one that does not. Furthermore, there are separate Pipelines for each data set in the each of the current data feeds. As illustrated in Figure 10 – TRF-CTC-PL Pipeline. The single MongoDB Connector shown in Figure 10 is representative of one or more connectors that may be present in the pipeline.

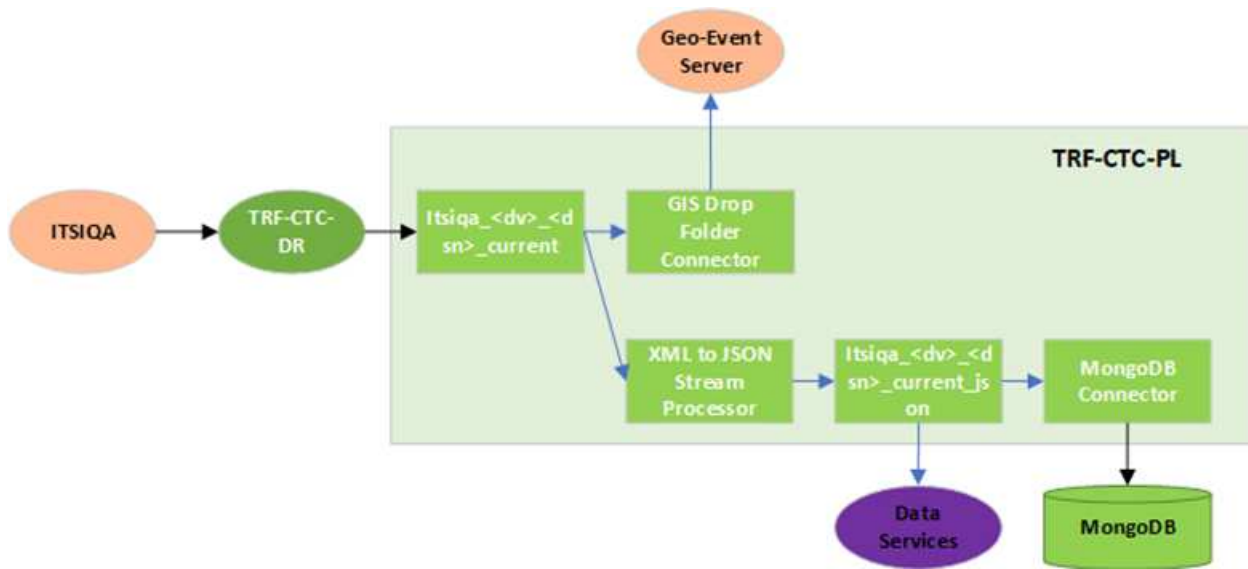


Figure 10 – TRF-CTC-PL Pipeline

- The head of the pipeline is the Kafka topic `itsqa_<data_version>_<data_set_name>_current` where `<data_version>` (abbreviated `<dv>`) is one of `all_sources` or `no_here` and `<data_set_name>` (abbreviated `<dsn>`) is taken from the list of datasets enumerated in 3.4.1.1.2 ITSQA Traffic Conditions Drivers. E.g.
 - `itsqa_all_sources_link_config_current`,
 - `itsqa_all_sources_traffic_data_current`,
 - `itsqa_all_sources_lane_traffic_data_current`,
 - `itsqa_all_sources_class_data_current`,
 - `itsqa_all_sources_tmc_config_current`,
 - `itsqa_all_sources_tmc_data_current`,
 - `itsqa_no_here_link_config_current`,
 - `itsqa_no_here_traffic_data_current`,
 - `itsqa_no_here_lane_traffic_data_current`,
 - `itsqa_no_here_class_data_current`,
 - `itsqa_no_here_tmc_config_current`
 - `itsqa_no_here_tmc_data_current`
- All data is processed through a Stream Processor that transforms the data from ITSQA XML to JSON format. The JSON data is then published to an internal Kafka topic `itsqa_<dv>_<dsn>_current_json`.

- A GIS Drop Folder Connector transforms the ITSQA XML data from the itsqa_all_sources_traffic_data_current topic into XML suitable for ingestion by ArcGIS Geo-Event Server and saves the XML to files in a drop folder accessible by the Geo-Event Server.
- A MongoDB connector reads the itsqa_<dv>_<dsn>_current_json Kafka topic and writes to the MongoDB in BSON format. There is an additional MongoDB connector that reads from itsqa_all_sources_link_config_current_json, processes it to determine the upstream link for each link, and writes the resulting JSON to MongoDB in BSON format.
- ITSQA subscription services read the itsqa_<dv>_<dsn>_current_json topic through the consumer API for further distribution.

Table 23 – ITSQA Current Traffic Conditions Pipeline displays the Pipeline specifications for current traffic conditions retrieved from ITSQA.

Table 23 – ITSQA Current Traffic Conditions Pipeline Specifications

Pipeline Element	Description	
itsqa_<dv>_<dsn>_current	Type	Kafka Topic
	Source	TRF-CTC-CR
XML to JSON Stream Processor	Type	Stream Processor
	Description	Transforms data from XML to JSON
	Source Topic	itsqa_<dv>_<dsn>_current
	Source Format	ITSQA XML (gzip compressed)
	Target Topic	itsqa_<dv>_<dsn>_current_json
	Target Format	JSON (gzip compressed)
GIS Drop Folder Connector	Type	Connector
	Description	Transforms XML to XML suitable for Geo-Event-Server ingestion and saves files to drop folder accessible by Geo-Event-Server
	Source Topic	itsqa_all_sources_traffic_data_current
	Source Format	ITSQA XML (gzip compressed)
	Target File Name	traffic_YYYYMMdd_hhmmss.xml
	Target Format	XML
itsqa_<dv>_<dsn>_current_json	Type	Kafka Topic
	Source	XML to JSON Stream Processor
MongoDB Connector	Type	Connector
	Description	Minimally transforms source JSON (i.e., date/time deserialization) and saves data to MongoDB
	Source Topic	itsqa_<dv>_<dsn>_current_json
	Source Format	JSON (gzip compressed)
	Target Collection	Itsqa.<dv>_<dsn>_current
	Target Format	BSON
Upstream Links MongoDB Connector	Type	Connector
	Description	Produces the list of upstream links for each link in LinkConfig and saves data to MongoDB
	Source Topic	itsqa_all_sources_link_config_current_json
	Source Format	JSON (gzip compressed)
	Target Collection	Itsqa.all_sources_upstream_links
	Target Format	BSON

Pipeline Element	Description

3.4.1.2.1.2 ITSQA Archive Traffic Conditions (TRF-ATC-PL)

The ITSQA Archive Traffic Conditions Pipeline (TRF-ATC-PL) receives data from the TRF-ATC-DR driver. TRF-ATC-PL writes the data to the HDFS archive of raw data and the MongoDB to fill gaps as shown in Figure 11. Table 24 – ITSQA Archive Traffic Conditions Pipeline provides the Pipeline specifications for historical traffic conditions retrieved from ITSQA. The processing is sufficiently similar so textual descriptions are not provided. The single MongoDB Connector shown in Figure 11 is representative of one or more connectors that may be present in the pipeline.

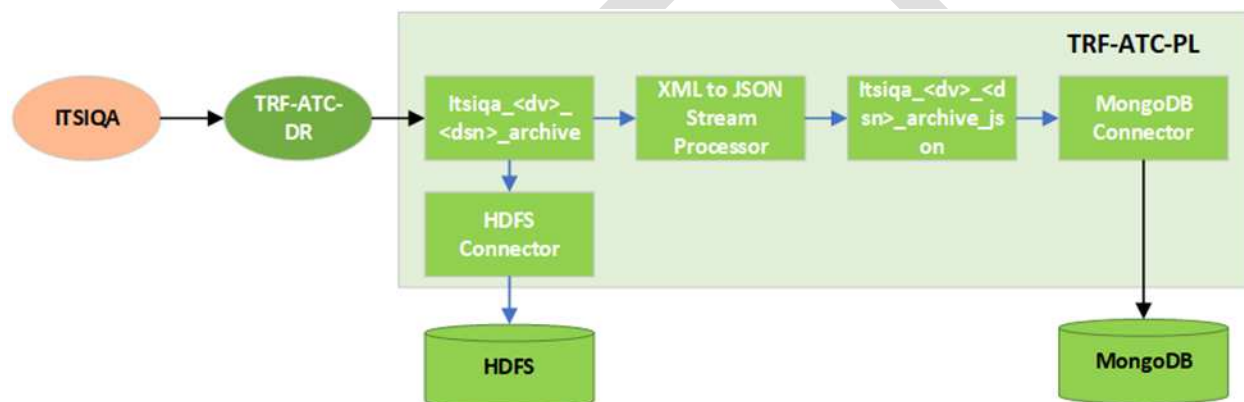


Figure 11 – TRF-ATC-PL Diagram

Table 24 – ITSQA Archive Traffic Conditions Pipeline

Pipeline Element	Description	
itsqa_<dv>_<dsn>_archive	Type	Kafka Topic
	Source	TRF-ATC-DR
HDFS Connector	Type	Connector
	Description	Merges multiple source XML documents into more optimally sized XML files and saves the merged files to HDFS
	Source Topic	itsqa_<dv>_<dsn>_archive
	Source Format	XML (gzip compressed)
	Target File Name	<dsn_pascal_case>_<yyyyMMddhhmmss.xml Notes: 1. dsn_pascal_case is simply the dsn string converted from snake_case to PascalCase. For example, ClassData, TrafficData, LaneTrafficData, etc. 2. yyyyMMddhhmmss is the timestamp of the first source message to be stored in the target file. RICMS may merge multiple source messages into a single target file and there may be multiple target files per day.
	Target Format	XML (many files merged into single file)

Pipeline Element	Description	
XML to JSON Stream Processor	Type	Stream Processor
	Description	Transforms data from XML to JSON
	Source Topic	Itsiqua_<dv>_<dsn>_archive_json
	Source Format	XML (gzip compressed)
	Target Topic	Itsiqua_<dv>_<dsn>_archive_json
	Target Format	JSON (gzip compressed)
itsiqua_<dv>_<dsn>_archive_json	Type	Kafka Topic
	Source	XML to JSON Stream Processor
MongoDB Connector	Type	Connector
	Description	Minimally transforms source JSON (i.e., date/time deserialization) and saves data to MongoDB
	Source Topic	itsiqua_<dv>_<dsn>_archive_json
	Source Format	JSON (gzip compressed)
	Target Collection	Itsiqua.<dv>_<dsn>_historical
	Target Format	BSON
TMC Data for Aggregation MongoDB Connector	Type	Connector
	Description	Reshapes TMCData to facilitate aggregation and saves data to MongoDB
	Source Topic	itsiqua_all_sources_tmc_data_archive_json
	Source Format	JSON (gzip compressed)
	Target Collection	itsiqua.all_sources_tmc_data_for_aggregation
	Target Format	BSON

3.4.1.2.2 SunGuide (SG-PL)

The SunGuide pipelines are of the general form illustrated in Figure 12. There are separate pipelines for each of the SunGuide data types e.g. DMS, CCTV, RM, Events, etc. The flow through the pipeline is similar to that of the flow through TRF-CTC-PL and is not repeated here; the exception being that an additional connector is introduced to write data to HDFS.

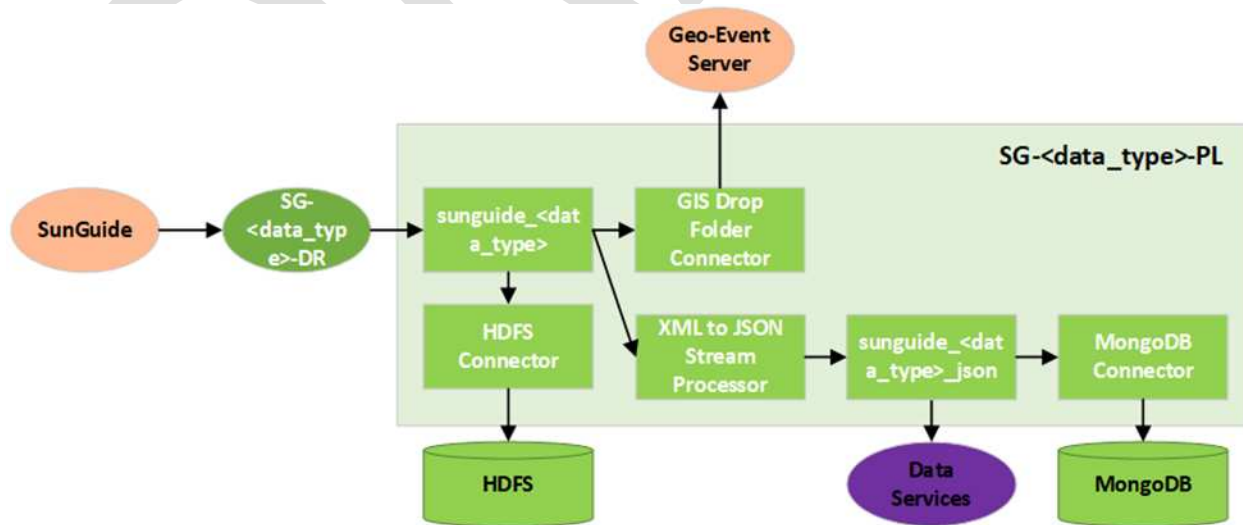


Figure 12 – SunGuide Generalized Pipeline Diagram

Records in the Kafka topic at the head of the Pipeline include a header that consists of:

- response-type: The xml root element tag of the SunGuide response
- message-type: The value derived from the SunGuide response which indicates the type of the operation associated with the response
- date-type: The particular kind of SunGuide in the xml
- received-date-time: The timestamp when the SunGuide response is received by the driver
- compression-type: The type of compression applied to the XML (to reduce the message size in the Kafka topic)

3.4.1.2.2.1 SunGuide DMS (SG-DMS-PL)

The structure of the SunGuide DMS Pipeline (SG-DMS-PL) is described in Table 25. Note that there will be a separate instance of the HDFS Connector for each message type but Table 25 describes this connector in general and not a specific instance of the connector.

Table 25 – SunGuide DMS Pipeline

Pipeline Element	SunGuide DMS (Dynamic Message Signs)	
sunguide_dms	Type	Kafka Topic
	Source	SG-DMS-DR
HDFS Connector	Type	Connector
	Description	Merges multiple XML files into an optimally sized files and stores the merged files to HDFS
	Source	sunguide_dms
	Source Format	SunGuide XML (gzip compressed)
	Target File Name	<base_file_name>_<yyyyMMddhhmmss>.xml Where: <ol style="list-style-type: none"> 1. base_file_name is the root element tag. For example, retrieveDataResp, statusUpdateMsg, etc. 2. yyyyMMddhhmmss is the timestamp of the first source message to be stored in the target file. RICMS may merge multiple source messages into a single target file and there may be multiple target files per day.
Target Format	XML	
GIS Drop Folder Connector	Type	Connector
	Description	Transforms SunGuide XML to XML suitable for ESRI Geo-Event Server ingestion and saves XML files to drop folder accessible by Geo-Event-Server
	Source	sunguide_dms
	Source Format	XML (gzip compressed)
	Target File Name	dms_yyyyMMdd_hhmmss.xml
Target Format	XML	
XML to JSON Stream Processor	Type	Stream Processor
	Description	Transforms data from XML to JSON
	Source	sunguide_dms
	Source Format	XML (gzip compressed)
	Target Topic	sunguide_dms_json
Target Format	JSON (gzip compressed)	
sunguide_dms_json	Type	Kafka Topic
	Source	XML to JSON Stream Processor

Pipeline Element	SunGuide DMS (Dynamic Message Signs)	
MongoDB Connector	Type	Connector
	Description	Minimally transforms source JSON (i.e., date/time deserialization) and saves data to MongoDB
	Source	sunguide_dms_json
	Source Format	JSON (gzip compressed)
	Target Collections	sunguide.dms_config_current sunguide.dms_config_historical sunguide.dms_status_current sunguide.dms_status_historical
	Target Format	BSON

3.4.1.2.2 SunGuide CCTV (SG-CCTV-PL)

The structure of the SunGuide CCTV Pipeline (SG-CCTV-PL) is described in Table 26. Note that there will be a separate instance of the HDFS Connector for each message type but Table 26 describes this connector in general and not a specific instance of the connector.

Table 26 – SunGuide CCTV Pipeline

Pipeline Element	SunGuide CCTV (Closed Circuit Television)	
sunguide_camera	Type	Kafka Topic
	Source	SG-CCTV-DR
HDFS Connector	Type	Connector
	Description	Merges multiple XML files into an optimally sized files and stores the merged files to HDFS
	Source	sunguide_camera
	Source Format	XML (gzip compressed)
	Target File Name	<base_file_name>_<yyyyMMddhhmmss>.xml Where: <ol style="list-style-type: none"> base_file_name is the root element tag. For example, retrieveDataResp, statusUpdateMsg, etc. yyyyMMddhhmmss is the timestamp of the first source message to be stored in the target file. RICMS may merge multiple source messages into a single target file and there may be multiple target files per day.
	Target Format	XML
GIS Drop Folder Connector	Type	Connector
	Description	Transforms SunGuide XML to XML suitable for ESRI Geo-Event Server ingestion and saves XML files to drop folder accessible by Geo-Event-Server
	Source	sunguide_camera
	Source Format	XML (gzip compressed)
	Target File Name	camera_yyyyMMdd_hhmmss.xml
	Target Format	XML
XML to JSON Stream Processor	Type	Stream Processor
	Description	Transforms data from XML to JSON
	Source	sunguide_camera
	Source Format	XML (gzip compressed)
	Target	sunguide_camera_json
	Target Format	JSON (gzip compressed)

Pipeline Element	SunGuide CCTV (Closed Circuit Television)	
sunguide_camera_json	Type	Kafka Topic
	Source	XML to JSON Stream Processor
MongoDB Connector	Type	Connector
	Description	Minimally transforms source JSON (i.e., date/time deserialization) and saves data to MongoDB
	Source	sunguide_camera_json
	Source Format	JSON (gzip compressed)
	Target Collections	sunguide.camera_config_current sunguide.camera_config_historical sunguide.camera_status_current sunguide.camera_status_historical
	Target Format	BSON

3.4.1.2.2.3 SunGuide Events (SG-EV-PL)

The structure of the SunGuide Events Pipeline (SG-EV-PL) is described in Table 27. Note that there will be a separate instance of the HDFS Connector for each message type but Table 27 describes this connector in general and not a specific instance of the connector.

Table 27 – SunGuide Events Pipeline

Pipeline Element	SunGuide Events	
sunguide_event	Type	Kafka Topic
	Source	SG-EV-DR
HDFS Connector	Type	Connector
	Description	Merges multiple XML files into an optimally sized files and stores the merged files to HDFS
	Source	sunguide_event
	Source Format	XML (gzip compressed)
	Target File Name	<base_file_name>_<yyyyMMddhhmmss>.xml Where: <ol style="list-style-type: none"> base_file_name is the root element tag. For example, statusResp, statusUpdateMsg, etc. yyyyMMddhhmmss is the timestamp of the first source message to be stored in the target file. RICMS may merge multiple source messages into a single target file and there may be multiple target files per day.
	Target Format	XML
GIS Drop Folder Connector	Type	Connector
	Description	Transforms SunGuide XML to XML suitable for ESRI Geo-Event Server ingestion and saves XML files to drop folder accessible by Geo-Event-Server
	Source	sunguide_event
	Source Format	XML (gzip compressed)
	Target File Name	event_YYYYMMDD_hhmmss.xml
	Target Format	XML
XML to JSON Stream Processor	Type	Stream Processor
	Description	Transforms data from XML to JSON
	Source	sunguide_event
	Source Format	XML (gzip compressed)

Pipeline Element	SunGuide Events	
	Target	sunguide_event_json
	Target Format	JSON (gzip compressed)
sunguide_event_json	Type	Kafka Topic
	Source	XML to JSON Stream Processor
MongoDB Connector	Type	Connector
	Description	Minimally transforms source JSON (i.e., date/time deserialization) and saves data to MongoDB
	Source	sunguide_event_json
	Source Format	JSON (gzip compressed)
	Target Collections	sunguide.event_current sunguide.event_historical
	Target Format	BSON
MSSQL Connector	Type	Connector
	Description	Extracts data required for the event list application and loads it into a Microsoft SQL Server database and publishes the changes to Kafka to support dynamic update of the event list
	Source	sunguide_event_json
	Source Format	JSON (gzip compressed)
	Target SQL Server Tables	EventDb.dbo.Center EventDb.dbo.Event EventDb.dbo.EventApproach EventDb.dbo.EventApproachLane EventDb.dbo.EventBlockage EventDb.dbo.EventChronology EventDb.dbo.EventChronologyType EventDb.dbo.EventLaneType EventDb.dbo.EventSeverity EventDb.dbo.EventStatus EventDb.dbo.EventType
	Target Topic	event_list_json
	Target Topic Format	JSON

3.4.1.2.2.4 SunGuide Truck Parking (SG-TP-PL)

The structure of the SunGuide Truck Parking Pipeline (SG-TP-PL) is described in Table 28. Note that there will be a separate instance of the HDFS Connector for each message type but Table 28 describes this connector in general and not a specific instance of the connector.

Table 28 – SunGuide Truck Parking Pipeline

Pipeline Element	SunGuide Truck Parking	
Kafka Topic	Type	sunguide_truck_parking_facility
	Source	SG-EV-DR
HDFS Connector	Type	Connector
	Description	Merges multiple XML files into an optimally sized files and stores the merged files to HDFS
	Source	sunguide_truck_parking_facility
	Source Format	XML (gzip compressed)
	Target File Name	<base_file_name>_<yyyyMMddhhmmss>.xml Where:

Pipeline Element	SunGuide Truck Parking	
		<ol style="list-style-type: none"> 1. base_file_name is the root element tag. For example, retrieveDataResp, statusUpdateMsg, etc. 2. yyyyMMddhhmmss is the timestamp of the first source message to be stored in the target file. RICMS may merge multiple source messages into a single target file and there may be multiple target files per day.
	Target Format	XML
GIS Drop Folder Connector	Type	Connector
	Description	Transforms SunGuide XML to XML suitable for ESRI Geo-Event Server ingestion and saves XML files to drop folder accessible by Geo-Event-Server
	Source	sunguide_truck_parking_facility
	Source Format	XML (gzip compressed)
	Target File Name	truck_parking_yyyyMMdd_hhmmss.xml
	Target Format	XML
XML to JSON Stream Processor	Type	Stream Processor
	Description	Transforms data from XML to JSON
	Source	sunguide_truck_parking_facility
	Source Format	XML (gzip compressed)
	Target Topic	sunguide_truck_parking_facility_json
	Target Format	JSON (gzip compressed)
sunguide_truck_parking_facility_json	Type	Kafka Topic
	Source	XML to JSON Stream Processor
MongoDB Connector	Type	Connector
	Description	Minimally transforms source JSON (i.e., date/time deserialization) and saves data to MongoDB
	Source	sunguide_truck_parking_facility_json
	Source Format	JSON (gzip compressed)
	Target Collections	sunguide.truck_parking_facility_config_current sunguide.truck_parking_facility_config_historical sunguide.truck_parking_facility_status_current sunguide.truck_parking_facility_status_historical
	Target Format	BSON

3.4.1.2.2.5 SunGuide Traffic Signals (SG-TS-PL)

The structure of the SunGuide Traffic Signals Pipeline (SG-TS-PL) is described in Table 29. Note that there will be a separate instance of the HDFS Connector for each message type but Table 29 describes this connector in general and not a specific instance of the connector.

Table 29 – SunGuide Traffic Signals Pipeline

Pipeline Element	SunGuide Traffic Signals	
sunguide_traffic_signal	Type	Kafka Topic
	Source	SG-EV-DR
HDFS Connector	Type	Connector
	Description	Merges multiple XML files into an optimally sized files and stores the merged files to HDFS
	Source	sunguide_traffic_signal
	Source Format	XML (gzip compressed)

Pipeline Element	SunGuide Traffic Signals	
	Target File Name	<base_file_name>_<yyyyMMddhhmmss>.xml Where: <ol style="list-style-type: none"> base_file_name is the root element tag. For example, retrieveDataResp, statusUpdateMsg, etc. yyyyMMddhhmmss is the timestamp of the first source message to be stored in the target file. RICMS may merge multiple source messages into a single target file and there may be multiple target files per day.
	Target Format	XML
GIS Drop Folder Connector	Type	Connector
	Description	Transforms SunGuide XML to XML suitable for ESRI Geo-Event Server ingestion and saves XML files to drop folder accessible by Geo-Event-Server
	Source	sunguide_traffic_signal
	Source Format	XML (gzip compressed)
	Target File Name	traffic_signal_<yyyyMMdd_hhmmss>.xml
	Target Format	XML
XML to JSON Stream Processor	Type	Stream Processor
	Description	Transforms data from XML to JSON
	Source	sunguide_traffic_signal
	Source Format	XML (gzip compressed)
	Target Topic	sunguide_traffic_signal_json
	Target Format	JSON (gzip compressed)
sunguide_traffic_signal_json	Type	Kafka Topic
	Source	XML to JSON Stream Processor
MongoDB Connector	Type	Connector
	Description	Minimally transforms source JSON (i.e., date/time deserialization) and saves data to MongoDB
	Source	sunguide_traffic_signal_json
	Source Format	JSON (gzip compressed)
	Target Collections	sunguide.traffic_signal_config_current sunguide.traffic_signal_config_historical sunguide.traffic_signal_schedule_config_current sunguide.traffic_signal_schedule_config_historical sunguide.traffic_signal_timing_plan_config_current sunguide.traffic_signal_timing_plan_config_historical
	Target Format	BSON

3.4.1.2.2.6 SunGuide Ramp Meters (SG-RM-PL)

The structure of the SunGuide Ramp Meters Pipeline (SG-RM-PL) is described in Table 30. Note that there will be a separate instance of the HDFS Connector for each message type but Table 30 describes this connector in general and not a specific instance of the connector.

Table 30 – SunGuide Ramp Meters Pipeline

Pipeline Element	SunGuide Ramp Meters	
sunguide_ramp_meter	Type	Kafka Topic

System Design Document for Regional Integrated Corridor Management System (R-ICMS)

Pipeline Element	SunGuide Ramp Meters	
	Source	SG-EV-DR
HDFS Connector	Type	Connector
	Description	Merges multiple XML files into an optimally sized files and stores the merged files to HDFS
	Source	sunguide_ramp_meter
	Source Format	XML (gzip compressed)
	Target File Name	<base_file_name>_<yyyyMMddhhmmss>.xml Where: <ol style="list-style-type: none"> base_file_name is the root element tag. For example, retrieveDataResp, statusUpdateMsg, etc. yyyyMMddhhmmss is the timestamp of the first source message to be stored in the target file. RICMS may merge multiple source messages into a single target file and there may be multiple target files per day.
Target Format	XML	
GIS Drop Folder Connector	Type	Connector
	Description	Transforms SunGuide XML to XML suitable for ESRI Geo-Event Server ingestion and saves XML files to drop folder accessible by Geo-Event-Server
	Source	sunguide_ramp_meter
	Source Format	XML (gzip compressed)
	Target File Name	ramp_meter_<yyyyMMdd_hhmmss>.xml
Target Format	XML	
XML to JSON Stream Processor	Type	Stream Processor
	Description	Transforms data from XML to JSON
	Source	sunguide_ramp_meter
	Source Format	XML (gzip compressed)
	Target	sunguide_ramp_meter_json
Target Format	JSON (gzip compressed)	
sunguide_ramp_meter_json	Type	Kafka Topic
	Source	XML to JSON Stream Processor
MongoDB Connector	Type	Connector
	Description	Minimally transforms source JSON (i.e., date/time deserialization) and saves data to MongoDB
	Source	sunguide_ramp_meter_json
	Source Format	JSON (gzip compressed)
	Target Collections	sunguide.ramp_meter_config_current sunguide.ramp_meter_config_historical sunguide.ramp_meter_status_current sunguide.ramp_meter_status_historical
Target Format	BSON	

3.4.1.2.2.7 SunGuide Connected Vehicles (SG-CV-PL)

The structure of the SunGuide Connected Vehicles Pipeline (SG-CV-PL) is described in Table 31. Note that there will be a separate instance of the HDFS Connector for each message type but Table 31 describes this connector in general and not a specific instance of the connector.

Table 31 - SunGuide Connected Vehicles Pipeline

Pipeline Element	SunGuide Ramp Meters	
sunguide_cvs	Type	Kafka Topic
	Source	SG-CV-DR
HDFS Connector	Type	Connector
	Description	Merges multiple XML files into an optimally sized files and stores the merged files to HDFS
	Source	sunguide_cvs
	Source Format	XML (gzip compressed)
	Target File Name	<base_file_name>_<yyyyMMddhhmmss>.xml Where: <ol style="list-style-type: none"> 1. base_file_name is the root element tag. For example, retrieveDataResp, statusUpdateMsg, etc. 2. yyyyMMddhhmmss is the timestamp of the first source message to be stored in the target file. RICMS may merge multiple source messages into a single target file and there may be multiple target files per day.
	Target Format	XML
GIS Drop Folder Connector	Type	Connector
	Description	Transforms SunGuide XML to XML suitable for ESRI Geo-Event Server ingestion and saves XML files to drop folder accessible by Geo-Event-Server
	Source	sunguide_cvs
	Source Format	XML (gzip compressed)
	Target File Name	roadside_equipment_YYYYMMDD_hhmmss.xml
	Target Format	XML
XML to JSON Stream Processor	Type	Stream Processor
	Description	Transforms data from XML to JSON
	Source	sunguide_cvs
	Source Format	XML (gzip compressed)
	Target	sunguide_cvs_json
	Target Format	JSON (gzip compressed)
sunguide_cvs_json	Type	Kafka Topic
	Source	XML to JSON Stream Processor
MongoDB Connector	Type	Connector
	Description	Minimally transforms source JSON (i.e., date/time deserialization) and saves data to MongoDB
	Source	sunguide_cvs_json
	Source Format	JSON (gzip compressed)
	Target Collections	sunguide.rse_config_current sunguide.rse_config_historical sunguide.rse_status_current sunguide.rse_status_historical sunguide.tam_current sunguide.tam_historical
	Target Format	BSON

3.4.1.2.3 Transit AVL (TAVL-PL)

Transit AVL Pipelines (TAVL-PL) are pipelines that distribute real-time transit data including trip updates and vehicle positions by the Driver layer throughout the remainder of the R-ICMS architecture.

3.4.1.2.3.1 Transit AVL Current Data Pipeline (TAVL-CU-PL)

The Transit AVL Current Data Pipeline (TAVL-CU-PL) will provide a pipeline with incoming data from the GTFS Aggregator REST API providing transit AVL data to the R-ICMS. Since there are separate drivers to ingest trip updates and vehicle positions data there will also be separate pipelines to process and store each type of data. Figure 13 illustrates the general data flow through the pipeline but note that the GIS Drop Folder connector is only applicable for the vehicle positions data type.

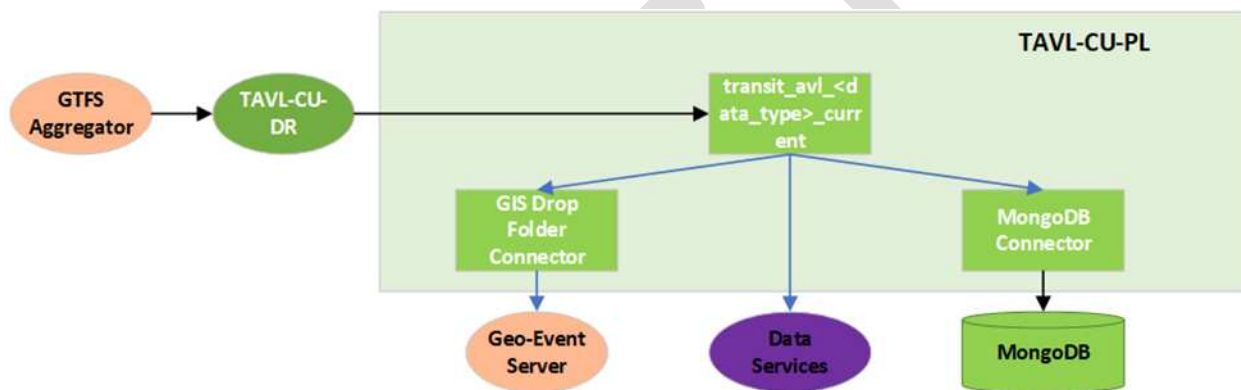


Figure 13 – Transit AVL Current Data Pipeline Diagram

Table 32 displays the Pipeline specifications for the transit AVL current data.

Table 32 – Transit AVL Current Data Pipeline

Pipeline Element	Transit Static Data	
transit_avl_<data_type>_current	Type	Kafka Topic <data_type> is either trip_updates or vehicle_positions
	Source	TAVL-CU-DR
MongoDB Connector	Type	Connector
	Description	Saves current Transit AVL data to MongoDB collections
	Source	transit_avl_<data_type>_current
	Source Format	JSON
	Target Collection	transit.avl_<data_type>_current
	Target Format	BSON
GIS Drop Folder Connector	Type	Connector
	Description	Transforms XML to XML suitable for Geo-Event-Server ingestion and saves files to drop folder accessible by Geo-Event-Server
	Source Topic	transit_avl_vehicle_positions_current
	Source Format	JSON (gzip compressed)

Pipeline Element	Transit Static Data	
	Target File Name	avl_yyyyMMdd_hhmmss.xml
	Target Format	XML

3.4.1.2.3.2 Transit AVL Archive Data Pipeline (TAVL-AR-PL)

The Transit AVL Archive Data Pipeline (TAVL-AR-PL) will provide a pipeline with incoming data from the GTFS Aggregator REST API providing transit AVL data to the R-ICMS. Since there are separate drivers to ingest trip updates and vehicle positions data there will also be separate pipelines to process and store each type of data. Figure 14 illustrates the general data flow through the pipeline but note that the GIS Drop Folder connector is only applicable for the vehicle positions data type.

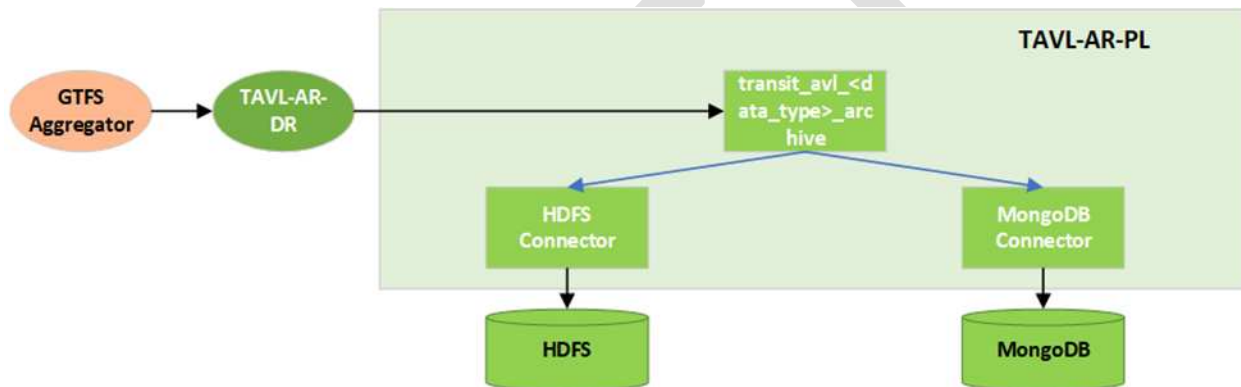


Figure 14 – Transit AVL Archive Data Pipeline Diagram

Table 33 displays the Pipeline specifications for the transit AVL archive data.

Table 33 – Transit AVL Archive Data Pipeline

Pipeline Element	Transit Static Data	
transit_avl_<data_type>_archive	Type	Kafka Topic <data_type> is either trip_updates or vehicle_positions
	Source	TAVL-AR-DR
MongoDB Connector	Type	Connector
	Description	Saves current Transit AVL data to MongoDB collections
	Source	transit_avl_<data_type>_archive
	Source Format	JSON
	Target Collection	transit.avl_<data_type>_historical
	Target Format	BSON
HDFS Connector	Type	Connector
	Description	Merges multiple JSON files into optimally sized files and stores the merged files to HDFS
	Source	transit_avl_<data_type>_archive
	Source Format	JSON (gzip compressed)

Pipeline Element	Transit Static Data	
	Target File Name	<base_file_name>_<yyyyMMddhhmmss>.json Where: <ol style="list-style-type: none"> base_file_name is tripUpdate or vehicleResponse (according to data_type) yyyyMMddhhmmss is the timestamp of the first source message to be stored in the target file. RICMS will merge multiple source messages into a single target file and there will be multiple target files per day.
	Target Format	JSON

3.4.1.2.4 Transit Static Data (TSD-PL)

The Transit Static Data Pipeline (TSD-PL) will provide a single pipeline with incoming data from the GTFS Aggregator REST API providing transit static data to the R-ICMS. Figure 15 illustrates the data flow through the pipeline.

As noted in the driver section, the retrieved feed data itself is too large to publish to a Kafka topic so a reference (file id) to the feed data in the MongoDB GridFS file storage system is published to the topic instead. Each processor and connector in the pipeline will need to retrieve the file from the GridFS bucket by file id.

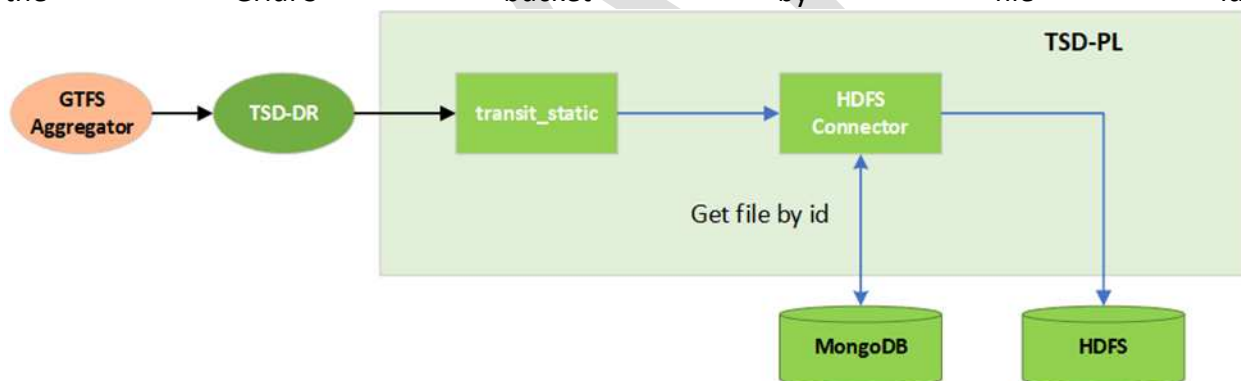


Figure 15 – Transit Static Data Pipeline Diagram

Table 34 displays the Pipeline specifications for transit static data.

Table 34 – Transit Static Data Pipeline

Pipeline Element	Transit Static Data	
transit_static	Type	Kafka Topic
	Source	TSD-DR
HDFS Connector	Type	Connector
	Description	Saves a single the data for a single agency to HDFS
	Source	transit_static (file id) MongoDB GridFS bucket transit_static (file content)
	Source Format	BSON
	Target File Name	<agency>_<yyyyMMdd>.json

Pipeline Element	Transit Static Data	
		Where yyyyMMdd is from the timestamp the feed data was last updated per the source system (GTFS Aggregator)
	Target Format	JSON

3.4.1.2.5 Signal Controller Log (SCL-PL)

The Signal Controller Log Pipeline (SCL-PL) will provide a single pipeline with incoming data from the signal controllers providing signal event code data to the R-ICMS. Figure 16 – Signal Controller Log Pipeline Diagram illustrates the data flow through the pipeline.

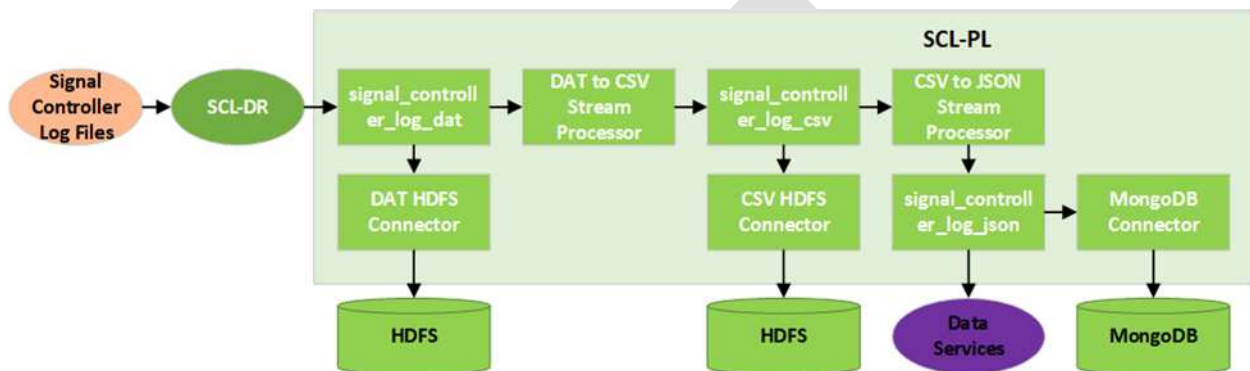


Figure 16 – Signal Controller Log Pipeline Diagram

Table 35 - Signal Controller Log Pipeline displays the Pipeline specifications for signal controller logs.

Table 35 - Signal Controller Log Pipeline

Pipeline Element	Signal Controller Log CSV	
signal_controller_log_dat	Type	Kafka Topic
	Source	SCL-CSV-DR
DAT HDFS Connector	Type	Connector
	Description	Archives multiple DAT files into a ZIP archive by date and saves the ZIP archive to HDFS
	Source	signal_controller_log_dat
	Source Format	Binary (vendor proprietary)
	Target File Name	<vendor>_<yyyyMMdd>_<now_utc>.zip Where: 1. yyyyMMdd are from the input DAT file name 2. now_utc is UTC datetime in form yyyyMMddhhmm at the time the target ZIP archive file was opened. Each ZIP archive file may contain many DAT files and there may be multiple ZIP archive files per day so including now_utc in the filename creates unique filenames.
	Target Format	ZIP archive
DAT to CSV Stream Processor	Type	Stream Processor
	Description	Decodes each DAT file into CSV by running the appropriate vendor-specific decoder
	Source	signal_controller_log_dat

System Design Document for Regional Integrated Corridor Management System (R-ICMS)

Pipeline Element	Signal Controller Log CSV	
	Source Format	Binary (vendor proprietary)
	Target	signal_controller_log_csv
	Target Format	CSV (gzip compressed)
signal_controller_log_csv	Type	Kafka Topic
	Source	DAT to CSV Stream Processor
CSV HDFS Connector	Type	Connector
	Description	Archives multiple CSV files into a ZIP archive by date and saves the ZIP archive to HDFS
	Source	signal_controller_log_csv
	Source Format	CSV
	Target File Name	<vendor>_<yyyyMMdd>_<now_utc>.zip Where: <ol style="list-style-type: none"> 1. yyyyMMdd are from the input DAT file name 2. now_utc is UTC datetime in form yyyyMMddhhmm at the time the target ZIP archive file was opened. Each ZIP archive file may contain many DAT files and there may be multiple ZIP archive files per day so including now_utc in the filename creates unique filenames.
	Target Format	ZIP archive
CSV to JSON Stream Processor	Type	Stream Processor
	Description	Converts decoded CSV signal controller log data to JSON
	Source	signal_controller_log_csv
	Source Format	CSV
	Target	signal_controller_log_json
	Target Format	JSON (gzip compressed)
signal_controller_log_json	Type	Kafka Topic
	Source	CSV to JSON Stream Processor
Mongo DB Connector	Type	Connector
	Description	
	Source	signal_controller_log_json
	Source Format	JSON
	Target Collection	signal_controller_log.signal_controller_log_historical
	Target Format	BSON

3.4.1.2.6 Signalized Intersections (SI-PL)

The Signalized Intersections Pipeline (SI-PL) will provide a single pipeline with incoming data from the Signalized Intersections Inventory Application REST API providing intersection data to the R-ICMS. Figure 17 illustrates the data flow through the pipeline.

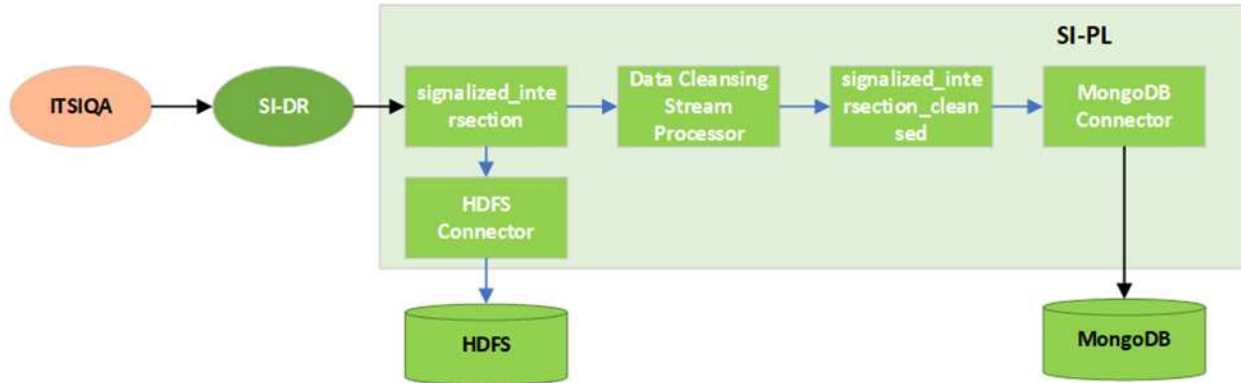


Figure 17 – Signalized Intersections Pipeline Diagram

The structure of the Signalized Intersections Pipeline (SI-PL) is described in Table 36.

Table 36 – Signalized Intersections Pipeline

Pipeline Element	Signalized Intersections	
sii	Type	Kafka Topic
	Source	SI-DR
HDFS Connector	Type	Connector
	Description	Merges multiple JSON files into an optimally sized files and stores the merged files to HDFS
	Source	sii
	Source Format	JSON
	Target	HDFS
Data Cleansing Stream Processor	Type	Stream Processor
	Description	Applies data conversions to prepare data for downstream processors/connectors.
	Source	sii
	Source Format	JSON (gzip compressed)
	Target	sii_json_transformed
MongoDB Connector	Type	Connector
	Description	Minimally transforms source JSON (i.e., date/time deserialization) and saves data to MongoDB
	Source	sii_json_transformed
	Source Format	JSON (gzip compressed)
	Target Collections	sii.intersection_current sii.intersection_historical
	Target Format	BSON

3.4.1.2.7 Weather Alerts (WEA-PL)

The Weather Alerts Pipeline (WEA-PL) will provide a single pipeline with incoming data from the National Weather Service REST API providing weather alert data to the R-ICMS. Figure 18 illustrates the data flow through the pipeline.

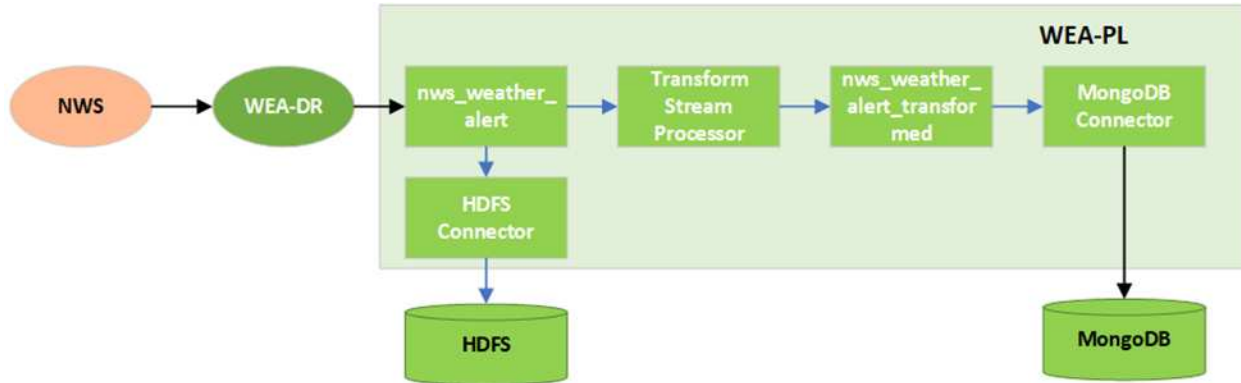


Figure 18 – Weather Alerts Pipeline Diagram

The structure of the Weather Alerts Pipeline (WEA-PL) is described in Table 37.

Table 37 – Weather Alerts Pipeline

Pipeline Element	Weather Alerts	
nws_weather_alert	Type	Kafka Topic
	Source	WEA-DR
HDFS Connector	Type	Connector
	Description	Merges multiple JSON files into an optimally sized files and stores the merged files to HDFS
	Source	nws_weather_alert
	Source Format	JSON
	Target	HDFS
	Target Format	JSON
Transform Stream Processor	Type	Stream Processor
	Description	Applies data conversions to prepare data for downstream processors/connectors.
	Source	nws_weather_alert
	Source Format	JSON (gzip compressed)
	Target	nws_weather_alert_transformed
	Target Format	JSON (gzip compressed)
MongoDB Connector	Type	Connector
	Description	Minimally transforms source JSON (i.e., date/time deserialization) and saves data to MongoDB
	Source	nws_weather_alert_transformed
	Source Format	JSON (gzip compressed)
	Target Collections	nws.weather_alert_current nws.weather_alert_historical
	Target Format	BSON

3.4.1.3 Data Store

R-ICMS application is dependent on various data sources and formats coming from different systems/sources. All the data collected from various sources need to be stored and made

available to the application/subsystems as needed. R-ICMS may transform the information to meet its needs, and then will need to store the information and make it available to external users via a REST API for a minimum of 3 years. The R-ICMS will make use of different storage methods to support the subsystem needs as well as future use of data for analytics within Big Data Hadoop eco-system. Following is the list of storage mechanisms being adopted for R-ICMS application:

1. **File Store:** R-ICMS will use the Hadoop Distributed File System (HDFS) to store all raw data coming from various data sources as binary files. This will enable Big Data components to build needed analytics applications at a later stage.
2. **NoSQL:** MongoDB will serve as NoSQL data store for system's operational use. The storage format will be JSON document model (MongoDB internally stores as proprietary BSON). All the data that fits write once and read many pattern will be stored into MongoDB.
3. **SQL Server:** R-ICMS will also make use of a Relational Database System – SQL Server to store all application related structural, transactional data as needed.
4. **GIS Data Store:** The primary UI operations of R-ICMS application will be using map data, which requires a GIS Data Store. R-ICMS application will make use of ArcGIS COTS map application for the operational use. All map data related to R-ICMS will be loaded into ArcGIS tool, which will serve the Map UI within the application.

Additional details about these data stores are described in the following sections:

3.4.1.3.1 File Store (FS-ST)

Whole files will be stored in the File Store. Example of whole files that may be stored in the data store include signal timing plans, signed signal timing plans, and GIS files received from other systems. The File Store will also be used for long term storage and distributed analytics access for the data streams received via the pipelines.

R-ICMS will use the HDFS as the File Store. While this file system serves as basic storage for files, it also allows systems to utilize these files to access, process (map reduce jobs) and run analytics very efficiently within the Hadoop stack of tools. The Hadoop eco-system provides the capabilities for advanced analytics, machine learning and Artificial Intelligence. Proper planning and design of the file store enables these capabilities for future use. One important consideration during implementation of the R-ICMS is that storing very small files in large numbers is against the principle of HDFS storage due to the block size (128MB default) it uses for storage.

For the initial version of the R-ICMS, the Hadoop system will be used as a basic “data lake” where the system will bring in various formats of data and store it in one place. Once the data is being collected, users can begin evaluating the value of the data and design applications to utilize the data; applying the advanced capabilities like machine learning and artificial intelligence.

For R-ICMS, the data will be organized in HDFS as described below:

- The base folder will be “/user” by default
- The data source or system name will be the next level of hierarchy

- Each data source will be designated as a separate folder with the data source name, which will group the data files together. Within the data source, chronological order will be assigned with sub-folders (year, month and day)
- The base Uniform Resource Locator (URL) for storage will be based on the parameters listed above
- To avoid internal fragmentation caused by HDFS large block sizes, messages will be combined with the date and timestamp of the filename to be the first message in that file. Message will be in the original format received with the exception of the header; binary files will be copied as binary, compressed will be concatenated compressed files, text files will be concatenated text files.
 - When an HDFS block fills, a new file will be created. The size of the HDFS blocks are an attribute of the HDFS file system, not a configuration parameter of the R-ICMS application.
 - A new file will be created whenever the day changes
- While the messages are being appended to the file, the file name will have an underscore at the beginning of the file so that HDFS doesn't process the file until the file has been closed.
- On close of the file, it will be renamed to remove the underscore

For example, ITSQA traffic conditions files typical storage file names include:

/user/itsqa/<data_set_name>/<year>/<month>/<day>/<data_set_name>_<date>_<time>.xml

/user/itsqa/link_config/2018/07/25/link_config_20180725_161020.xml

/user/itsqa/traffic_data/2018/07/25 /traffic_data_20180725_161020.xml

/user/itsqa/class_data/2018/07/25/class_data_20180725_161020.xml

For GIS data files, the base URL will be

/user/gis/school_zones/2018/07/25/school_zones_20180725_161020.zip

/user/gis/basemap/2018/07/25/basemap_20180725_161020.zip

The storage locations for each data source are documented under respective data source driver or pipeline sections.

3.4.1.3.2 NoSQL (NOS-ST)

MongoDB will be used for R-ICMS data storage where applicable. MongoDB is a NoSQL database using the JSON document model and is most suitable for semi-structured and sensor-based information.

MongoDB supports the use of multiple databases allowing the design to have the flexibility of storing the data in different databases depending on the type and subject of the data.

The data from each data source will be stored in a separate database and data sets within the data source will be stored in separate collections.

Storage pattern in MongoDB:

1. Each data source will be located in a separate database (e.g., SunGuide, ITSQA)
2. The sub-sets of data source can be further divided into collections, or have additional collections depending on the need for segregating and reformatting the data for application needs

Within the database, a separate collection will be configured to store each feed type. For example:

- all_sources_traffic_data_historical
- no_here_link_config_current

Additional collections will be created to store merged/reformatted data to support queries needed by the APIs and other clients. These collections will evolve during the course of later iterations and will be documented within the respective iteration.

3.4.1.3.3 SQL Database (SQL-ST)

The SQL database will be the District's Microsoft SQL Server 2016 Enterprise Edition. Transactional data from within the R-ICMS will be stored in the SQL database.

3.4.1.3.4 GIS Database (GIS-ST)

The GIS Database, GIS-ST, is internal to the ArcGIS server external application used to generate the User Interface Map for the R-ICMS application. The GIS-ST receives data in one of two ways.

- Static or slowly changing data is loaded manually through the ArcGIS user interface. This data includes the ARBM, School zones, emergency responder locations, etc. This data is received or gathered from external sources by the system administrator as ESRI shape files, converted to a common projection, if necessary and loaded into the system.
- Certain DFE pipelines write XML files to a shared folder that is watched by the GoeEvent Service. The GeoEvent services processes each XML file into the GIS Database and deletes the file from the shared folder.
- Figure 19 shows how the incoming data will be used and stored in the GIS data store.
- Figure 19 illustrates the position of the GIS-ST within the ArcGIS Server software.

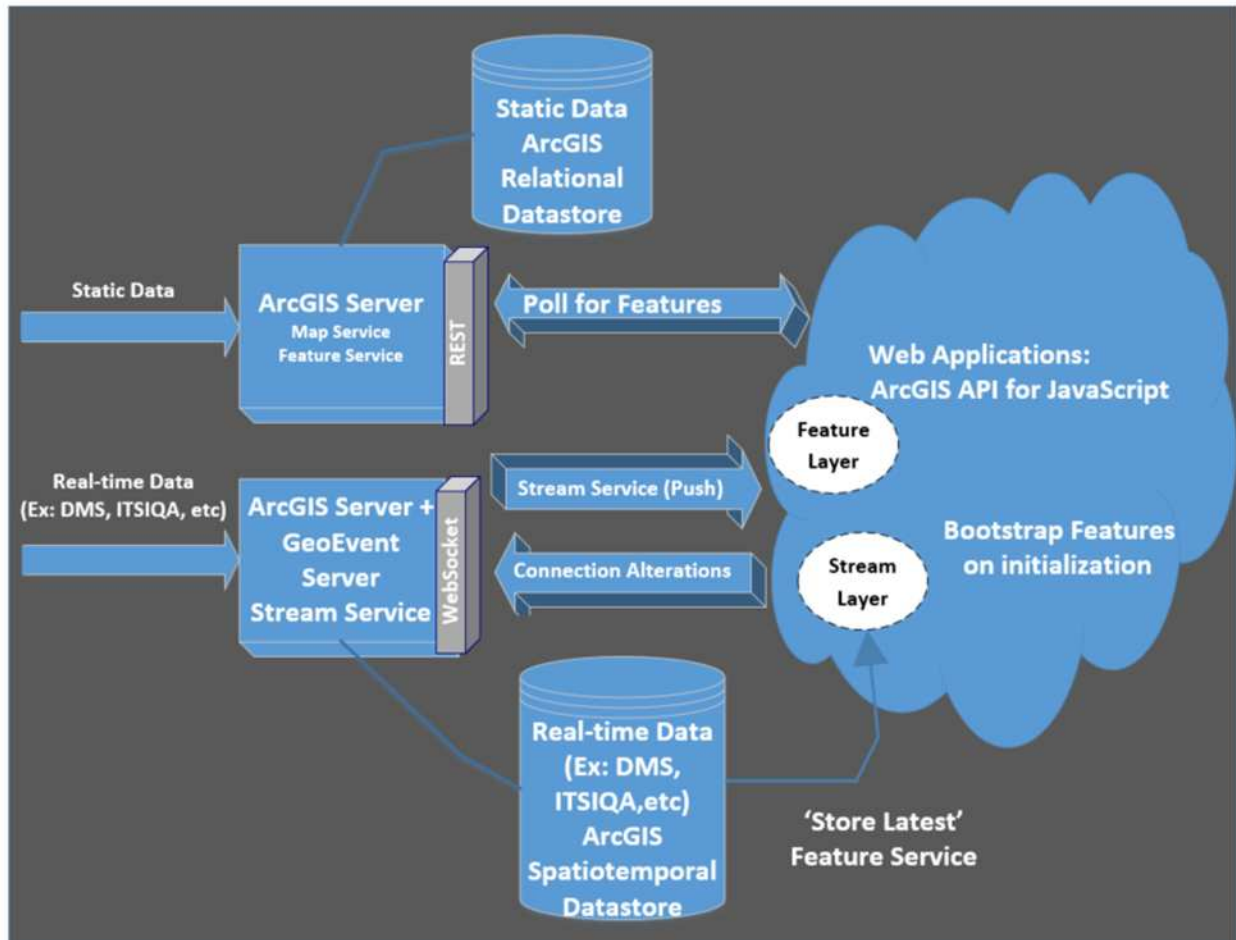


Figure 19 – GIS Data Store

The “Store Latest” option will enable the user to either choose from an existing feature or publish a new feature service, to use as a cache for maintaining the most recent event record for each uniquely received LINKID. To make use of this capability, the ArcGIS Server will use a managed enterprise GIS database, which will be created using ArcGIS Desktop or a spatiotemporal, big data store as a module of the ArcGIS Enterprise edition.

The data flow for shape files and Geodata stores is illustrated in Figure 20 and describe in the following steps.

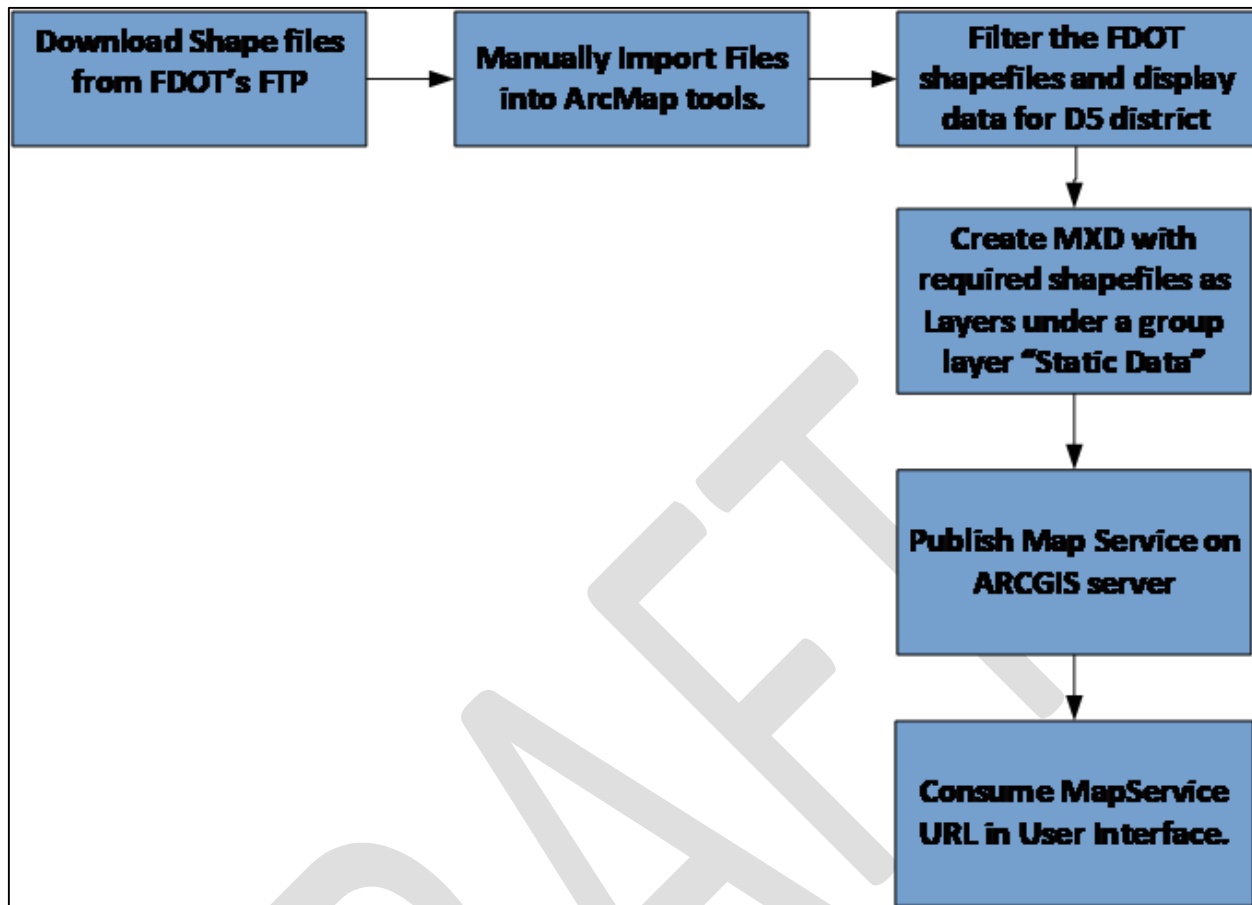


Figure 20 – Geodata Shape File Data Flow

The following procedures below will be used for publishing Geodata into the ArcGIS Server:

1. The shape files will be manually downloaded from FDOT's FTP site
2. The downloaded files will be unzipped to a local staging area that will be specified in operational manuals.
3. A .mxd (map service) (example: R-ICMS_StaticData.MXD) will be created using ArcMap for each shape file as a layer under the corresponding groups as shown in Figure 21
4. The Spatial Analysis tools within the ArcToolbox will be used to analyze the layers and filter display data corresponding to FDOT D5
5. Projections will be updated to 'NAD 1983 UTM Zone_17n' for projecting all data sources.
6. A new enterprise geodatabase named 'R-ICMS_StaticData' will be created using ArcMap/ArcCatalog. A connection will be established to R-ICMS_StaticData.gdb using Catalog from the ArcGIS for Desktop application. The newly created enterprise geodatabase will be registered as a data store in the ArcGIS Server. The map image tiles from ArcMap will be published as a map service using 'Share as Service'.
7. A URL will be created for the published map service that will be further accessed using the ArcGIS Server Manager. The created REST URL will be consumed using corresponding methods from the ESRI JS API 3.25 user interface.

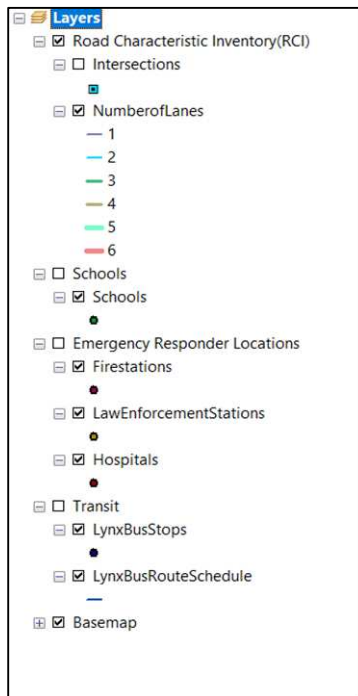


Figure 21 – Map Layer Shape File

3.4.1.4 Data Services

R-ICMS data services provide both streaming and query access to data stored within R-ICMS for both internal and external R-ICMS users. Data services supporting streaming data are responsible for maintaining a cache of the current status of the data for which they are responsible. Consumers of data services data will have the capability of setting up subscriptions to receive streaming updates as well as receiving the current state of the data. Additionally, through the use of defined filters, data consumers will have the capability to query historical data. Identified data services for use in the R-ICMS are listed below. There are three types of data services within the R-ICMS architecture:

1. Generic Data Services provide access to the transformed, or combined data collected from the data sources
2. GIS Data Services provide access to data that can be displayed on a map by the user interface
3. R-ICMS Data Services provide access to data generated by the R-ICMS itself (including data generated by the Modeling Engine, where appropriate)

3.4.1.4.1 Generic Data Services

Each Data service provides two types of services:

1. REST or Historical Service: This service will return historical data as per the filter criteria passed to it
2. Subscription Service: This service provides updates to which the client has subscribed

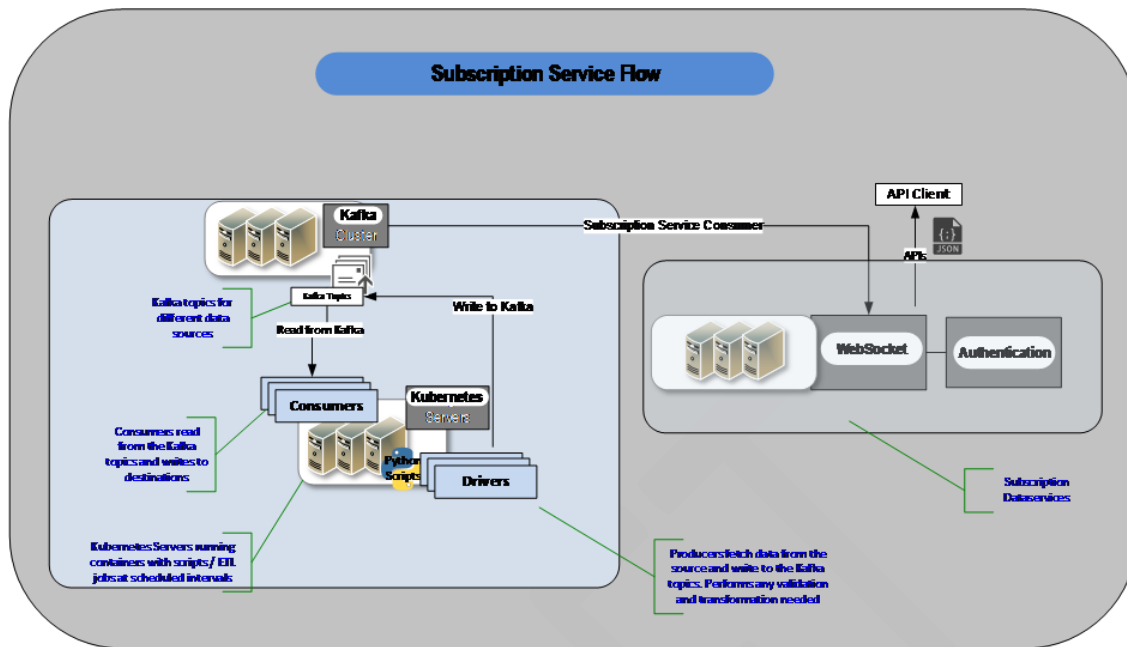


Figure 22 – Subscription Service

The R-ICMS will provide access to real time data using Subscription Services. Clients will be able to subscribe to real time data using WebSockets. The subscription services consumer will read the data from the Kafka topic and will send the data back to the subscribed client.

Authorization Services will be used to provide secure access to these WebSocket based API's.

For each type of data received from external systems, a data service will be provided to allow other services and systems access to the data. Filters will be defined for each data source to limit the way in which each type of data can be accessed. Types of generic data services are listed in Table 38 – Generic Data Services.

Table 38 – Generic Data Services

Data Service	Name	Description
SunGuide - DMS	SG-DMS-DS	Provides the status and messages on DMS devices
SunGuide – Events	SG-EV-DS	Provides traffic event data within the corridor from SunGuide
SunGuide – CCTV	SG-CCTV-DS	Provides the active status of Closed-circuit television (CCTV) devices
SunGuide – Truck Parking	SG-TP-DS	Provides configuration and status of truck parking facilities including total and available parking spots.
ITSIQA - Traffic Condition	TRF-DS	Provides traffic condition data for the corridor from the ITSIQA feed, including intersection movement count (IMC) data
Ramp Meter Status	SG-RM-DS	Provides the active status of ramp meter devices
Connected Vehicle Unit Status	SG-CV-DS	Provides the status of the connected vehicle RSEs and TAMs deployed in the network
Transit (GTFS) Static Data	GTFS-DS	Provides the static GTFS data.
Transit AVL (Various Systems)	TAVL-DS	Provides the AVL status for various transit systems (LYNX, SunRail etc.)
Origin Destination Data	OD-DS	Provides Origin Destination data

Data Service	Name	Description
Signal Controller Settings	SC-SET-DS	Provides signal controller settings, phase timing, coordination schedules, and status information from signal vendors via SunGuide
Signal Controller Logs	SC-LOG-DS	Provides decoded traffic signal controller log events from the ATSPM system
School Schedule Data	TBD	Provides school schedule information
Weather Data	WEA-DS	Provides weather alert data collected from external sources
Intersection Inventory	SI-DS	Provides information from the FDOT Intersection Inventory Application (SIIA)

The generic Data Services developed by iteration include:

- Iteration 1:
 - SunGuide - DMS (SG-DMS-DS)
- Iteration 2:
 - SunGuide – CCTV (SG-CCTV-DS)
 - SunGuide – Event (SG-EV-DS)
 - Signal Controller Logs (SC-LOG-DS)
 - Signal Controller Settings (SC-SET-DS)
 - ITSIQA (TRF-DS)
- Iteration 3:
 - SunGuide – Truck Parking (SG-TP-DS)
 - Ramp Meter Status (SG-RM-DS)
 - Connected Vehicle Unit Status (SG-CV-DS)
 - Intersection Inventory (SI-DS)
- Iteration 4
 - Transit (GTFS) (GTFS-DS)
 - Origin Destination Data (OD-DS)
 - Transit AVL (TAVL-DS)

3.4.1.4.2 GIS Data Services

Two types of GIS data services will be made available for the purpose of providing geospatial data. These services will be responsible for sending the correct map tiles to the user interface depending on the user’s current zoom level and viewing location. The services are described in Table 39 – GIS Data Services.

Table 39 – GIS Data Services

Data Service	Description
GeoEvent Service	Provides status information for map features (e.g., DMS messages). This service provides the dynamic information for the static features provided by the ArcGIS Data Service
ArcGIS Service	Provides map tiles and feature (e.g., DMS location) information. This service provides the effectively static (rarely updating) data for rendering maps

3.4.1.4.2.1 GeoEvent Server (GEO-DS)

Real-time data will be streamed to a map page using the GeoEvent Server and web sockets. The real time data is output from Pipelines as noted in that section, it includes traffic data, SunGuide ITS device data, GTFS real time data (from the data aggregator), etc. The GeoEvent Server Flow is illustrated in Figure 23 – GeoEvent Server Flow.

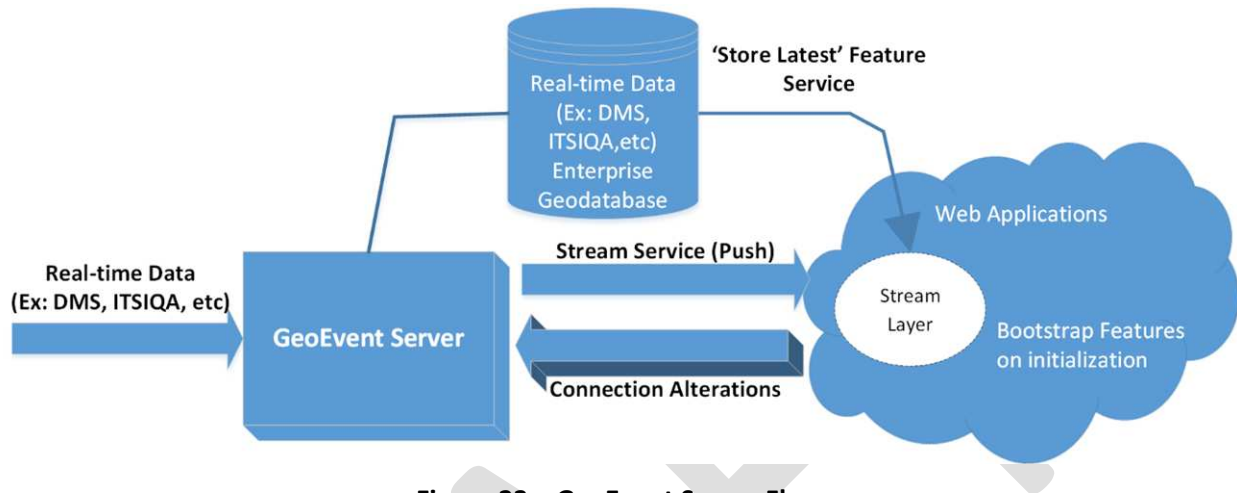


Figure 23 – GeoEvent Server Flow

Once the feed definition is created, an output connector will be created to push data from Kafka feed into the stream service. The default update interval (in seconds) of 0.01 seconds specifies that events routed to the pipeline will be batched and broadcasted one hundred times each second, which must be updated on the map within three (3) seconds for the ITSQA data.

A stream service will be published, and an output connector will be created. Stream services leverage WebSocket technology, which supports full-duplex, bidirectional communication. This enables clients to specify the data they want to receive without having to unsubscribe and reestablish their connection to the service. Clients can filter stream service data by specifying either spatial or attribute constraints. After creating the input and output connectors, a GeoEvent service will be published. The real-time data will flow from the pipeline to the service layer and will be consumed by the GeoEvent service.

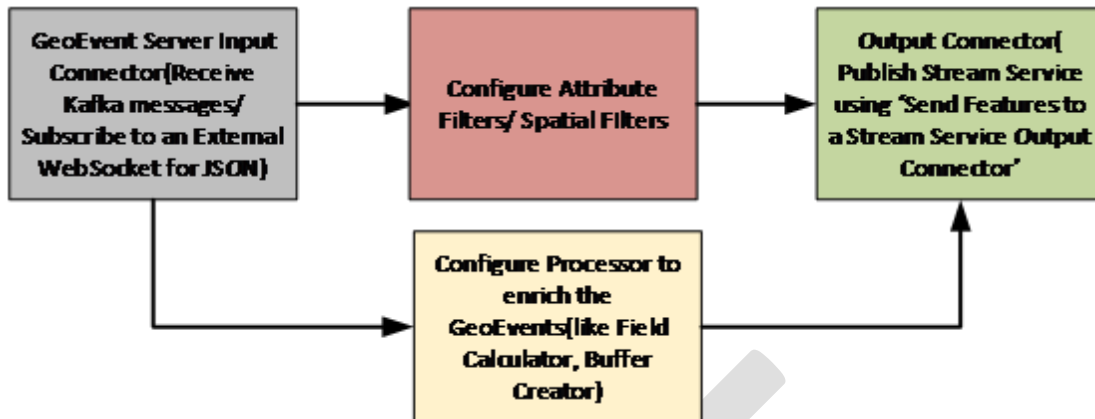


Figure 24 – Real-time GeoEvent Data Flow

In ArcGIS server manager, a Stream service will be deployed and the service endpoint (REST API) will be consumed from the user interface map page. Table 40 – Stream Service Details shows the stream service API details:

Table 40 – Stream Service Details

Title	<i>Streaming Service Layers</i>
URLs	Actual URLs TBD with deployment server configuration, samples below: DMS URL: url = 'https://<hostname>/<webadaptor>/rest/services/dms/StreamServer' ITSIQA URL: url = 'https://<hostname>/<webadaptor>/rest/services/itsiqa/StreamServer' Event URL: url = 'https://<hostname>/<webadaptor>/rest/services/event/StreamServer' Camera URL: url = 'https://<hostname>/<webadaptor>/rest/services/camera/StreamServer'
Method	GET
URL Params	NA
Data Params	NA
Success Response	The service request will respond with a status code of 200, and the content of the StreamLayer will be rendered in the map view.
Error Response	Initial error request page refresh from user. After consecutive/repeated errors, notify system admin to review system logs.
Sample Call	var url = 'https://<hostname>/<webadaptor>/rest/services/dms/StreamServer' var layer = new StreamLayer(url);
Notes	Note: replace <hostname> and <webadaptor> with deployment configuration values.

3.4.1.4.2.2 ArcGIS Service

A REST API based FeatureLayer service, part of the ArcGIS data service layer, will be made available from the ArcGIS server. This API method will be used for fetching static data from the ArcGIS map service. The FeatureLayer service is described in Table 41 – ArcGIS Feature Service API Details.

Table 41 – ArcGIS Feature Service API Details

Title	<i>Static Feature Layers</i>
URL	Actual URL TBD with deployment server configuration, sample below: Static Layer URL: url = 'https://<hostname>/<webadaptor>/rest/services/rcims/MapServer'
Method	GET

Title	<i>Static Feature Layers</i>
URL Params	NA
Data/Params	<p>Fire Stations</p> <ul style="list-style-type: none"> • NAME (type: esriFieldTypeString , alias: Name , length: 60) • Phone (type: esriFieldTypeString , alias: Phone , length: 15) • ADDRESS (type: esriFieldTypeString , alias: Address , length: 254) • CITY (type: esriFieldTypeString , alias: City , length: 20) • County (type: esriFieldTypeString , alias: County , length: 50) • STATE (type: esriFieldTypeString , alias: State , length: 2) • ZIP (type: esriFieldTypeString , alias: Zip , length: 254) • LoadDate (type: esriFieldTypeDate , alias: Load Date , length: 8) • Latitude (type: esriFieldTypeDouble , alias: Latitude) • Longitude (type: esriFieldTypeDouble , alias: Longitude) <p>Law Enforcement</p> <ul style="list-style-type: none"> • NAME (type: esriFieldTypeString , alias: Name , length: 60) • Type (type: esriFieldTypeString , alias: Type , length: 20) • PHONE (type: esriFieldTypeString , alias: Phone , length: 14) • ADDRESS (type: esriFieldTypeString , alias: Address , length: 75) • CITY (type: esriFieldTypeString , alias: City , length: 40) • County (type: esriFieldTypeString , alias: County , length: 50) • STATE (type: esriFieldTypeString , alias: State , length: 2) • Zip (type: esriFieldTypeInteger , alias: Zip) • LoadDate (type: esriFieldTypeDate , alias: Load Date , length: 8) • LATITUDE (type: esriFieldTypeDouble , alias: Latitude) • LONGITUDE (type: esriFieldTypeDouble , alias: Longitude) <p>Healthcare</p> <ul style="list-style-type: none"> • NAME (type: esriFieldTypeString , alias: Name , length: 60) • FacilityTy (type: esriFieldTypeString , alias: Type , length: 70) • PHONE (type: esriFieldTypeString , alias: Phone , length: 14) • ADDRESS (type: esriFieldTypeString , alias: Address , length: 75) • CITY (type: esriFieldTypeString , alias: City , length: 40) • County (type: esriFieldTypeString , alias: County , length: 50) • STATE (type: esriFieldTypeString , alias: State , length: 2) • ZIPCODE (type: esriFieldTypeString , alias: Zip , length: 10) • LoadDate (type: esriFieldTypeDate , alias: Load Date , length: 8) • LATITUDE (type: esriFieldTypeDouble , alias: Latitude) • LONGITUDE (type: esriFieldTypeDouble , alias: Longitude) <p>Schools</p> <ul style="list-style-type: none"> • NAME (type: esriFieldTypeString , alias: Name , length: 100) • TYPE (type: esriFieldTypeString , alias: Type , length: 35) • OPERATING (type: esriFieldTypeString , alias: Operated By , length: 75) • PHONE (type: esriFieldTypeString , alias: Phone , length: 25) • ADDRESS (type: esriFieldTypeString , alias: Address , length: 65) • CITY (type: esriFieldTypeString , alias: City , length: 40) • COUNTY (type: esriFieldTypeString , alias: County , length: 20) • State (type: esriFieldTypeString , alias: State , length: 30) • ZIPCODE (type: esriFieldTypeDouble , alias: Zip) • LoadDate (type: esriFieldTypeDate , alias: Load Date , length: 8) • LAT_DD (type: esriFieldTypeDouble , alias: Latitude)

Title	<i>Static Feature Layers</i>
	<ul style="list-style-type: none"> • LONG_DD (type: esriFieldTypeDouble , alias: Longitude) <p>School Zones</p> <ul style="list-style-type: none"> • DESCR (type: esriFieldTypeString , alias: Road , length: 40) • Speed (type: esriFieldTypeString , alias: Speed Limit , length: 254) • City (type: esriFieldTypeString , alias: City , length: 30) • COUNTY (type: esriFieldTypeString , alias: County , length: 12) • State (type: esriFieldTypeString , alias: State , length: 30) • LoadDate (type: esriFieldTypeDate , alias: Load Date , length: 8) <p>Lynx Stops</p> <ul style="list-style-type: none"> • stop_name (type: esriFieldTypeString , alias: Name , length: 254) • stop_id (type: esriFieldTypeString , alias: Stop ID , length: 254) • ICMS_Date (type: esriFieldTypeDate , alias: Load Date , length: 8) • stop_lat (type: esriFieldTypeString , alias: Latitude , length: 254) • stop_lon (type: esriFieldTypeString , alias: Longitude , length: 254) <p>Lynx Routes</p> <ul style="list-style-type: none"> • rt_long_nm (type: esriFieldTypeString , alias: Name , length: 254) • rt_shrt_nm (type: esriFieldTypeString , alias: Route # , length: 254) • ICMS_Date (type: esriFieldTypeDate , alias: Load Date , length: 8) <p>Sunrail Stops</p> <ul style="list-style-type: none"> • stop_name (type: esriFieldTypeString , alias: Name , length: 254) • stop_id (type: esriFieldTypeString , alias: Stop ID , length: 254) • stop_lat (type: esriFieldTypeString , alias: Latitude , length: 254) • stop_lon (type: esriFieldTypeString , alias: Longitude , length: 254) • ICMS_Date (type: esriFieldTypeDate , alias: Load Date , length: 8) <p>Sunrail Routes</p> <ul style="list-style-type: none"> • rt_long_nm (type: esriFieldTypeString , alias: Name , length: 254) • route_id (type: esriFieldTypeString , alias: Route ID , length: 254) • rt_shrt_nm (type: esriFieldTypeString , alias: Route # , length: 254) • ICMS_Date (type: esriFieldTypeDate , alias: Load Date , length: 8) <p>Local Roads</p> <ul style="list-style-type: none"> • ST_NAME (type: esriFieldTypeString , alias: Name , length: 240) • LINK_ID (type: esriFieldTypeDouble , alias: Link ID) • ROADWAY (type: esriFieldTypeString , alias: Roadway , length: 8) • ROADSIDE (type: esriFieldTypeString , alias: Roadside , length: 1) • BMP (type: esriFieldTypeDouble , alias: BMP) • EMP (type: esriFieldTypeDouble , alias: EMP) • FUNC_CLASS (type: esriFieldTypeString , alias: Functional Class , length: 1) • SPEED_CAT (type: esriFieldTypeString , alias: Speed Category , length: 1) • DIR_TRAVEL (type: esriFieldTypeString , alias: Direction , length: 1) • EXPR_LANE (type: esriFieldTypeString , alias: Express Lane , length: 1) • FDOT_DISTR (type: esriFieldTypeInteger , alias: District) • COUNTY (type: esriFieldTypeString , alias: County , length: 4) • ICMS_Date (type: esriFieldTypeDate , alias: Load Date , length: 8)
Success Response	The service request will respond with a status code of 200, and the content of the MapServer will be rendered in the map view.
Error Response	Initial error request page refresh from user. After consecutive/repeated errors, notify system admin to review system logs.
Sample Call	var url = 'https://<hostname>/<webadaptor>/rest/services/rcims/MapServer'

Title	Static Feature Layers
	var layer = new MapImageLayer(url);
Notes	Note: replace <hostname> and <webadaptor> with deployment configuration values.

Basemaps published in the ArcGIS server will provide the foundation to visualize and analyze data using the ArcGIS JavaScript API from the user interface. Basemaps will be configured as a vector tile service, which will be an ArcGIS Server web service originating from a vector tile package in ArcGIS Pro. This vector tile service (also known as vector tile layers) will be used to share and consume vector tiles to the map page in a web user interface. Vector tile layers are rendered together on a map and can be individually toggled on or off. Table 42 – ArcGIS Vector Layer Server API describes the API.

Table 42 – ArcGIS Vector Layer Server API

Title	Basemap & Vector Tile Layers
URL	Actual URL TBD with deployment server configuration, sample below: Basemap URL: url = 'https://<hostname>/<webadaptor>/rest/services/ricms_base/VectorTileServer'
Method	GET
URL Params	NA
Data Params	NA
Success Response	The service request will respond with a status code of 200, and the content of the MapServer will be rendered in the map view.
Error Response	Initial error request page refresh from user. After consecutive/repeated errors, notify system admin to review system logs.
Sample Call	var url = 'https://<hostname>/<webadaptor>/rest/services/ricms_base/VectorTileServer ' var layer = new VectorTileLayer(url);
Notes	Note: replace <hostname> and <webadaptor> with deployment configuration values.

3.4.1.4.3 R-ICMS Data Services

The R-ICMS data services are responsible for enabling communication between the R-ICMS Business Services and the Data Stores. Any Business Service that needs to store/retrieve data will have an associated Data Service that supports that communication. Additionally, R-ICMS Data Services will make data generated from within the R-ICMS available for consumption. The following data services will be provided for other systems and components to access R-ICMS generated data.

Table 43 – R-ICMS Data Services

Data Service	Name	Description
Predictive Engine Data	TBD	Provides data produced by the Predictive Engine (e.g., modeling engine inputs and results)
Response Plan Data	TBD	Provides configuration and data produced by the Response Plan Business Service; e.g., business rule execution, business rule initiated response plan evaluations
Evaluation Engine Data	TBD	Provides data updates regarding evaluation results (e.g., Measures of Effectiveness (MOE) for evaluated response plans based on current data

Data Service	Name	Description
		and mesoscopic modeling projections of network impacts such as total delay, link delays, volumes of various types of vehicles)
Authorization	AuthZ-DS	Backs the Authorization Business Service
Notification	NOT-DS	Backs the Notification Business Service
Event Data	TBD	Provides event data and updates
SOT Data	SOT-DS	Backs the SOT Business Service and provides SOT data and updates
Metadata	TBD	Provides metadata regarding the data sources

3.4.2 DSS / Business Services

The R-ICMS will include various business services in order to achieve the desired overall functionality of the system. Business services will be transactional in nature and will serve as an intermediary between the user interface and the big data environment. Business services may connect to data services to receive streaming data or may interface directly with pipelines if needed to ensure adequate response to changing conditions.

Like the R-ICMS data services, APIs for the business services will be fully defined and documented in the source code repository using the OpenAPI Specification (OAS) format. The following identified business services for use in the R-ICMS are listed below.

3.4.2.1 Authentication (AuthN-BS)

The Authentication Business Service provides functionality to login users, refresh user tokens, logout users, and change user passwords.

Users are authenticated through the FDOT D5 Active Directory. This means that users of the R-ICMS system must possess accounts on either the FDOT D5 domain or a domain that the FDOT D5 domain has a trust relationship with. When logging in, it will be assumed that the user is a member of the FDOT D5 domain unless otherwise specified, i.e., <domain>\<username>.

When logging in, or when the JSON Web Token (JWT) is refreshed, the AuthN-BS will retrieve the account status of a user via Active Directory. The status can be one of the following:

- Normal: There is nothing out of place with the user's account. The user will be logged in or the token will be renewed.
- Locked: The account has been locked out of Active Directory, possibly because of too many incorrect login attempts. The user will not be logged in, or in the case of a token refresh, the refresh will be rejected, and the user will be logged out.
- Disabled: The account has been disabled in Active Directory. The user will not be logged in or the refresh will be rejected, and the user will be logged out.
- Password Expired: The password of the user account has expired. The user will be redirected to the change password page.
- Not Found: The user was not found in Active Directory (an existing user might have been removed). The user will be logged out.

If a user logs in with valid credentials and a valid account status, the AuthN-BS will build a JWT and return it to the user. Because the JWT is set to expire after a set period of time, the user interface will be configured to periodically send a token refresh request to the AuthN-BS. This will cause the AuthN-BS to build a new JWT, invalidating the old one, and return this JWT to the user.

The JWT is stored in the user's browser and maintains the user's session. All requests that the user makes to business or data services include this token, which is used to authenticate and authorize the user.

The JWT itself consists of a set of claims. Included in these claims are:

- **Expiration Time:** After a configured period of time, a user's JWT will become invalid. This mitigates the risk of a user's JWT being stolen by a malicious adversary.
- **Signing Credentials:** To protect the integrity of the JWT, the entire contents of the JWT are signed with a secret key, and this signature is appended to the JWT. Each JWT uses a randomly generated signing key. The signature itself is a SHA256 HMAC of the contents of the JWT. Every JWT, along with its respective signing key, is stored in an in-memory database to be retrieved when the JWT is validated.
- **Permissions and Devices:** The JWT contains every permission and device to which the user has been granted access. This allows any business or data service to authorize a user's action simply by checking that user's JWT.

3.4.2.1.1 Windows Login API (WLA-BS)

When a user tries to log in, the system will be capable of distinguishing between a user with invalid credentials, a user whose account has been disabled or locked out, and a user with an expired password. Microsoft Active Directory does not communicate these details through the LDAP protocol. Therefore, the system will use a Windows API call, LogonUser, to validate a user's credentials. This API call requires that the calling server be on the same domain as the Active Directory it is validating against. Docker containers cannot be joined to a domain, which means that the API call itself must be located in a service outside the Kubernetes and Docker Swarm infrastructure. For that reason, there will be a small service, the WLA-BS, whose sole purpose is to call two Windows API methods, one to authenticate a user's credentials and one to change a user's password.

3.4.2.1.2 Active Directory Library

With the exception of validating credentials and changing passwords, which must be done in the WLA-BS as described above, all interfacing with Active Directory will be done through one library. This library will communicate with the FDOT domain via LDAP, to retrieve the statuses and groups associated with users. The library will also be used by the AuthN-BS as well as the Authorization Business Service (AuthZ-BS).

3.4.2.2 Authorization (AuthZ-BS)

The Authorization Business Service provides functionality for handling roles and device groups, as well as for modifying approval profiles. Note that the actual actions of reading from and writing to the database are handled by the Authorization Data Service. On the other hand, the AuthZ-BS oversees the business logic required for these functions. Though the AuthZ-DS will expose endpoints for mutating roles and device groups, the only thing allowed to hit these endpoints is the AuthZ-BS. For example, while both the AuthZ-DS and the AuthZ-BS expose endpoints to add roles, the AuthZ-BS will be the only service accessing the AuthZ-DS Add Role endpoint. All other services will use the AuthZ-BS Add Role endpoint.

3.4.2.2.1 Permissions, Devices, Roles and Device Groups

Authorization in R-ICMS revolves around two concepts, permissions and devices. A permission represents the ability to conduct an action. For example, the ability to approve response a plan is a permission. Several actions associated with the system involve physical devices. For example, the previously mentioned response plan is associated with a set of signal controllers. In order to approve a subset of a response plan, an R-ICMS user must hold both any permissions required that are associated with taking an action, and if applicable, any devices associated with taking that action.

However, users of the system are not assigned permissions and devices directly. Instead, there exist roles and device groups. A role is a set of permissions, and similarly, a device group is a set of devices. Generally, device groups will correlate with the devices managed by an agency. The users of the system will be assigned to both roles and device groups. When a user attempts to conduct a protected action, the system will retrieve the permissions and devices associated with that user's roles and device groups and determine whether the action is allowed.

To facilitate ease of use, assignment of users to roles and device groups will be done via the FDOT D5 Active Directory system. The actual management of roles and device groups, including adding and removing permissions and devices, will be handled within the authorization-related services of R-ICMS.

3.4.2.2.2 Approval Profiles

The authorization services also provide functionality regarding assigning and modifying approval profiles. An approval profile is attached to a device or device group and enables the automatic approval or rejection of response plans involving those devices after a configurable interval of time.

For any given time, each approval profile has a defined status. There are four possible statuses:

1. **Normal:** A normal status does not have any effect. When an approval profile is first created, which happens when a new device or device group is introduced, the status is set to normal. If a device approval profile status is set to normal, it inherits its status from its device group.

2. **Auto-approve:** If the devices associated with a response plan have the auto-approve status, then after a configurable delay without being approved, the response plan will be approved automatically.
3. **Auto-reject:** Analogous to auto-approve, if a response plan's devices are set to auto-reject, then after a configurable interval, the response plan will be automatically rejected if it has not yet been approved.
4. **Override:** Override is used in the following situation: Suppose a device group is set to auto-approve, but there is one device in that group that should require the manual approval of any response plan within which it appears. In this case, the device should be set to override. Note that if the device was instead set to normal, it would inherit the auto-approve status from the device group to which it belongs.

The status of an approval profile at a given time is determined based on three different factors. These factors are listed based on increasing priority, in that the lowest factor applicable for a given time is the one used to calculate status.

1. **Default:** Each approval profile has a default status that is active when neither of the below factors are set for the current time.
2. **Timespans:** Each approval profile can contain multiple disjoint timespans, which consist of a set of different days of the week, a start time, and an end time. Unless the last factor is applicable, if the current time falls in a timespan, then that timespan's status is used.
3. **Holiday:** Each approval profile has a holiday status, which is active on dates designated as holidays. Note that individual holidays are not distinguished, there is one status common to all the holidays.

3.4.2.3 Notification (NOT-BS)

The Notification Business Service is responsible for creating and resolving notifications and sending these notifications to users via both the UI and email. The NOT-BS exposes three endpoints, for creating and resolving notifications, and for notification retrieval. The NOT-BS will communicate back and forth with the UI via a WebSocket.

To create a notification, a service will send the NOT-BS a notification request with the data to be included in the notification and set of permissions and devices to be used for routing the notification. Similar to how authorization is handled, the NOT-BS will examine these permissions and devices and send the notification to every user who possesses all of them. The NOT-BS will create the notification and return to the requesting service a unique ID. To resolve the notification, the requesting service can send this unique ID, along with additional information to be included in the resolution message, to the NOT-BS resolve endpoint.

The NOT-BS has two representations of notifications in the database. Each notification request creates a base notification, which has information specific to the notification itself. Then, when the NOT-BS determines which users will receive that notification, it generates a user notification which keeps track of information such as snoozes and dismissals.

3.4.2.3.1 Types of Notifications

The NOT-BS supports two different kinds of notifications, alerts and announcements:

An alert requires that some action be taken. Until this action is taken, the alert is considered unresolved. Resolving the alert might require navigating to a different service and doing something there or taking an action out of band. In all cases, a service besides the NOT-BS will sense that the action has been completed and send the notification service a resolve request. The NOT-BS will mark the notification as resolved and propagate this information to the UI.

An announcement does not require any action. It is simply a statement about something that has happened in the system. Therefore, an announcement has no concept of resolution.

Notifications can also be of two different magnitudes of importance, minor and critical:

A notification of critical importance sends an email to all recipients of the notification. It also triggers a pop-up on each user's screen. This pop-up has two options, snooze and dismiss in the case of announcements, and snooze and acknowledge in the case of alerts. Snoozing a critical notification hides the pop-up, starts a timer on the UI and sends a timestamp back to the NOT-BS. When the timer expires, the UI will retrigger the pop-up. Sending the timestamp back to the notification service allows the user to navigate away from the page, and then return, at which time the notification service will send to the UI all timestamps for in-progress snoozes and the UI can resume where it left off. Dismissing an announcement permanently hides the pop-up and removes the announcement from the user's notification feed. Similarly, acknowledging an alert also permanently hides the pop-up, however it does not remove the alert from the notification feed. This removal is only done once the system resolves the alert. Dismissal and acknowledgement of notifications are done on a per-user basis and only affect the UI.

A notification of minor importance does not send an email and does not generate a pop-up. It is only visible via the notification feed. Both minor and critical announcements can also be dismissed via the notification feed.

Here are some examples of all four combinations:

- A response plan requiring approval triggers a critical alert. The notification will contain a redirect link to a page connected to the response plan service that will allow the user to approve or reject a response plan. When the user does so, the response plan service will send a resolve request to the notification service.
- A successful signal optimization result triggers a minor alert. This notification will contain a redirect to a page where this result can be viewed.
- A data source being down triggers a critical announcement. When the data source comes back up again, another critical announcement will be broadcast.
- An event being modified triggers a minor announcement.

3.4.2.4 Response Plan Selection (RPS)

The Response Plan Selection (RPS) Service will be responsible for evaluating the business rules that select response plan sets for modeling and evaluation in response to SunGuide events. This service implements a set of configurable parameterized rules for selecting response plans. The business logic to implement each type of rule is coded into the system using configurable parameters to allow flexible re-use for a variety of different deployment scenarios. The system is designed so that new rule types may be added with minimal coding effort, and changing or applying a new or modified rule of an existing type involves merely changing a deployment configuration. These rules are run through, in order, each time an event occurs. Results from the evaluation of these rules will be stored. The Response Plan Selection Service will use system-wide configuration values to select response plan sets. These values are specified below in the Response Plan Selection Data subsection.

The initial set of identified rules and logic flow is defined below:

Upon being notified of a new SunGuide event:

1. Ensure the event's roadway ID is in a configurable list of roadways. If it is not, end the evaluation.
2. Get the associated R-ICMS GIS segment.
 - a. If the event has an associated ITS IQA link, use that.
 - b. If it doesn't, find the segment using the event's roadway name, direction, latitude, and longitude.
 - c. If this fails, send a notification to all users with the permission to Receive GIS Notifications and end the evaluation.
3. Find the end of the congestion queue.
 - a. Evaluate each upstream segment until a list of contiguous, uncongested segments that is at least a configured length has been found. The end of the queue is the last congested segment before this span of uncongested segments.
 - b. A segment is considered congested if it is traveling below a configured absolute minimum speed or a configured relative percent speed below the historical norm for that segment.
4. Gather all candidate response plans.
 - a. Find all response plans that include all of links that are included in the congestion queue.
5. Filter candidate response plans.
 - a. Remove response plans that have fewer than the configured percent of signals in a state that ICMS can change.
 - b. Remove response plans that have any critical signals that are not in a state that ICMS can change.
 - c. Remove response plans that have a signal controller that does not have the corresponding timing plan stored.
 - d. Remove response plans that have a signal that is already participating in an active response plan.

- e. Remove response plans where the volume/capacity ratio of the diversion route are over a configured threshold based on the event's severity.
 - f. Remove response plans that have fewer than the configured percent of DMS available.
6. Ensure the number of plans plus the do-nothing response fit the configured number of simulation slots.
 - a. Request travel times from Aimsun for all remaining diversion routes.
 - b. Sort the response plans by the total travel time returned
 - c. Take the number of response plans that have the lowest travel time equal to the configured number of simulation slots minus one.
 - d. If the length of the congestion queue is shorter than the previous evaluation, then also include the return-to-normal scenario to evaluate.
7. Send the remaining response plans to the Simulation Engine for simulation.
 - a. After the simulations are completed, the Simulation Engine will send the high level KPIs and score for each of the response plans, the do-nothing plan, and the return to normal plan if included.
8. Limit the number of plans that will be available for selection.
 - a. Sort the evaluated plans in descending order by score and remove all plans that did not score above the benefit threshold for an event on this road segment.
 - b. If there are more plans than the maximum number of plans for an event on this segment, remove all plans after that limit from the list.
9. Send a notification to a user with the Select Response Plan permission indicating that a plan is ready for selection.
 - a. If there are no plans remaining, schedule a re-evaluation to occur and terminate the current suggestion process.
10. A user with the Select Response Plan permission selects a response plan to poll device-owners with Approve Response Plan Devices permission for approval to alter the devices used in the plan or to do nothing at this time.
 - a. If the user selects a plan, send notifications and emails to device-owners with Approve Response Plan Devices permission.
 - b. If the user selects the do-nothing plan, schedule a re-evaluation to occur and terminate the current suggestion process.
11. The users in the corresponding device groups approve or reject the devices they have control over.
 - a. Once all devices are approved, send notifications and emails to the users with the Activate Response Plan permission whom can then activate a response plan.
 - b. The user may choose to deploy the response plan regardless of agency approval.
 - c. The Response Plan Selection Service sends the approved response plan information to SunGuide for implementation.
 - d. The time the next re-evaluation should occur is saved and a timer is started.
12. Re-evaluate the response plans once the re-evaluation time has been met or events are updated in the following ways:
 - a. Location is updated
 - b. Confirmed state changes

- c. Severity changes
- d. Event Active status changes

DRAFT

System Design Document for Regional Integrated Corridor Management System (R-ICMS)

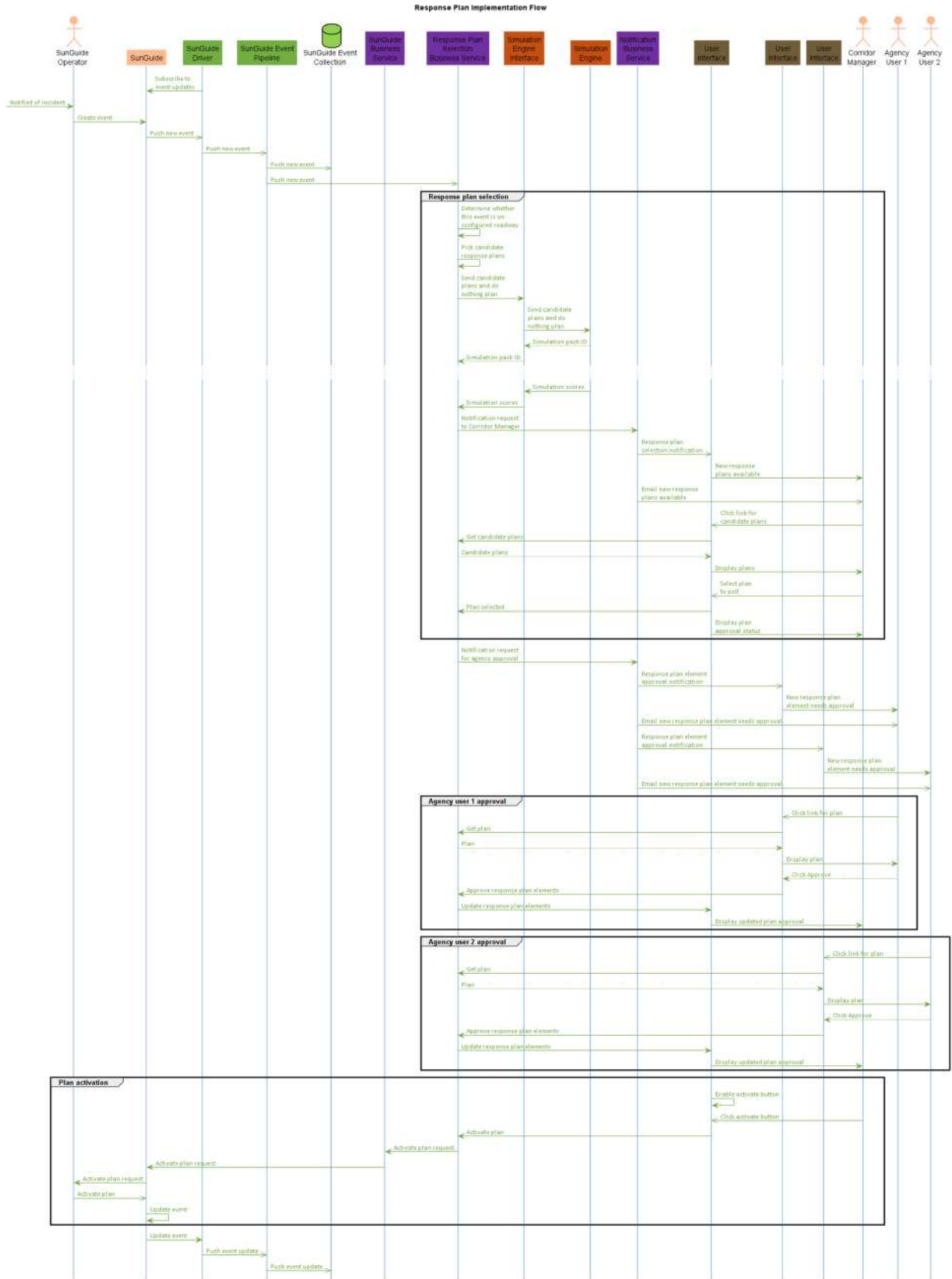


Figure 25 – Response plan selection sequence diagram

3.4.2.4.1 Response Plan Selection Data

The Response Plan Selection Service is responsible for storage and retrieval of system-wide configuration values for selection of candidate response plans. The parameters it should store are:

- The absolute speed threshold for congestion in miles/hour.
- The relative speed threshold for congestion in percent.
- The percent of signals in a controllable state needed to consider a response plan.
- The total number of simultaneous simulation slots available.
- The response plan benefit threshold.
- The retry time in minutes between response plan evaluations when -
 - No response plan exceeded the benefit threshold.
 - A response plan exceeded the benefit threshold but was not activated.
 - A response plan has been activated. This will have a lower bound of fifteen minutes.
- The combined consecutive segment length threshold for determining the end of the congestion queue.
- The rejection threshold percent of volume of capacity for events of each severity level.
- The required percent of available DMS to consider a response plan.
- The maximum number of plans to suggest to the user.

Additionally, the Response Plan Selection Data Service is responsible for storing metrics as rules are being run. These metrics aid in the determination of time that it takes to generate response plans for evaluation by the modelling engine. For each event evaluated, it will contain all fields that are relevant to how far the evaluation proceeded from this list:

- The ID of the triggering event.
- The system that generated the event.
- The timestamp the evaluation of the event was started.
- The ITSQA link of the event.
- The set of selection parameters used.
- The calculated length of the congestion queue.
- The list of each considered response plan with the following:
 - The state of each plan, whether that's considering, rejected, simulating, suggested, polling, approved, or enacted.
 - If the plan is rejected, the reason for the rejection, such as critical signal not available or rejected by the ICM manager.
 - The scores of each simulated plan and the do-nothing plan.
- The timestamp that the plans were sent to the simulation engine for simulation.
- The timestamp that the simulated plans results are returned from the simulation engine.
- The timestamp when the ICM manager opens the simulation results to select a plan for polling.

- The timestamp that a plan was selected for polling.
- The timestamp that a plan was sent out for polling.
- For a plan being polled, each device in the plan will have:
 - The device type.
 - The device ID.
 - The automatic approval profile used.
 - The ID of the user that acted on the device approval status.
 - The time the device was approved or rejected.
 - The approval status.
- The timestamp when the ICM manager activates a response plan.
- The timestamp of when the activated response plan was sent to SunGuide.

3.4.2.5 Simulation Interfaces

The following interfaces define how RICMS and the external simulation tool will interact.

3.4.2.5.1 Simulation Interface

This is a REST API by which the RICMS calls the simulation engine. The API is specified in the Open API 2.0 format by the RICMS team and implemented by the simulation team.

Paths:

- POST /simulation/timing-plans
- GET /volume-capacity-ratio
- GET /travel-time
- POST /simulation/response-plans
- GET /simulation/{packId}
- DELETE /simulation/{packId}
- PUT /simulation/{packId}/priority
- POST /historical-norm

For more detail, see the API specification in the source code repository. An example usage of this interface can be seen in Figure 139. Message classes are shown in the figures below:

System Design Document for Regional Integrated Corridor Management System (R-ICMS)

/simulation/timing-plans

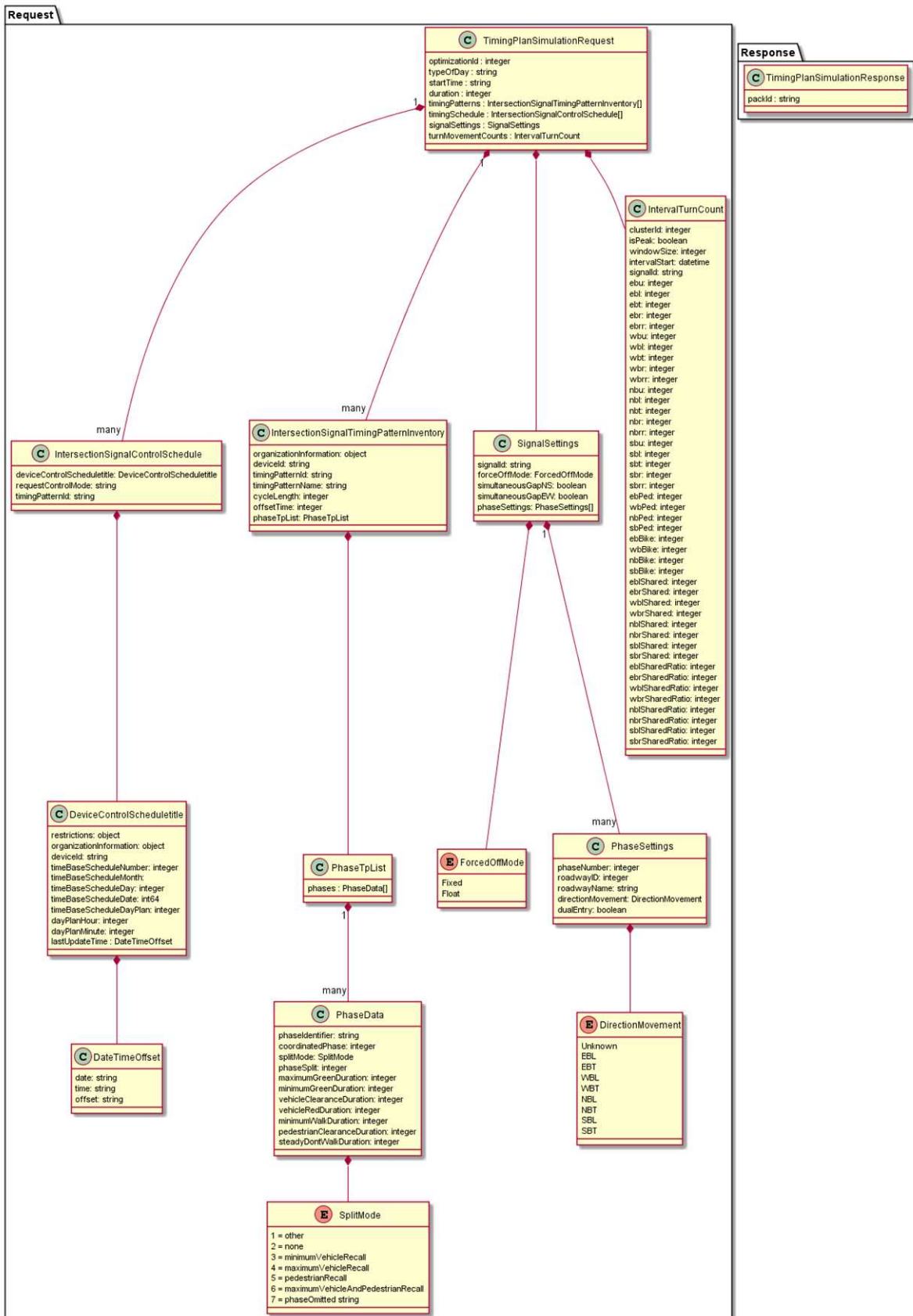


Figure 26 : Simulation Interface – Timing Plans API classes

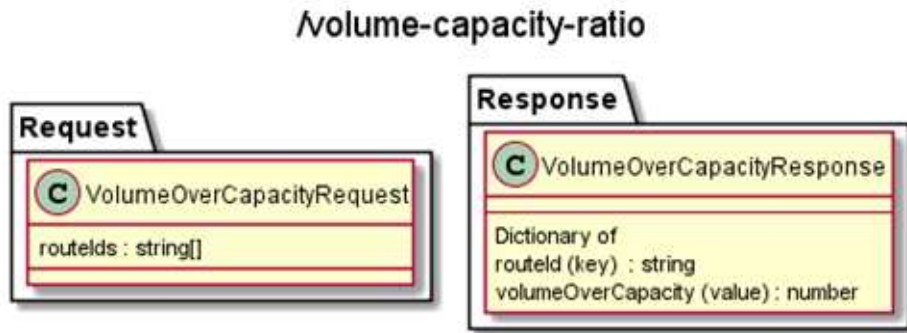


Figure 27 : Simulation Interface – Volume over capacity classes

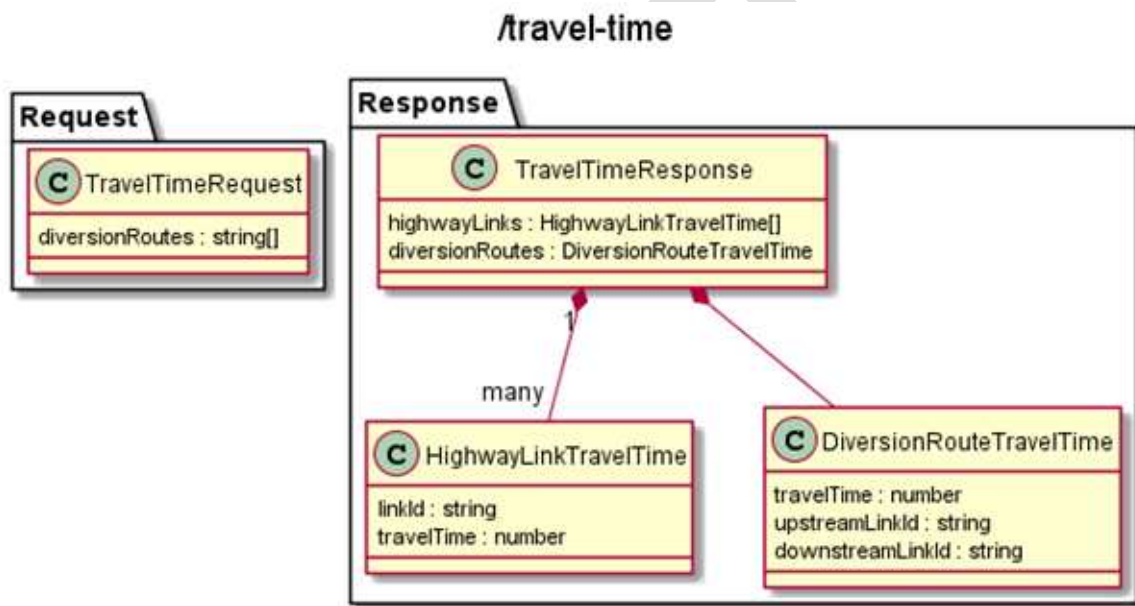


Figure 28 : Simulation Interface – Travel Times classes

/simulation/response-plans

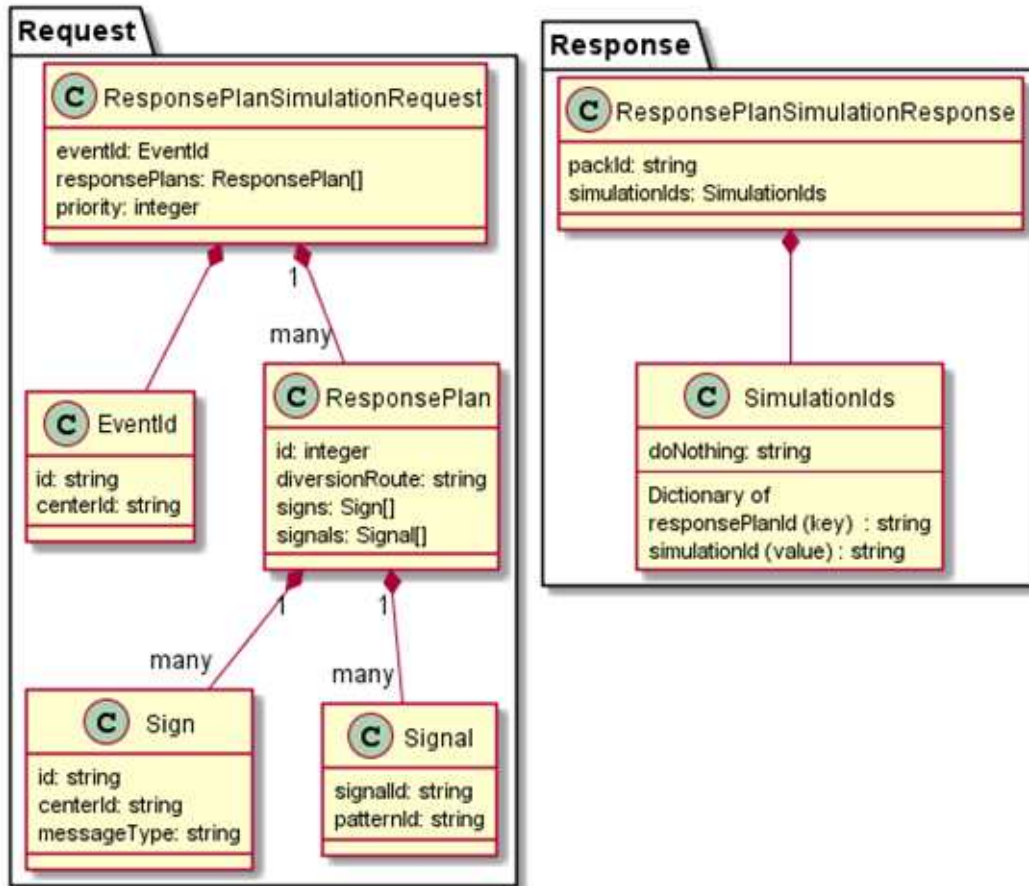


Figure 29 : Simulation Interface – Response Plan Simulation classes

/historical-norm

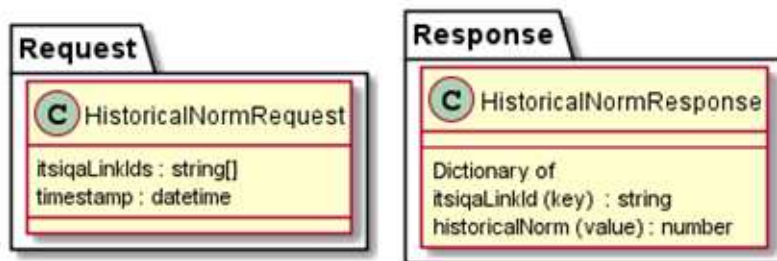


Figure 30 : Simulation Interface – Historical Norm classes

3.4.2.5.2 Simulation Callback Interface

System Design Document for Regional Integrated Corridor Management System (R-ICMS)

This is a REST API by which the simulation engine calls the RICMS to post responses and results. The API is specified in the Open API 2.0 format by the RICMS team and implemented by the RICMS team.

Paths:

- POST /api/sot/timing-plan/scores/{evaluationId}
- POST /api/response-plan/selection/scores/{evaluationId}

For more detail, see the API specification in the source code repository. An example usage of this interface can be seen in seen in Figure 139. Message classes are shown in the figures below:

DRAFT

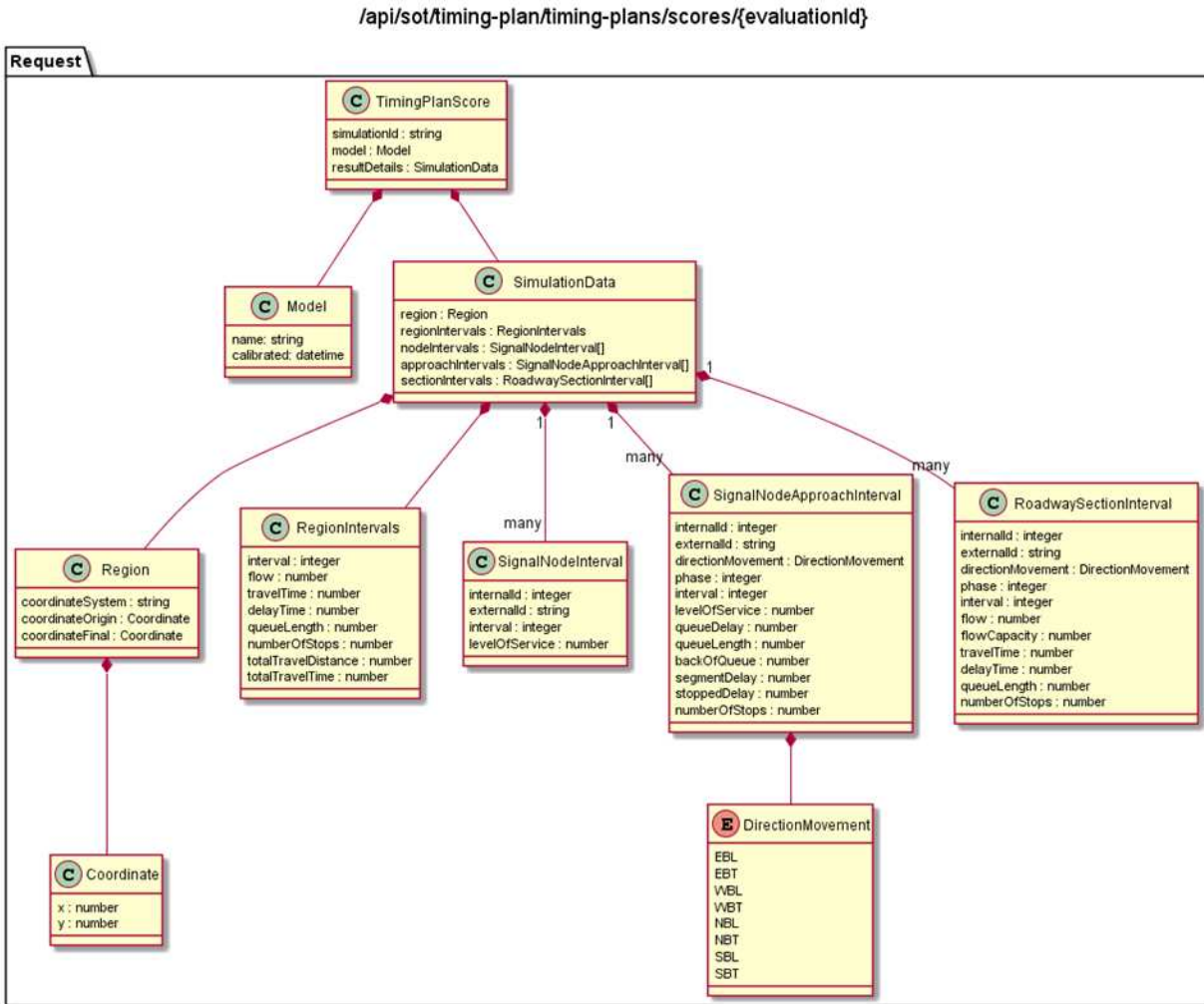


Figure 31 : Simulation Callback Interface – Timing Plan Score classes

/api/response-plan/selection/scores/{evaluationId}

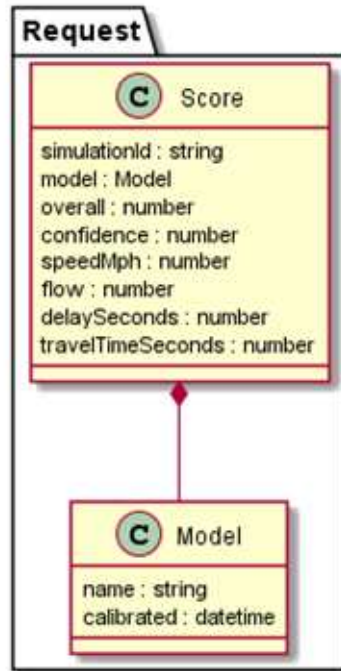


Figure 32 : Simulation Callback Interface – Response Plan Scores

3.4.2.6 Signal Optimization Timing (SOT-BS)

The Signal Optimization Timing Business Service will serve as the back-end process for handling SOT requests. It will host the API to call the HCS7 COTS tool and will connect to the external modeling engine to model signal timing changes. It will be responsible for allowing authorized users to evaluate, modify, and manage signal timing plans. Additionally, it will be responsible for creating and making available the files containing the signal timing plans needed for download into external traffic signal systems.

3.4.2.6.1 Overview and Goals

Running a set of traffic signals in coordination during specified time periods, as a corridor, currently requires extensive manual data collection, analysis, and configuration by traffic engineering professionals. Because of the high degree of resources required, re-evaluation of corridors is typically limited to every few years at best, and it is impossible to keep pace with changing real-world traffic environments and demand.

The SOT-BS component of the R-ICMS will allow flexible and continuous re-evaluation of traffic signal corridors with minimal effort by the combination of leveraging a real-time traffic data ingestion platform, automating a COTS signal timing analysis package, and providing a streamlined user interface.

Following the selection from FDOT-D5 of HCS7 as the COTS signal timing analysis package, analysis was done to identify the minimal set of input data required to perform signal timing optimization using the tool, and real-time data sources for the required information were procured. Real-time data sources used by the SOT are listed below in Table 44, some of which were procured and developed concurrently with the R-ICMS.

Table 44 - SOT real-time data sources

No.	Data Source	Description
1	ITSIQA	Developed by FDOT-D5, ITSIQA provides turn movement count (TMC) data for signalized intersections in 1-minute time intervals. Details can be found in the ITSIQA design documents. Data is ingested continuously into the R-ICMS DFE to build a historical data archive used by the SOT.
2	SunGuide	The SunGuide TCS subsystem was updated to interface with traffic signals using the Traffic Management Data Dictionary (TMDD) protocol to share traffic signal inventory, timing patterns, and control schedules.
3	SIIA	Developed by FDOT-D5, SIIA provides detailed information about intersections, including lane geometry, approach speed limits and grades, and the assignment of signal phases to detectors and movements.

HCS7 is a standalone application developed for Microsoft Windows, which may be used via a Graphical User Interface or a user-driven Command Line Interface (CLI). HCS7 was automated as a SOT back end by containerizing the application and hosting is as a function-as-a-service (FaaS), callable by R-ICMS as an asynchronous web service. The SOT-BS implements a REST Hyper Text Transfer Protocol Secure (HTTPS) service with an API that defines requests to handled. Interaction with the SOT-BS API by internal services and R-ICMS user interfaces trigger the generation of HCS7 CLI processes, allowing end users to run HCS7 through R-ICMS.

The HCS7 product evaluation included the identification and handling of non-standard intersections configurations that have been marked as mission critical by the FDOT traffic engineers. Results of this assessment are shown in Table 45 below.

Table 45 – SOT non-standard intersection conditions

No.	Condition	Resolution
1	Twice per cycle left turn patterns	The HCS7 tool does not implicitly handle ring sequences including overlaps. If this condition is identifiable from source data, the SOT system may double the HCS7 input parameters for left turn minimum and maximum green. Signal engineers must manually divide the resultant HCS7 green time across phases implemented on the controller and ensure the correct ring sequence is used.
2	Protected/permitted left turn with flashing yellow arrow	Included in HCS7 optimizations; SOT system will set input parameters accordingly if condition identifiable from source data. May be identifiable by Signal Inventory Application lane signal head phasing and FYA.
3	Right turn overlap phase with green arrow	Included in HCS7 optimizations; SOT system will set input parameters accordingly if condition identifiable from source data. Unknown whether condition will be identifiable.
4	Side street split phasing	Included in HCS7 optimizations; SOT system will set input parameters accordingly if condition identifiable from source data. May be identifiable by timing patterns where splits for phase 4 match phase 7 and splits for phase 3 match phase 8, and all ring sequences in use run 4 with 7 and 3 with 8.
5	Pedestrian phases concurrent with opposing protected left turn phases	The SOT system will ensure that all minimum signal splits accommodate all pedestrian phases accordingly if condition identifiable from source data. May be identifiable by timing patterns with a phase > 8, which has a pedestrian walk time and overlaps with phase 1 or 5.
6	Exclusive pedestrian movement patterns	Included in HCS7 optimizations; SOT system will set input parameters accordingly if condition identifiable from source data. May be identifiable by timing patterns with non-overlapping pedestrian only phase time.
7	Intersections with 5 or more approaches	Modeling intersections with more than 4 legs in HCS7 is possible but requires non-standard usage. Because these intersections are uncommon, this was agreed to be outside the scope of this project.
8	Two traffic signals controlled by a single controller	This may be modeled by the SOT systems as a single intersection.
9	Movement restrictions per time of day	This data will be retrieved via the Signal Inventory Application and will be viewable within the SOT system. Handling of movement restrictions is described in Table 48 - SOT Intersection Restrictions .
10	Cross coordinated signals (scheduled coordination on major and side streets)	Cross-coordination requires a significant amount of effort and is outside the scope of this project.
11	Coordination on phase 4 or 8 instead of 2 or 6	HCS7 does not implicitly support this. The SOT system will convert phasing data for input to HCS7 and vice-versa for SOT results.

The design process for SOT includes:

- Ongoing review and refinement of the SOT user requirements

- Ongoing review and evaluation of the HCS7 software
- Definition of user interfaces, workflows, and system functionality
- Definition of input data stores and raw data sources
- Definition of operational and output data stores
- Specification of APIs, logic, and control flow for the business service layer
- Specification of APIs, stores, and schemas for data services layer
- Selection of implementation and deployment frameworks and tools

3.4.2.6.2 Operations

Sequence diagrams have been created to illustrate the overall design of the SOT system operations, including user interaction, API calls and control flow of the SOT-BS, and interfaces to data stores via R-ICMS data services layer. The following sequence diagrams show detailed design of the R-ICMS SOT system:

- Figure 133 – SOT Detailed Create Corridor Sequence Diagram
- Figure 134 – SOT Detailed Intersection Features Diagram
- Figure 135 – SOT Detailed Modify Corridor Sequence Diagram
- Figure 136 – SOT Detailed Corridor Optimization 1
- Figure 137 – SOT Detailed Corridor Optimization 2

3.4.2.6.3 Figure 138 – SOT Detailed Deploy Corridor Deployment

The following diagrams show the proposed deployment of physical and virtual components for the R-ICMS, including the SOT system:

- Figure 150 – Service Host Deployment Diagram
- Figure 151 – Containerized Service Orchestration Diagram

3.4.2.6.4 API Functionality

The following sections details functionality to be implemented by the SOT business and data services APIs. Full specification and documentation of the APIs will be available in the source code repository in the OAS format. Although not specified here, full SOT functionality relies on other internal services, such as the signal controller and traffic data services and the alerting and logging business services.

3.4.2.6.4.1 SOT-BS

The SOT-BS performs the following functionality:

- Manage signal corridors
 - Allows users to create, clone, review, approve/deny, deploy, retire, and archive corridor timing plan sets, as well as attached signed and sealed plans sets
 - Allow both periodic recurring and standalone signal timing optimizations
 - Fetch input data from real-time data sources.

- Cluster turn movement counts for a configured date range of historical traffic data to calculate an optimal timing plan schedule.
- Run optimization
 - Dynamically create multi-period HCS7 analysis input by identifying peak, lead-in, and follow-out traffic volumes for configured timing plan schedule and date range of historical traffic data.
 - Call SOT back end to run multiple optimizations asynchronously, one for each time cluster within the corridor activation time.
 - Stores the optimization input and output files
- Run simulation
 - Call the modeling engine (ME) back end to run initial set of traffic simulations for both the new and existing signal timing plans.
 - Calculate performance improvement scores and provides information required to evaluate signal timing plans.
- Manage review requests
 - Handle request for signal owning agencies to approve new signal timing plans, as well as approval and rejection of timing plans per signal
 - Generate reports and data extracts for use by signal engineers to evaluate timing plan sets.
- Manage deployment SOT timing plan sets
 - Handle timing plan set deployments, retirements, and allow management of signed timing plan sets via a file upload/download interface.

3.4.2.6.4.2 SOT-DS

The SOT-DS stores and retrieves data generated by the SOT-BS into an operational data store. A relational database is used for transactional updates to SOT corridor and optimization configurations as they are viewed and edited by multiple users. The data service provides functionality to retrieve, add, and update this data.

3.4.2.6.5 SOT Data stores

3.4.2.6.5.1 SQL-ST

A relational database implemented in MS SQL Server will be used to store highly structured transactional data used and generated within R-ICMS. The data tables, fields, and associated relationships required for SOT operations are available as an Entity-Relationship Diagram (ERD) in Figure 33.

System Design Document for Regional Integrated Corridor Management System (R-ICMS)

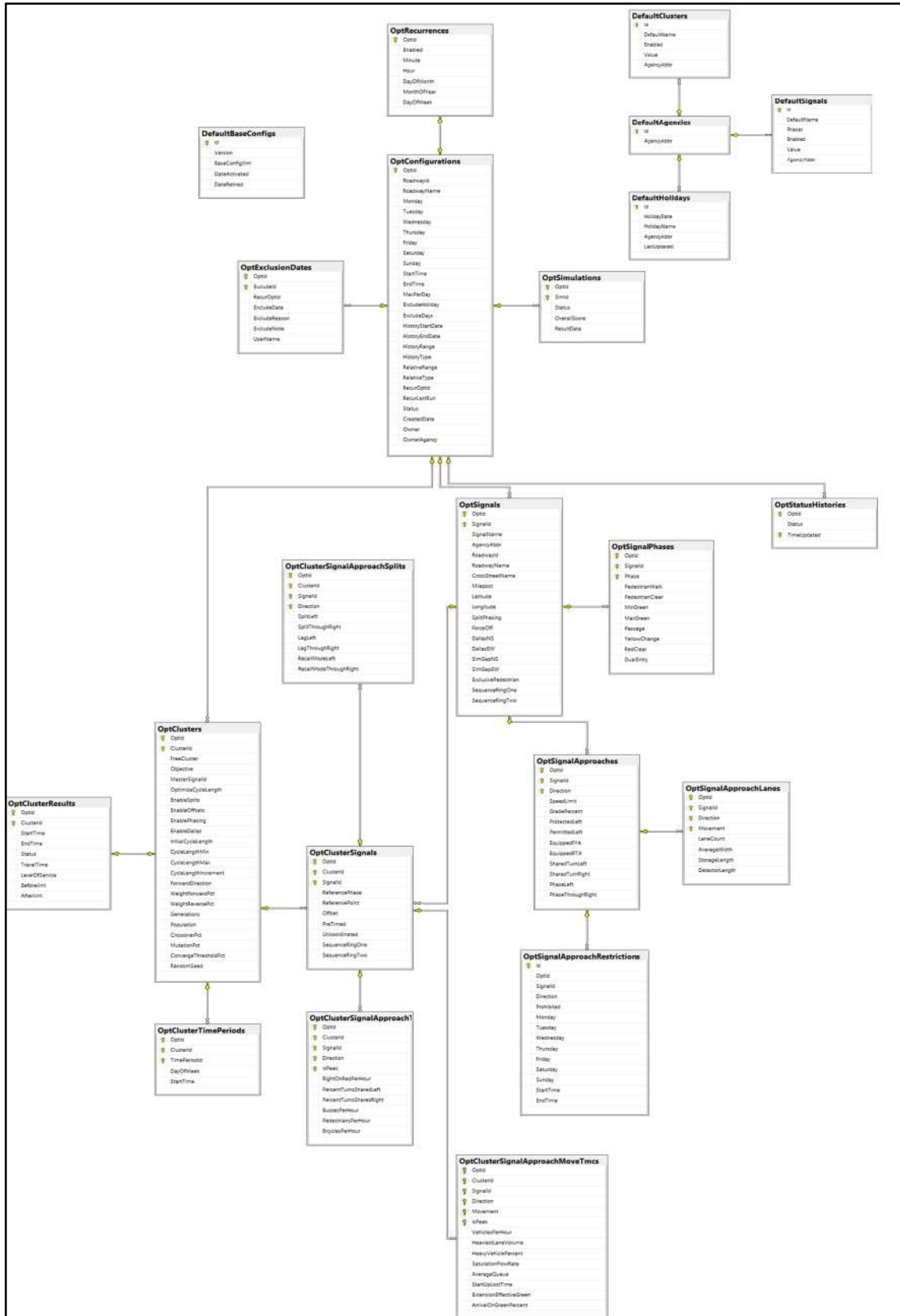


Figure 33 – SOT Operational Database Diagram

3.4.2.6.6 User Permissions

The main SOT permissions are “UseSot”, “ApproveSotDevices”, “AddSotSignatures”, and “AdministerSot”. These permissions, combined with device groups, allow grant access to SOT functionality. Device groups are created and managed in Active Directory, and each device group has an associated role in the R-ICMS system which may be granted to users.

Users with “UseSot” permission may do the following:

- View SOT corridor configurations
 - Users can view signal corridors. Corridors may include devices in groups to which the user has not been assigned a role.
 - User can view signal corridors and recommendations list and map along with the details of corridor configuration and optimization
 - User can download corridor summary report, timing plans, and input data
 - User can view the status of review requests and approvals
 - Users can view all information feed messages pertaining to SOT
- Edit SOT corridor configurations
 - Users can edit signal corridors containing only devices to which they have been assigned a device group access role. Corridors may only include devices in groups to which the user has been assigned a role.
 - User can create and clone signal corridor configurations
 - User can run optimize and simulate corridor timing plan sets
 - User can modify splits and offsets for optimized corridor timing plans sets
 - User can request or cancel agency review of corridor timing plan sets
 - User can comment on an optimized corridor timing plan set
 - User can mark corridor configuration as deployed or retired
 - User receives alerts on completion of signal optimization and simulation

Users with “ApproveSotDevices” permission may do the following:

- Approve/deny signal timing plans
 - Users can approve/deny timing plans for signals within device groups to which the user has assigned roles.
 - User receives alerts when timing plan reviews are requested, canceled, or resolved

Users with “AddSotSignatures” permission may do the following:

- Upload sets of signed and/or sealed timing plans

Users with “AdministerSot” permission may do all above for all corridors, created by any users or system service, in any status deployed or otherwise, and using any set of signals.

Devices may only be assigned to one group. The following device groups are recommended:

- Device groups for border signals, to which access by two agencies is desired

- Device groups per signal owner agency

3.4.2.6.7 Signal Timing Files

The R-ICMS will support the creation and retrieval of signal timing plan files for eventual import into signal ATMS software and traffic signal controllers. Files will be retrievable via the R-ICMS SOT UI. Supported formats are:

- Timing plan report – downloadable as a pdf file, this includes information about the signal corridor along with the signal timing plans in a format like timing sheets exports from signal controllers; the full list of information to be included and full format of this report is to-be-determined.
- Data files – a set of three csv formatted data files for in in both Synchro and TruTraffic, which includes phasing, timing plans, and turn movement counts.

The phasing data file contains signal phasing data for one or more intersections. The example below shows the format and sample data for signal 1001:

```
RECORDNAME,INTID,D1,D2,D3,D4,D5,D6,D7,D8  
BRP,1001,112,111,211,212,121,122,221,222  
MinGreen,1001,5,15,5,7,5,15,5,7  
MaxGreen,1001,10,92.4,31.1,54.4,11.9,91.5,42.3,43.6  
VehExt,1001,3,4,3,3,3,4,3,3  
TimeBeforeReduce,1001,0,0,0,0,0,0,0,0  
TimeToReduce,1001,0,0,0,0,0,0,0,0  
MinGap,1001,3,4,3,3,3,4,3,3  
Yellow,1001,5,4.1,5,4.2,4.1,5,4.2,5  
AllRed,1001,3,3.5,3.9,3.4,2,3.5,3.5,3.4  
Recall,1001,0,3,0,0,0,3,0,0  
Walk,1001,,7,,,,7,,  
DontWalk,1001,,26,,,,28,,  
PedCalls,1001,,0,,,,0,,  
MinSplit,1001,13,40.6,13.9,39.6,11.1,43.5,12.7,46.4  
DualEntry,1001,0,0,0,1,0,0,0,1
```

Phasing data that is not available from the SOT or current data sources will be set to defaults:

- MinGreen, set to SOT/TMDD min green
- MaxGreen, set to TMDD max green
- VehExt, set to SOT passage time
- TimeBeforeReduce, defaulted to 0
- TimeToReduce, defaulted to 0
- MinGap, set to SOT passage time
- Yellow, set to SOT/TMDD yellow time

- AllRed, set to SOT/TMDD red time
- Walk, set to SOT/TMDD pedestrian walk time
- DontWalk, set to SOT/TMDD pedestrian clearance time
- PedCalls, defaulted to 0 for phase 2 and 6, otherwise omitted
- MinSplit, set to SOT minimum split per phase, calculated as the maximum of:
 - Sum of yellow, red, and minimum green times
 - Sum of yellow, red, walk, and flash-don't-walk times
- DualEntry, set to SOT/TMDD min green

The signal timing data file contains timing plans for one or more intersections, including multiple plans per intersection. The example below shows a plan for time cluster (PLANID) 2 for signal 1001. will have the following format:

```
PLANID,INTID,S1,S2,S3,S4,S5,S6,S7,S8,CL,OFF,LD,REF,CLR
2,1001,18,100,40,62,18,100,50,52,220,187,2357,26-,7.6
```

The turn count data contains vehicles-per-hour turn counts for one or more intersections for a time period starting at the indicated date and time. The example below shows data for signal 1001 starting at 6:30 am:

```
Turning Movement Count,,,,,,,,,,,,,
60 Minute Counts,,,,,,,,,,,,,
DATE,TIME,INTID,NBU,NBL,NBT,NBR,SBU,SBL,SBT,SBR,EBU,EBL,EBT,EBR,WBU,WBL,WBT
,WBR
5/4/2016,630,1002,0,71,991,36,,136,1389,70,0,170,257,68,0,47,140,72
```

3.4.2.7 SunGuide (SG-BS)

A REST API (Business Service) will be provided to allow the User Interface (UI) to communicate with SunGuide. This REST API will allow addition of comments to an Event and will be the method of communication used to send response plans to SunGuide. When the Business Service (SG-BS), receives a request, it will in turn make a request to the SunGuide API (over TCP Socket). The response returned by SunGuide will be sent back as a response to the REST API request. If the SG-BS is not able to communicate with SunGuide, then the SG-BS will send a corresponding message indicating the communication error.

3.4.3 User Interface

The user interface will be the primary interface to the DFE and the DSS for R-ICMS users. It will be comprised of the necessary components for display of the data that the Data Services layer

provides. It will communicate with services in the business layer for appropriate transactional requests.

The following User Interface components have been identified:

- Development languages for the User Interface (UI): Angular JS, HTML5, CSS, Typescript and bootstrap
- UI implementation: Model View Controller (MVC) design pattern
- Folder structure of the application can be found in the coding standards document
- Map page and streaming services: ArcGIS API for JavaScript version [3.25](#) will be used.

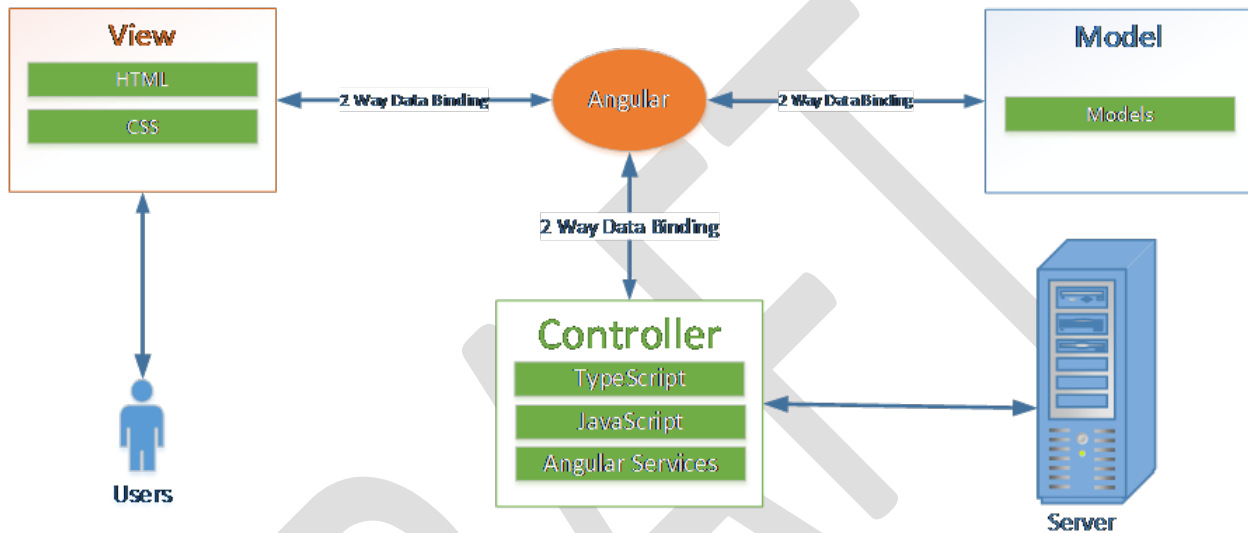


Figure 34 – Angular MVC

The overall navigation hierarchy of the application is illustrated in Figure 35 – User Interface Navigation Hierarchy.

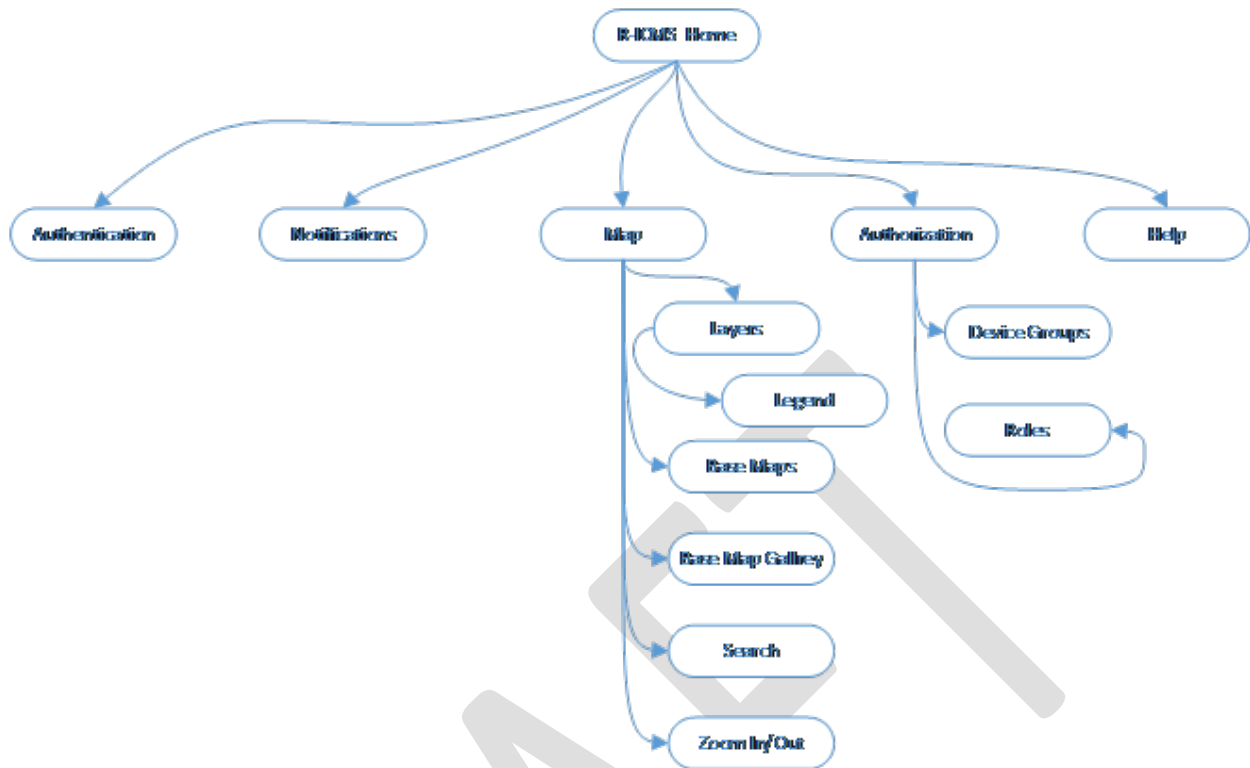


Figure 35 – User Interface Navigation Hierarchy

3.4.3.1 Authentication (Authn-UI)

Upon navigating to the base R-ICMS website, the user will be redirected to the login screen. The user can log into the R-ICMS Application using the login screen shown in Figure 36 – Login Screen. After the user logs into the R-ICMS, the user will be redirected to the designated landing page, which for most users will be the map page.

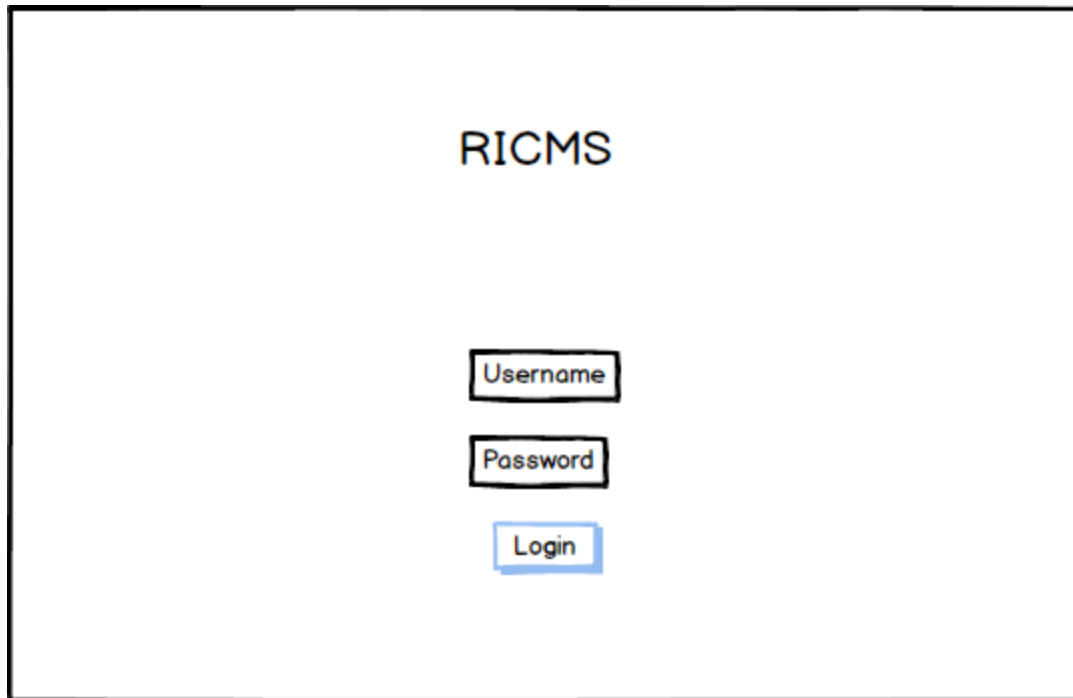


Figure 36 – Login Screen

3.4.3.2 Map (MAP-UI)

The map component will be one of the primary entry points into the UI. It will connect to and process data from the GeoEvent service and the ArcGIS service. It will display data to the user based on permissions and current filters. Additional global filters will be selectable which will apply to all data shown on the map (as applicable to the filter). Users will be able to pan and zoom per common web map interfaces. Two types of data will be shown on the map. Data associated with pre-defined locations, such as speed of a specific link (and therefore its color) will be linked in the UI itself with that portion of the map based on an ID of that portion of the map (e.g., the pixels making up the link). Data not associated with predefined locations on the map (such as AVI data from a transit vehicle) will need to be drawn on top of the map by the UI. These two types of data will be handled by different data services but will be drawn by the UI onto the same map.

- The user will be able to Search on the map
- The user will be able to zoom in or zoom out the map and pan the map
- The user will be able to view the basemap, static data and dynamic data on the map by toggling the respective layers
- User will be able to view static data such as:
 - Schools
 - Emergency Responder Locations
 - Transit
- The user will be able to view dynamic data such as

- Traffic conditions
 - The user will be able to view a map legend for a selected layer

The following diagram depicts the dataflow for static and dynamic data from the data store to map page:

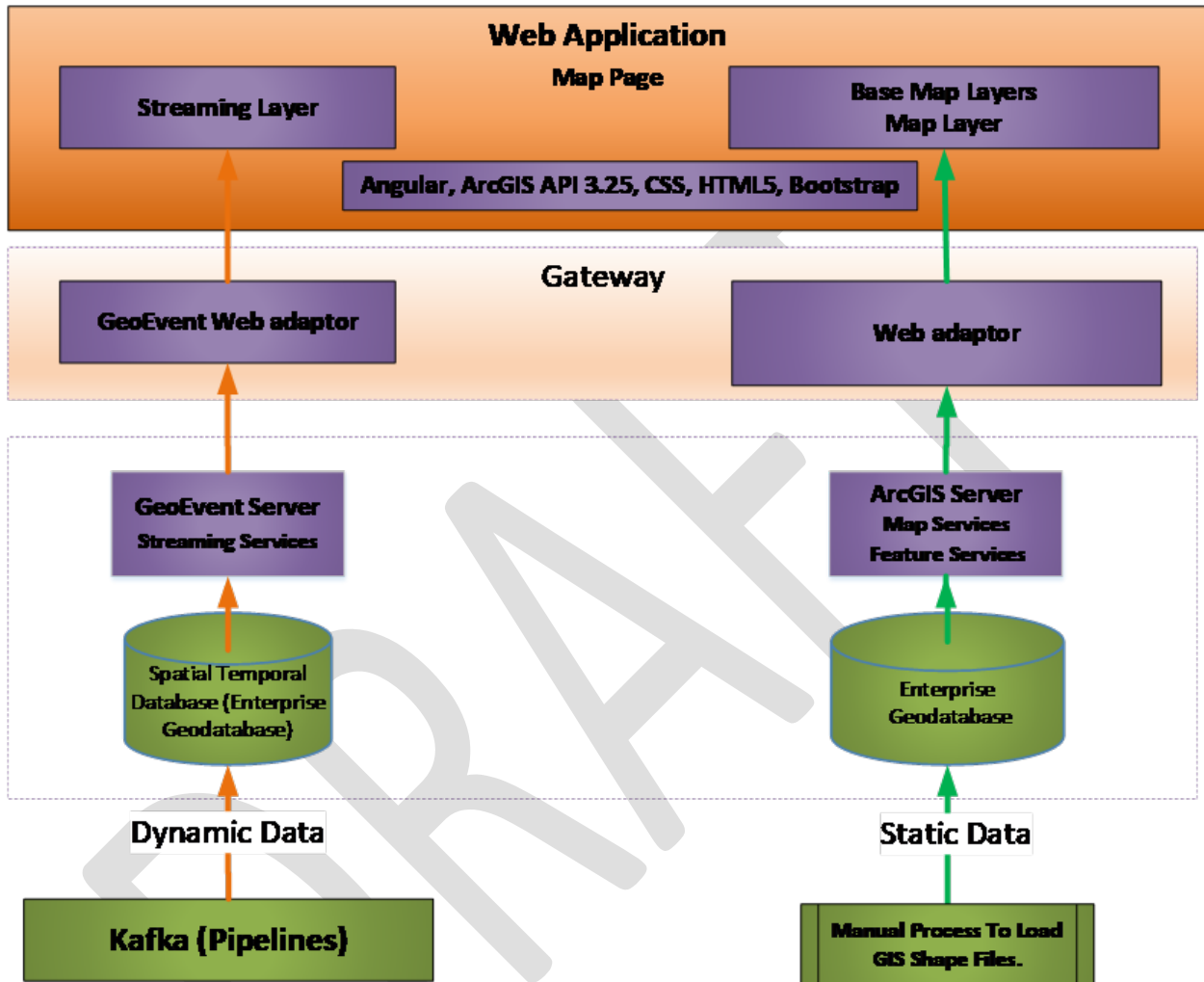


Figure 37 – Map Data Flow

The following screenshots show examples of the screens that users will see when viewing the map portion of the R-ICMS.

3.4.3.2.1 Main Screen

Figure 38 – Main Screen depicts the initial screen the user sees after logging on.

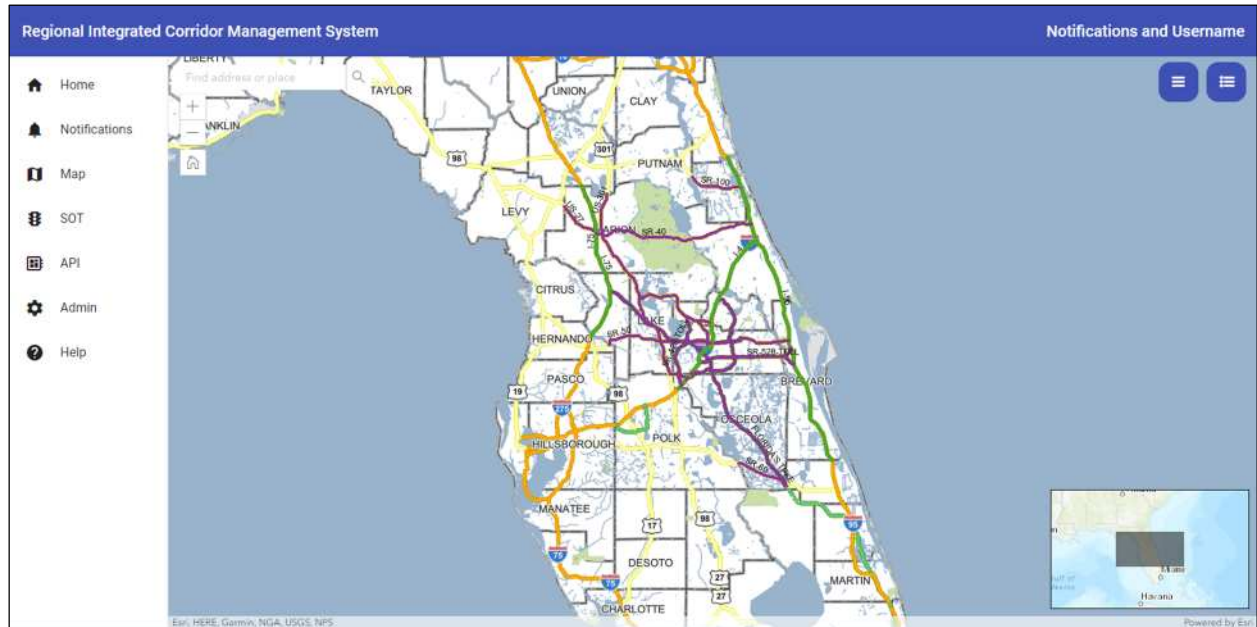


Figure 38 – Main Screen

3.4.3.2.2 Search

Users will be able to search by address, street name, zip code and road name. The map will be zoomed in/out to an extent to show all the search results. Figure 39 – Search Screen depicts the screen used to search.

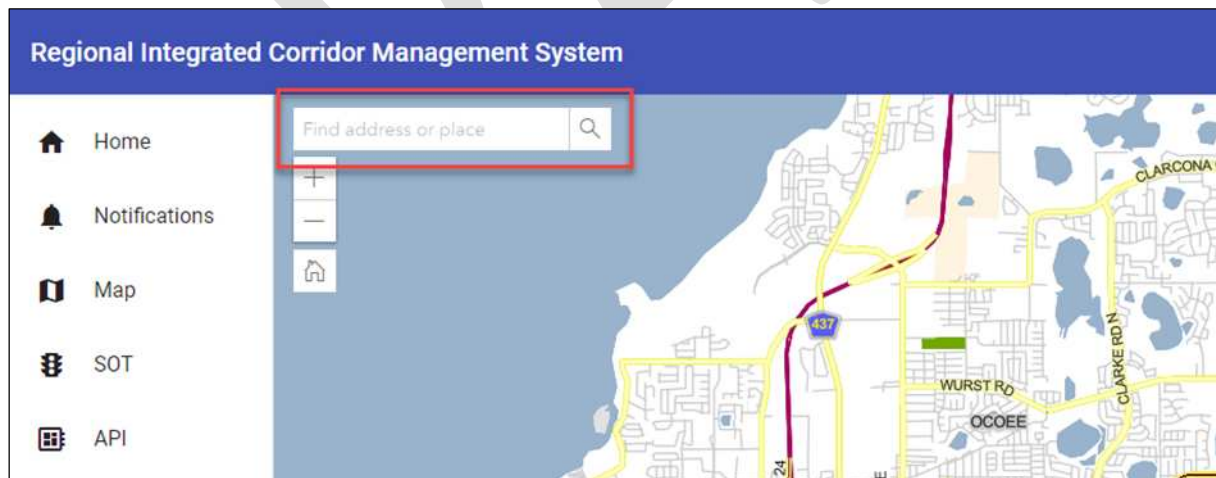


Figure 39 – Search Screen

3.4.3.2.3 Zoom In/Out, Pan

Users will have the ability to Zoom In/Out on the map page by selecting “+” and “-” icons. Also, users will have the ability to pan the map to a specific location. Figure 40 – Zoom features highlights the zoom features.

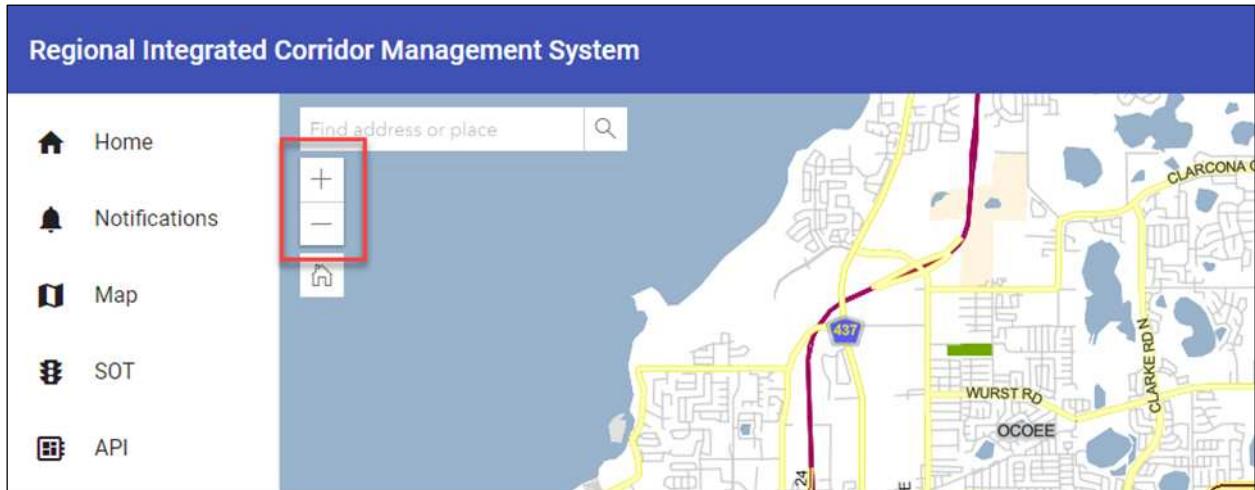


Figure 40 – Zoom features

3.4.3.2.4 Home

Users will have the ability to go back to the default extent when the Home button is selected. Figure 41 – Home button highlights the icon used to return to the home screen.

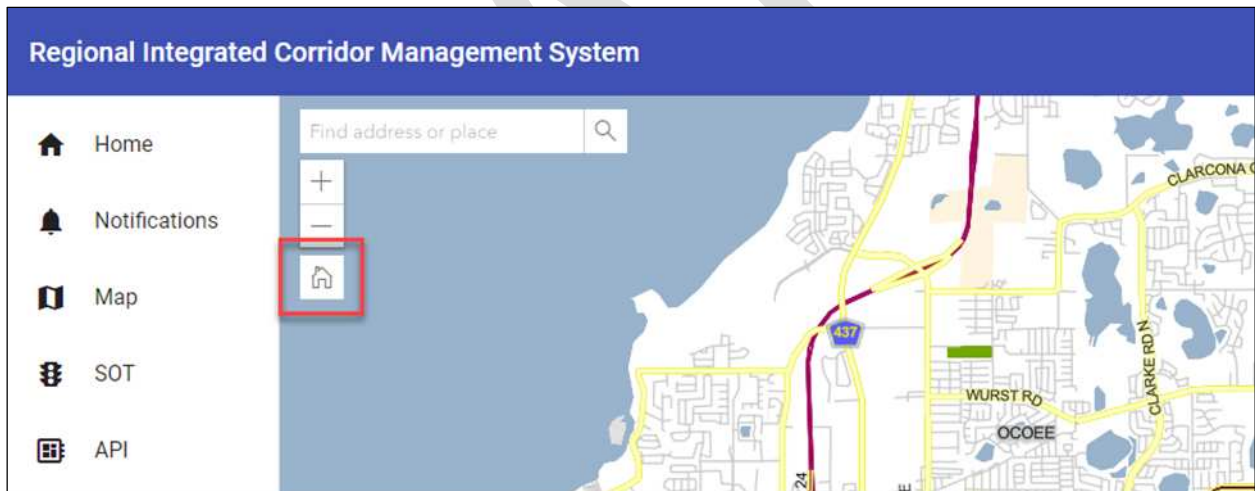


Figure 41 – Home button

3.4.3.2.5 Information Window

Details of a feature will be shown when the user selects a feature. Figure 42 – Feature Details illustrates the system showing details when a feature is selected.

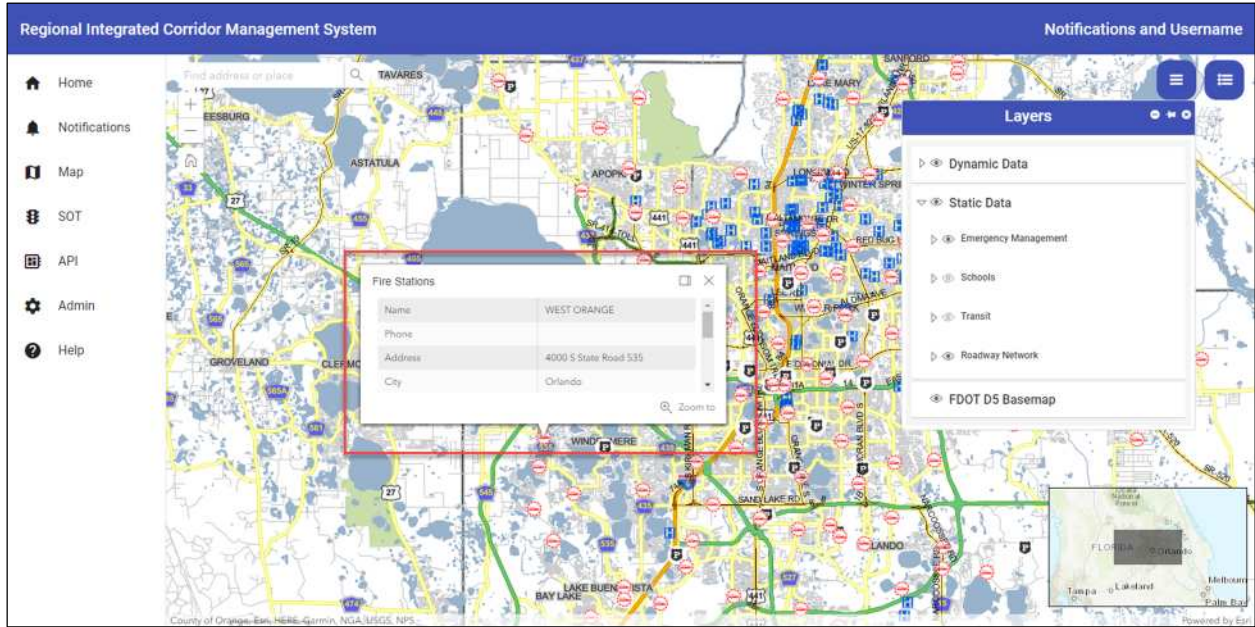


Figure 42 – Feature Details

3.4.3.2.6 Layers (Table of Contents)

3.4.3.2.6.1 Static Data

Users will be able to select static data layers. Based on the selected layers, the features will be displayed on the map. Below is a list of the layers that will be available for users to select.

- Schools
- Emergency Responder Locations
- Transit

Figure 43 – Layer Selection Screen illustrates the selection of layers.

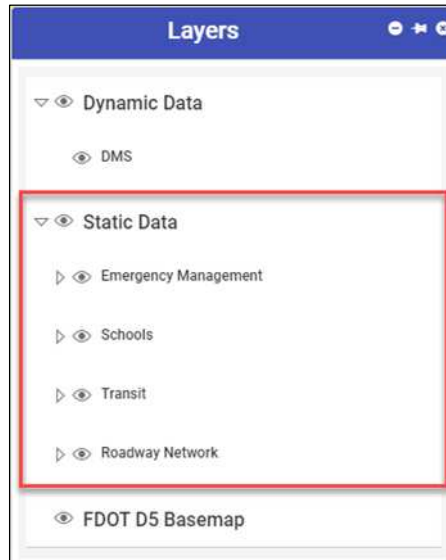


Figure 43 – Layer Selection Screen

3.4.3.2.6.2 Dynamic Data

Users will be able to select the dynamic data layer. When the data layer is selected, the map will display the appropriate data from GeoEvent Service. Figure 44 – Traffic Condition Layer Selection illustrates the selection of the Traffic Conditions layer.

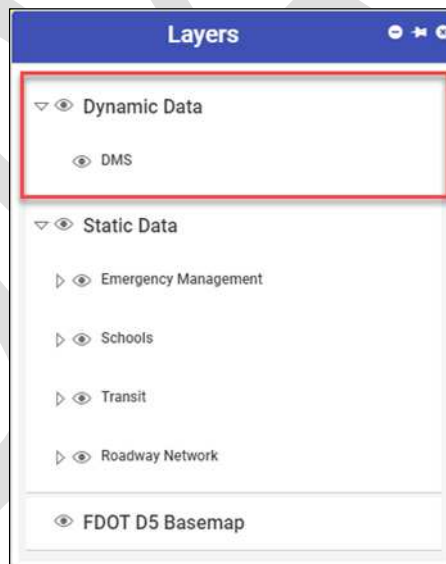


Figure 44 – Traffic Condition Layer Selection

3.4.3.2.6.3 Basemap

Users will be able to choose the basemaps from the basemap list as shown in Figure 45 – Basemap Selection.

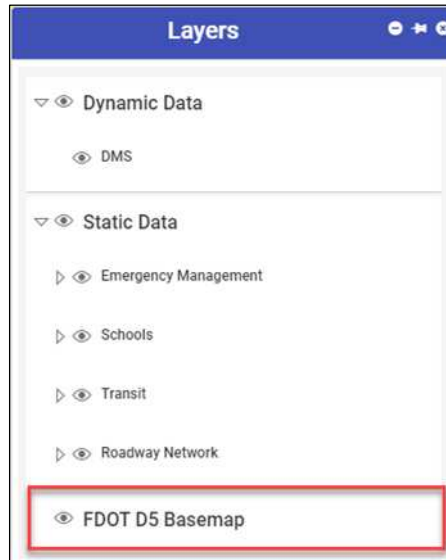


Figure 45 – Basemap Selection

3.4.3.2.6.4 Legend

Users will be able to view the legend for a selected layer, as shown in Figure 46 – Legend Selection.

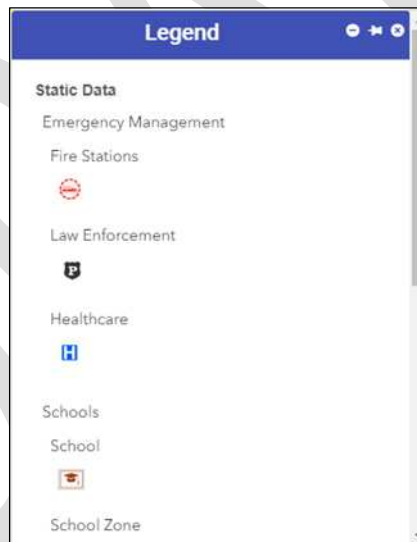


Figure 46 – Legend Selection

3.4.3.2.6.5 Navigation

Figure 47 – Navigation shows the navigation between screens.

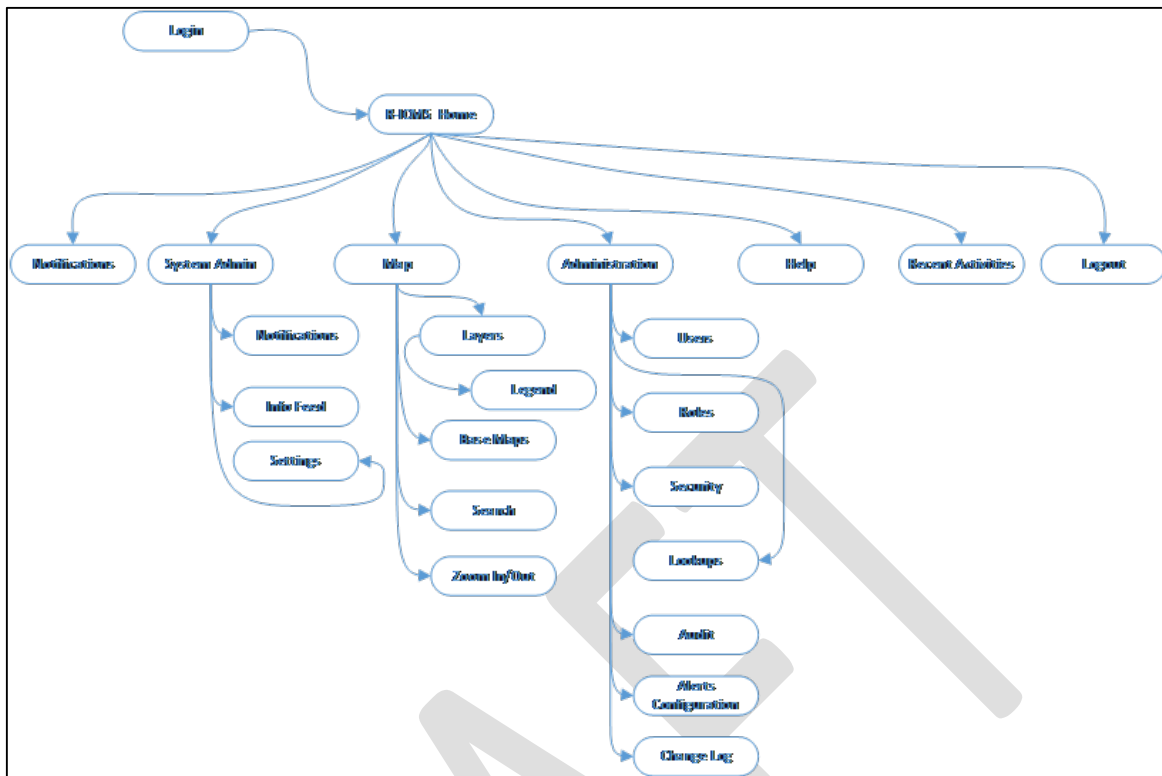


Figure 47 – Navigation

3.4.3.2.7 Weather

Weather data (watches, warnings, and forecasts) are accessible through streaming services from NOAA & NWS (<https://idpgis.ncep.noaa.gov/arcgis/rest/services>). The available services provide a wide range of weather data layers that can be leveraged within the R-ICMS. These services are managed and served via data contributors and managers – this provides a direct access convenience; however, use is limited to visibility and currently published metadata. Therefore, a user will be able to access the real-time data stream as provided by NOAA and NWS and be able to view pertinent feature information via info windows. However, further access and manipulation is restricted.

The following data will be made available through the Map UI as selectable layers:

- Radar: Total Storm Precipitation

The Radar layer will allow the user to toggle on/off the NOAA Radar Service as a layer and effectively shows the total precipitation for the currently forecasted storm.

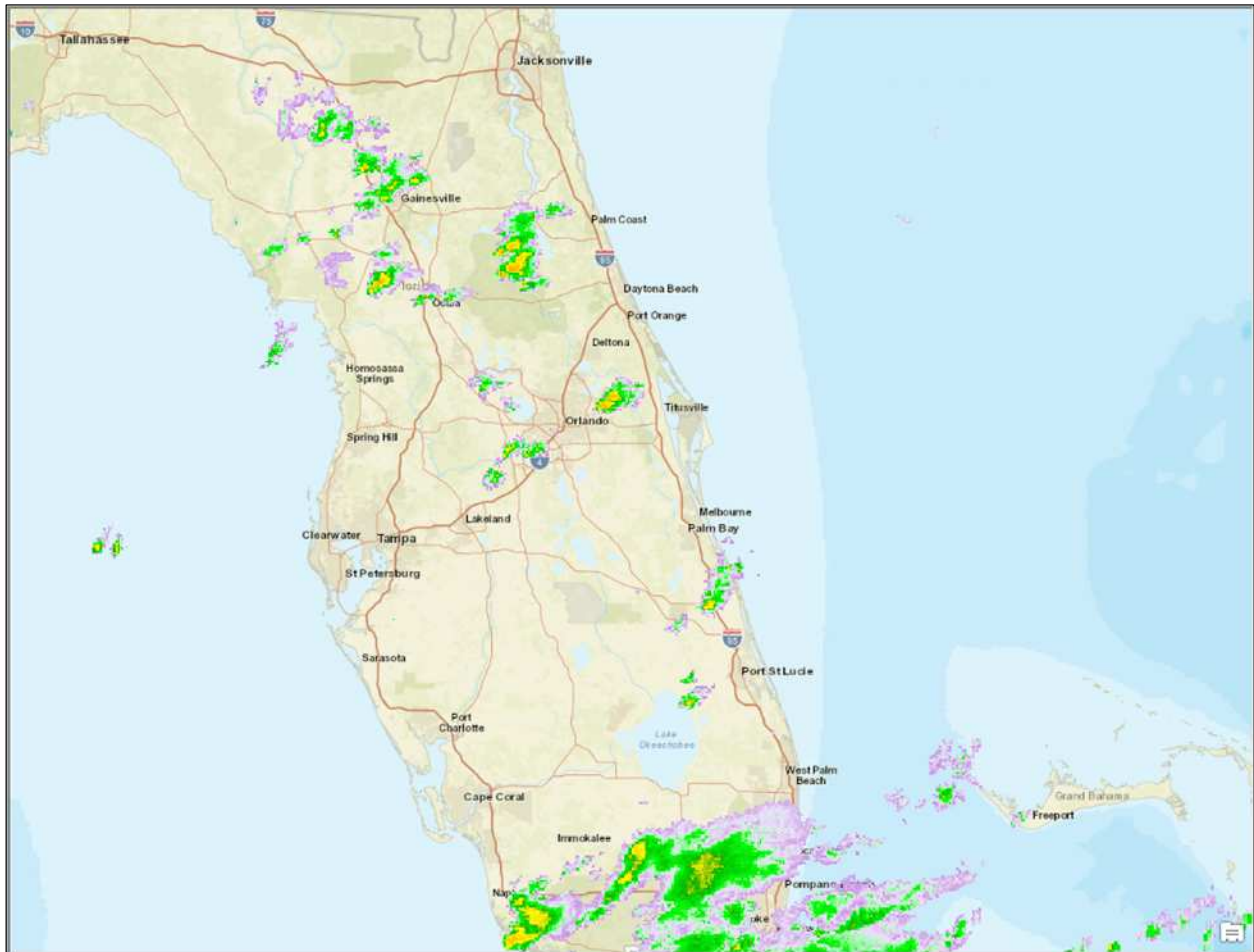


Figure 48 – Total Storm Precipitation

- Storm Watches & Storm Warnings

The Storm Watches & Warnings layer will allow the user to toggle on/off the NOAA Watches & Warnings Service as a layer. This layer shows the current extent and severity of the storm and the potential affects.

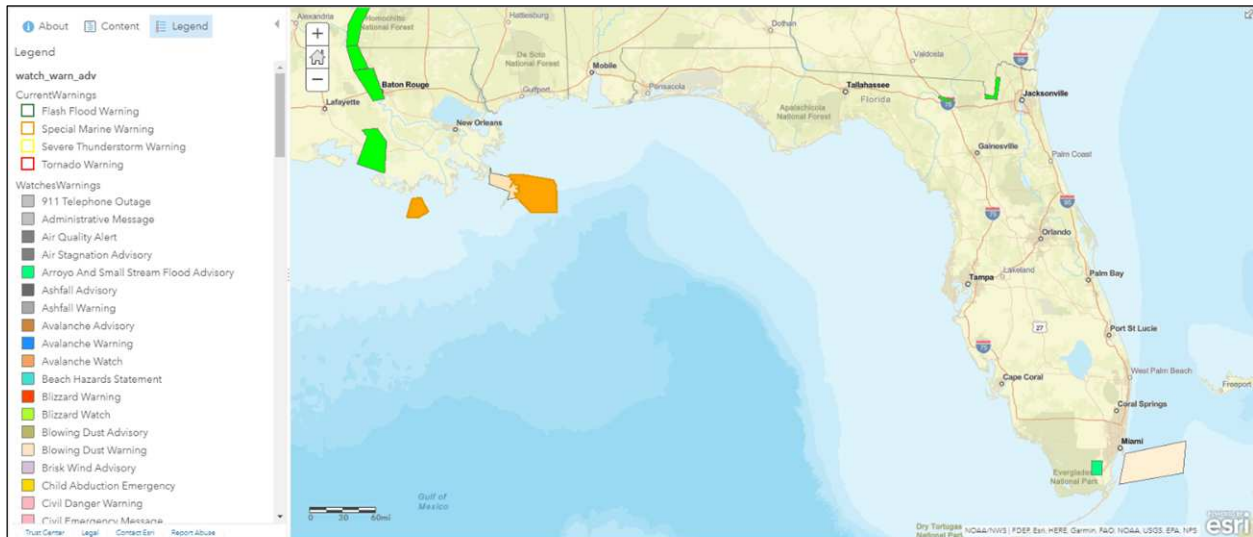


Figure 49 – Storm Watches & Warnings

- Tropical Cyclone Watches & Warnings
The Tropical Cyclone Watches & Warnings layer will allow the user to toggle on/off the NOAA Tropical Cyclone Watches & Warnings Service as a layer. This layer shows the currently forecasted track, intensity, cone of uncertainty, and wind speed and area of affect.

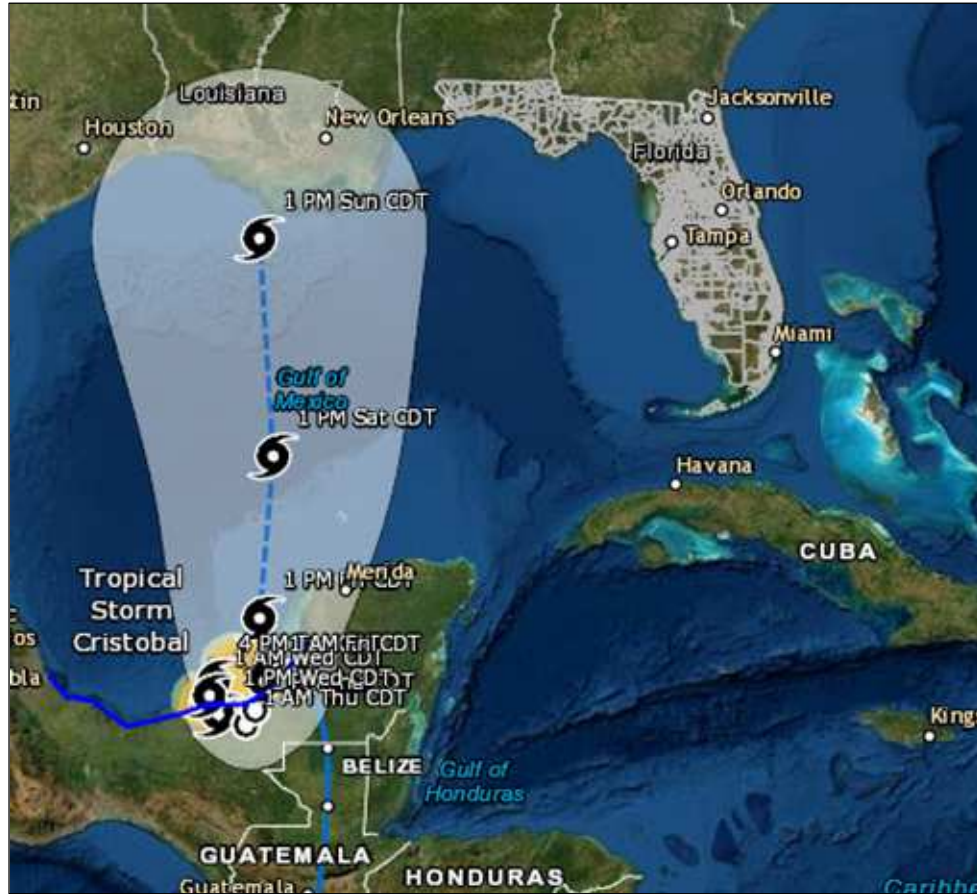


Figure 50 – Tropical Cyclone Watches & Warnings

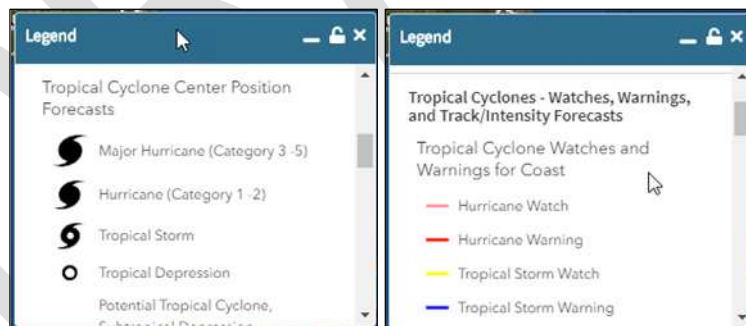


Figure 51 – Tropical Cyclone Watches & Warnings

- Other Warnings & Alerts
 - A full list of available watches and warnings are available from the NOAA Watches & Warnings Service:
 - (https://idpgis.ncep.noaa.gov/arcgis/rest/services/NWS_Forecasts_Guidance_Warnings/watch_warn_adv/MapServer/1)

3.4.3.3 Event List Interface

The event list will provide users with a searchable, filterable, and sortable interface to assist in viewing events, event details, creating new RICMS events, viewing an event on the map, emailing event information and creating an event list report.

3.4.3.3.1 Event List

The RICMS system will provide users a way to access the event list from the main menu and from the map page.

The screenshot shows the R-ICMS main menu with the 'Event List' option selected. The table below represents the data shown in the interface:

ID	Status	Event Type	Severity	Source	Location	Start Date/Time	End Date/Time	Owner	Organization	Code	Description
<input checked="" type="checkbox"/> 907361	Active	Construction	Minor	SunGuide	SR-417	12/24/2019 - 9:03	1/15/2020 - 9:25	KRMETLC	D5 RTMC	15	Scheduled Road Work
<input type="checkbox"/> 891381	Active	Construction	Severe	SunGuide	SR-417	12/23/2019 - 9:19	1/14/2020 - 9:30	KRMETLC	D5 RTMC	14	Scheduled Road Work
<input type="checkbox"/> 781584	Active	Congestion	Minor	SunGuide	SR-91	12/20/2019 - 9:36	1/13/2020 - 10:05	KRMETLC	D5 RTMC	13	Overpass construction
<input type="checkbox"/> 788302	Active	Disabled Vehicle	Minor	SunGuide	SR-436	12/17/2019 - 10:12	1/10/2020 - 10:35	KRMETLC	D5 RTMC	4	Westbound I-4 Closed
<input type="checkbox"/> 774285	Active	Incident	Moderate	RICMS	SR-414	12/4/2019 - 10:13	1/9/2020 - 10:50	Unassigned	FDOT District 5		Westbound I-4 Closed
<input type="checkbox"/> 771658	Active	Special Event	Minor	RICMS	SR-417	11/27/2019 - 10:22	1/8/2020 - 11:20	Unassigned	FDOT District 5		Presidential Motorcade
<input type="checkbox"/> 743027	Active	Construction	Minor	SunGuide	SR-50	11/26/2019 - 10:40	1/7/2020 - 11:35	KRMETLC	D5 RTMC	2	Scheduled Road Work
<input type="checkbox"/> 703241	Active	Incident	Minor	SunGuide	SR-91	11/12/2019 - 10:53	1/6/2020 - 11:55	KRMETLC	D5 RTMC	5	New on-ramp
<input type="checkbox"/> 774270	Active	Incident	Severe	SunGuide	SR-436	11/1/2019 - 11:46	1/3/2020 - 12:40	KRMETLC	D5 RTMC	6	Overpass construction
<input type="checkbox"/> 769729	Active	Congestion	Minor	SunGuide	SR-414	10/30/2019 - 12:37	1/2/2020 - 12:45	KRMETLC	D5 RTMC	13	Overpass construction
<input type="checkbox"/> 744040	Active	Disabled Vehicle	Minor	RICMS	SR-417	10/23/2019 - 12:45	1/1/2020 - 13:05	Unassigned	FDOT District 5		Westbound I-4 Closed
<input type="checkbox"/> 785435	Active	Construction	Moderate	RICMS	SR-417	10/21/2019 - 12:58	12/31/2019 - 13:15	Unassigned	FDOT District 5		Scheduled Road Work
<input type="checkbox"/> 812197	Active	Special Event	Important	RICMS	SR-91	10/17/2019 - 13:06	12/30/2019 - 13:25	KRMETLC	FDOT District 5		Overpass construction
<input type="checkbox"/> 740948	Active	Construction	Moderate	RICMS	SR-436	10/16/2019 - 13:07	12/27/2019 - 13:30	KRMETLC	FDOT District 5		Scheduled Road Work
<input type="checkbox"/> 883854	Active	Special Event	Moderate	SunGuide	SR-414	10/9/2019 - 13:15	12/26/2019 - 13:45	KRMETLC	D5 RTMC	3	Westbound I-4 Closed
<input type="checkbox"/> 744040	Active	Incident	Minor	SunGuide	SR-414	9/10/2019 - 13:24	12/20/2019 - 14:45	KRMETLC	D5 RTMC	2	Overpass construction
<input type="checkbox"/> 785435	Active	Congestion	Moderate	SunGuide	SR-417	9/3/2019 - 13:40	12/18/2019 - 14:50	KRMETLC	D5 RTMC	5	Overpass construction
<input type="checkbox"/> 812197	Active	Disabled Vehicle	Important	RICMS	SR-417	9/2/2019 - 13:41	12/17/2019 - 15:05	Unassigned	D5 RTMC	6	Westbound I-4 Closed
<input type="checkbox"/> 740948	Active	Construction	Moderate	RICMS	SR-91	8/28/2019 - 13:47	12/15/2019 - 15:10	Unassigned	D5 RTMC	13	Scheduled Road Work
<input type="checkbox"/> 883854	Active	Special Event	Moderate	RICMS	SR-436	8/27/2019 - 14:26	12/14/2019 - 15:50	KRMETLC	FDOT District 5		Overpass construction
<input type="checkbox"/> 744040	Active	Construction	Minor	SunGuide	SR-414	8/23/2019 - 14:32	12/13/2019 - 15:55	KRMETLC	FDOT District 5	2	Scheduled Road Work
<input type="checkbox"/> 785435	Active	Special Event	Moderate	SunGuide	SR-414	8/20/2019 - 14:20	12/10/2019 - 16:00	KRMETLC	FDOT District 5	5	Westbound I-4 Closed
<input type="checkbox"/> 812197	Active	Incident	Important	RICMS	SR-417	8/19/2019 - 14:15	12/9/2019 - 16:05	KRMETLC	FDOT District 5	6	Overpass construction

Figure 52 – Main Menu Event List

System Design Document for Regional Integrated Corridor Management System (R-ICMS)

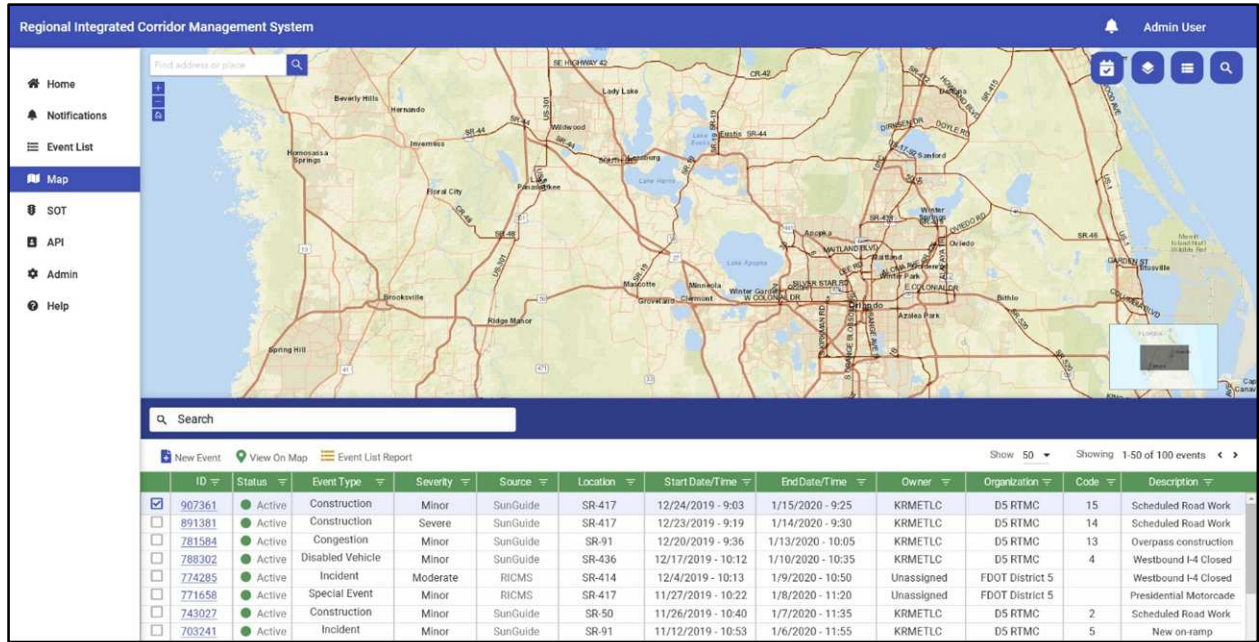


Figure 53 – Map Page Event List

3.4.3.3.2 Event Details

The Event List interface will provide users the capability to view and edit RICMS events and add comments to SunGuide events through the event details screens.

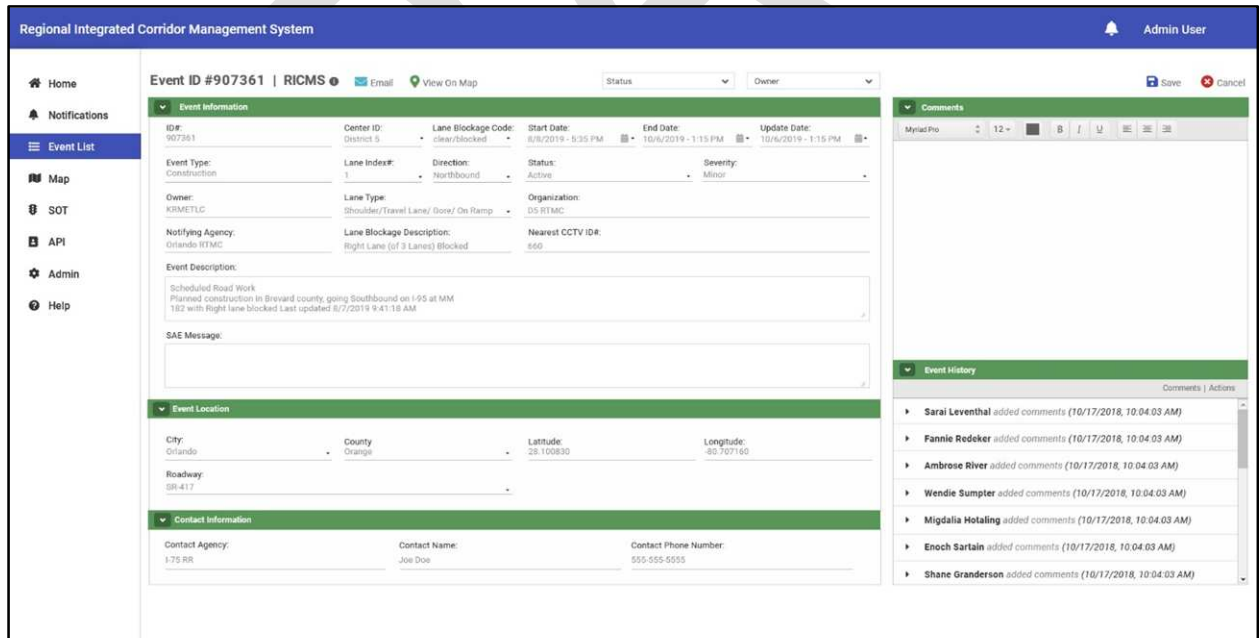


Figure 54 –Main Menu Event Details

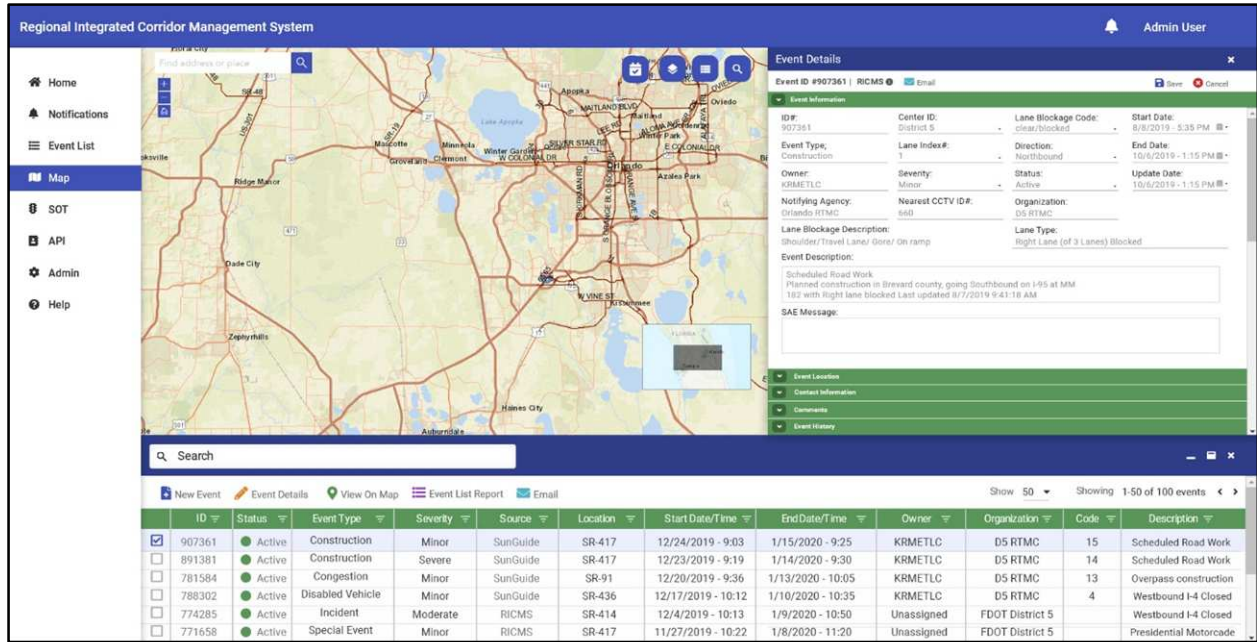


Figure 55 – Map Page Event Details

3.4.3.3.2.1 Lane Blockage Diagram

The RICMS system will ingest and display SunGuide Events lane blockage information within the event details. The lane blockage diagram shall include the number of lanes, lane types and lane blockage status for each individual lane. SunGuide Events lane blockage diagrams will not be editable from the RICMS system.

R-ICMS events will provide the capability for users to capture and edit lane blockage information directly within the event details. RICMS events lane blockage diagrams will only live within the RICMS application.

When a location for a new RICMS event is manually selected on the map, a default lane configuration will be configured based on available ITSQA data and displayed. The default lane configuration will then provide the capability to add/remove lanes, edit lane types, add/remove approaches and edit lane blockage status for each lane.

System Design Document for Regional Integrated Corridor Management System (R-ICMS)

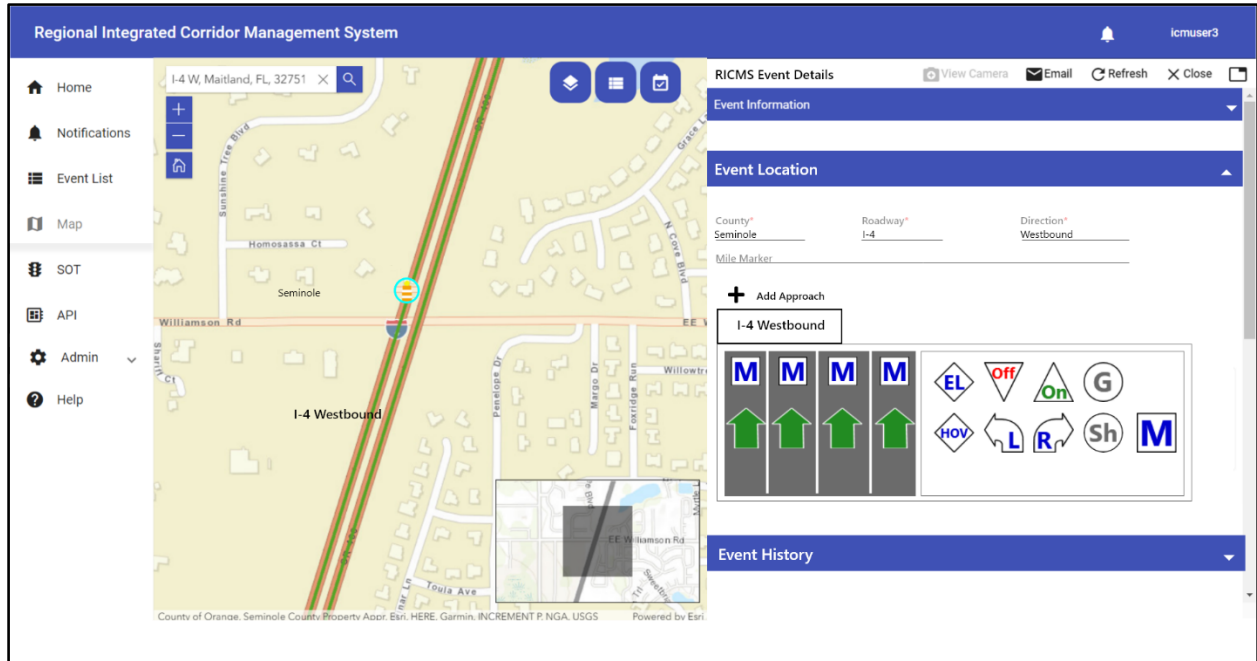


Figure 43 – Default Lane Configuration

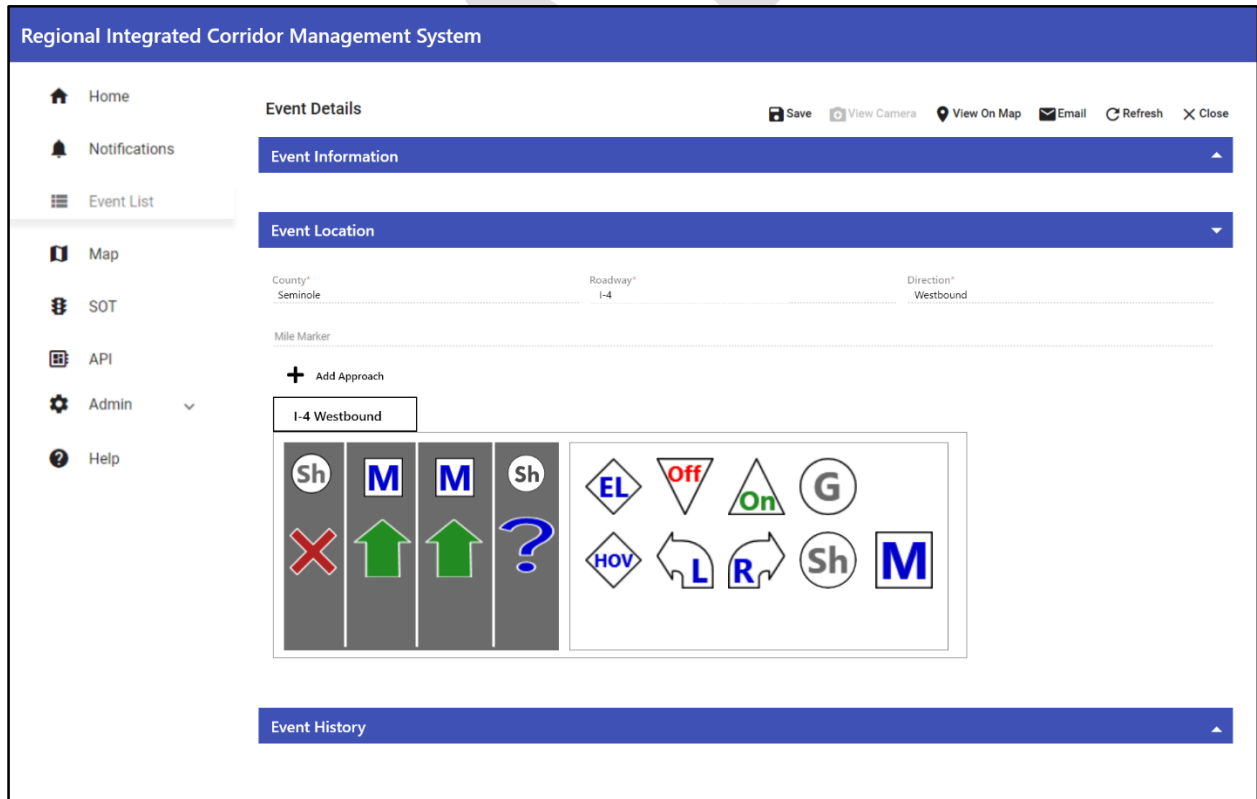


Figure 44 – User Updated Lane Type & Lane Status

System Design Document for Regional Integrated Corridor Management System (R-ICMS)

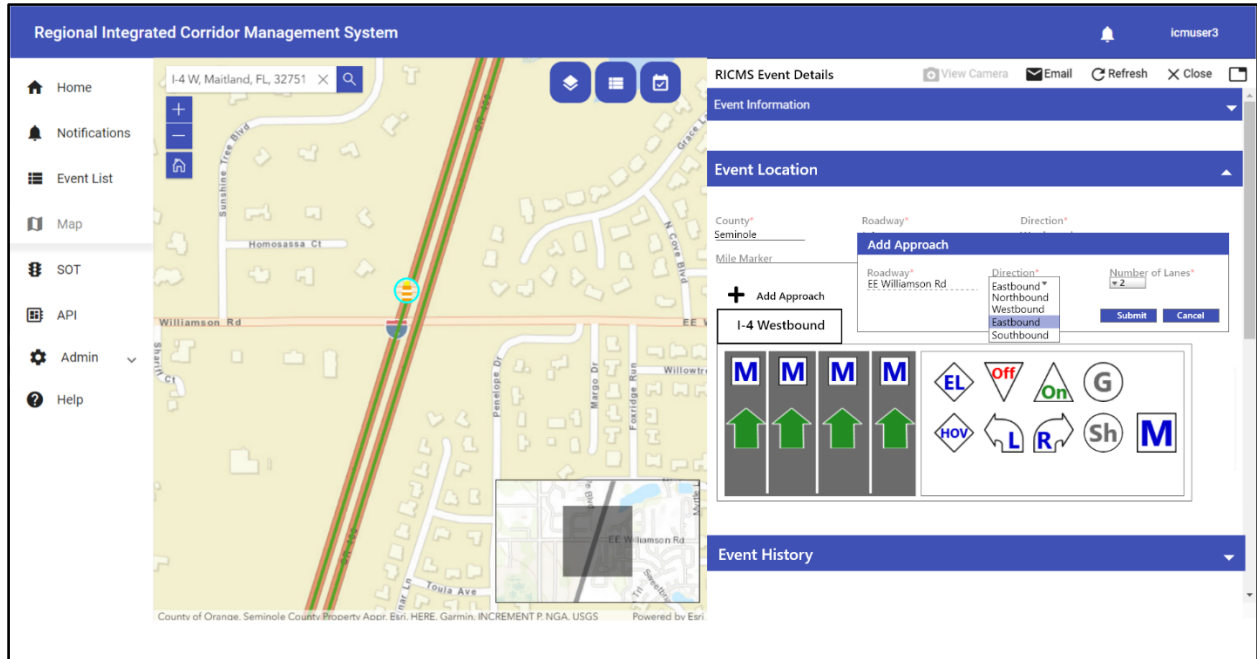


Figure 45 – Add Approach

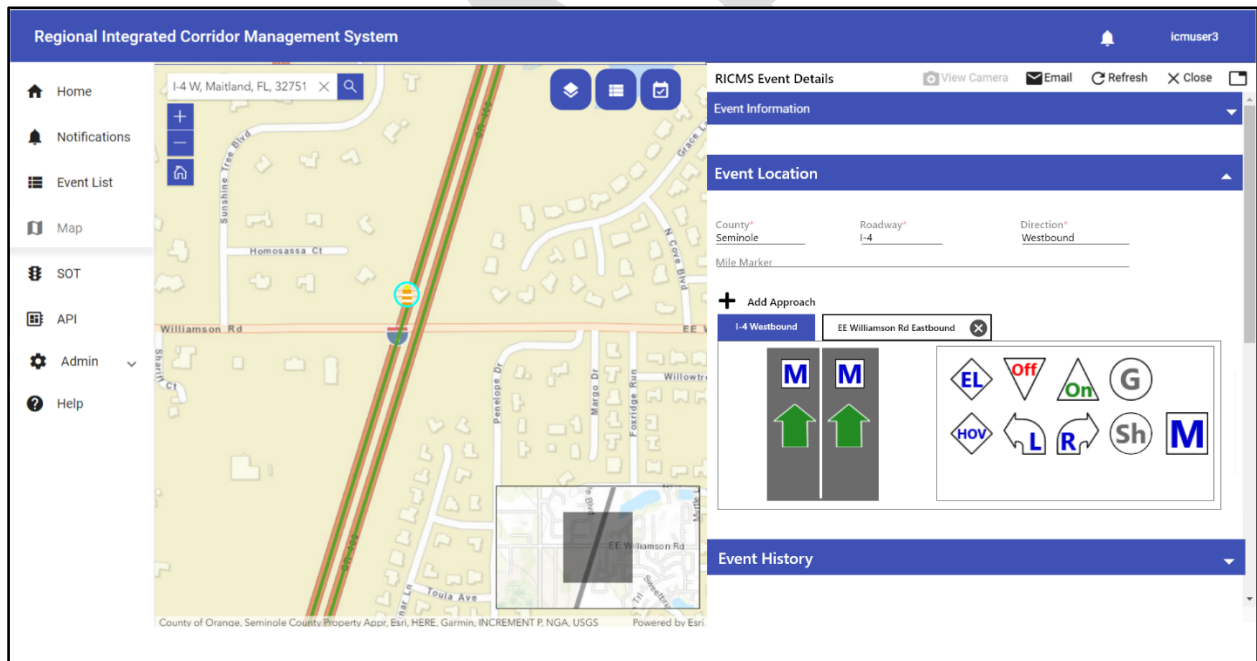


Figure 46 – Default Lanes After Added Approach

3.4.3.3.2.2 Event Nearerst CCTV

For SunGuide Events, Users will be able to click on the View Camera button to view the DIVAS feed for the nearest CCTV camera determined from the Events data.

System Design Document for Regional Integrated Corridor Management System (R-ICMS)

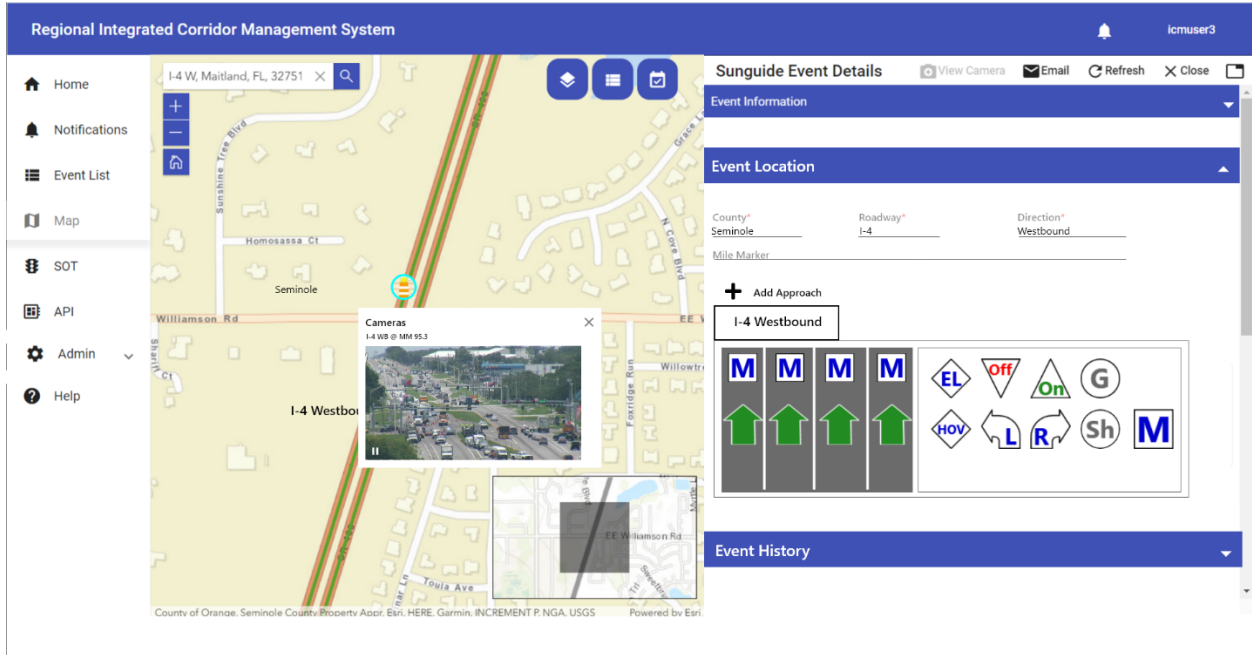


Figure 47 – Camera - Event Detail

3.4.3.3.3 Create RICMS Events

The Event List interface will provide users with the capability to create new RICMS events from the left main menu or from the map page.

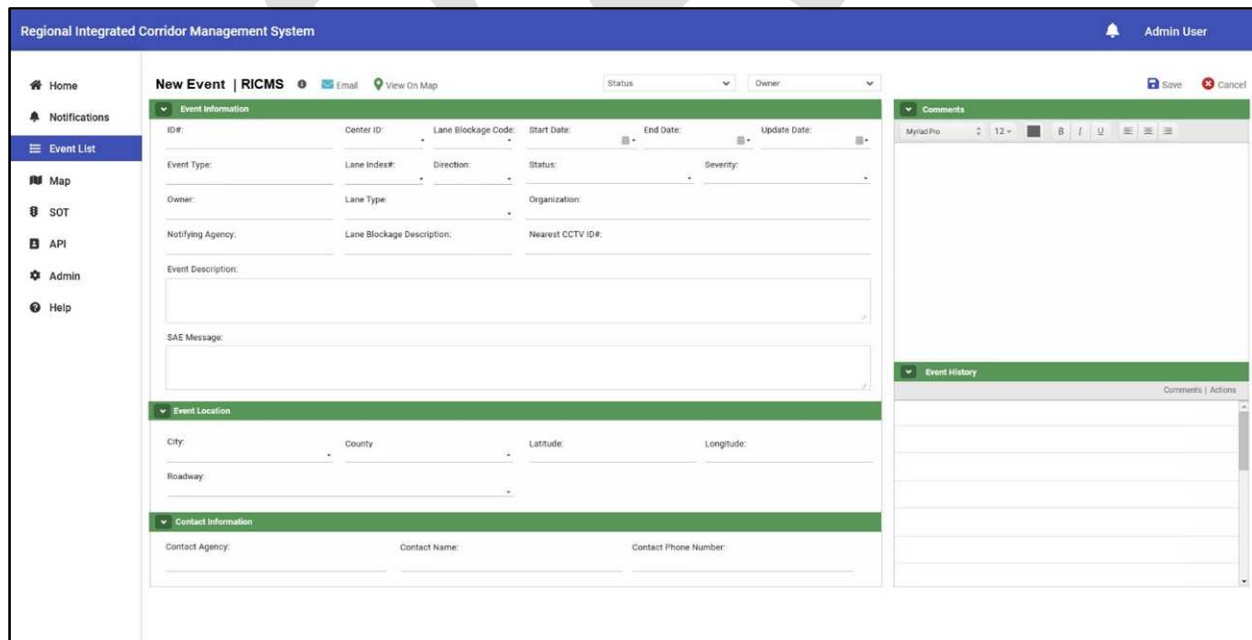


Figure 48 – Main Menu Create Event

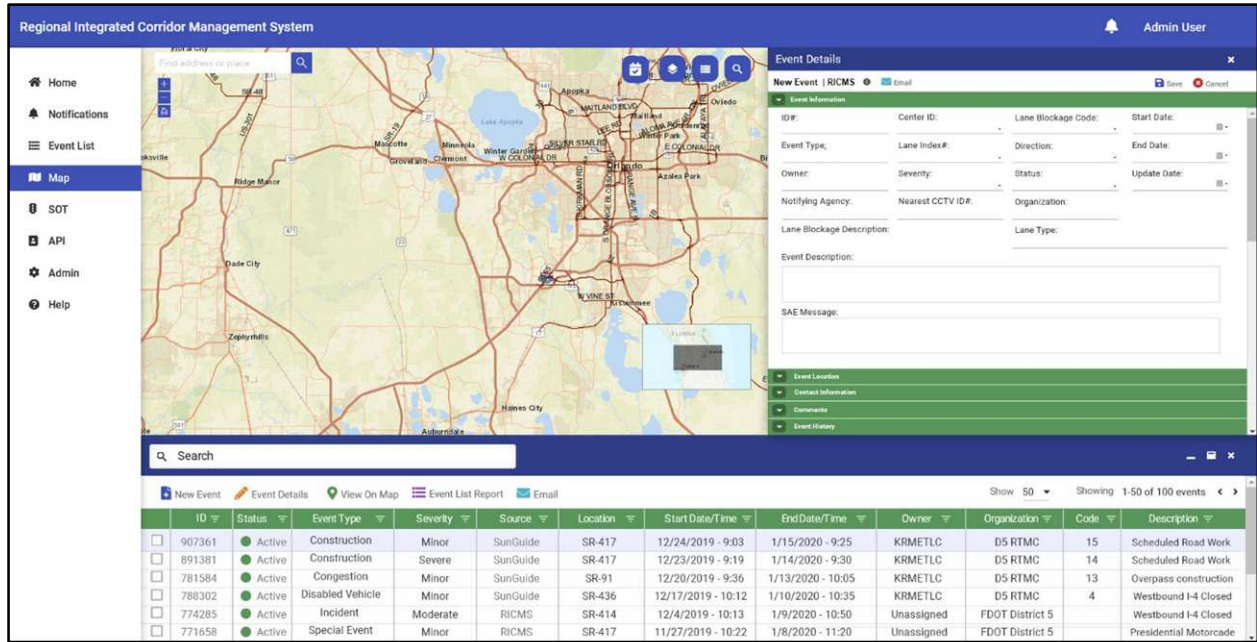


Figure 49 – Map Page Create Event

3.4.3.3.4 View Event on the Map

The Event list interface will provide users the capability to view events on the map. The user will be able to select the events that are desired to be viewed on the map and the map will zoom to those events at an appropriate level based on the ability to view all selected events.

System Design Document for Regional Integrated Corridor Management System (R-ICMS)

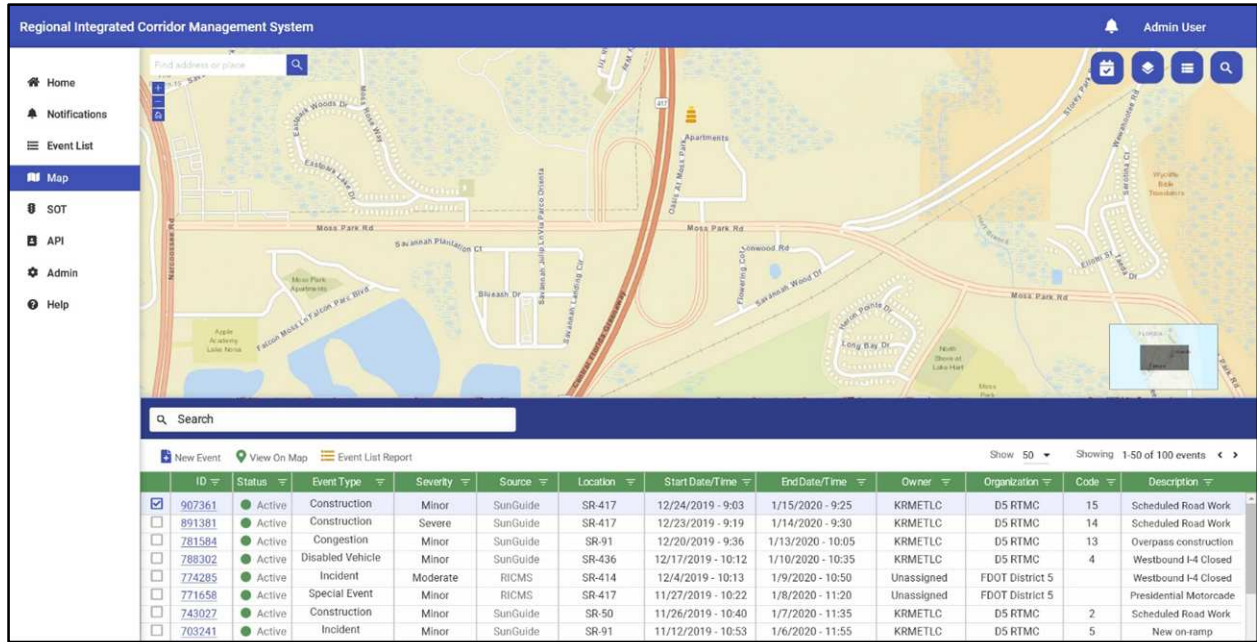


Figure 50 – View Event on Map

3.4.3.3.5 Email Event

The Event List interface will provide users the ability to send event information in an email when accessed from the main menu or the map page. This interface will allow users to email event detail information to a preconfigured default distribution list and provide the ability to add additional email recipients and remove any email recipients at the time of the email creation.

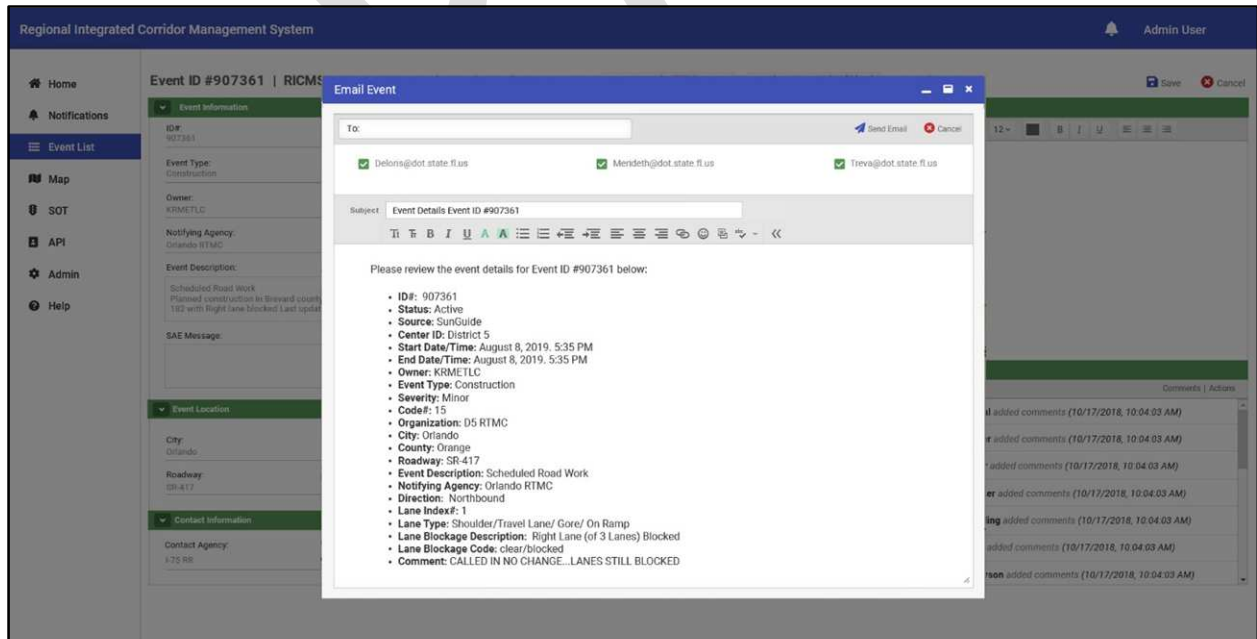


Figure 56 – Email Event

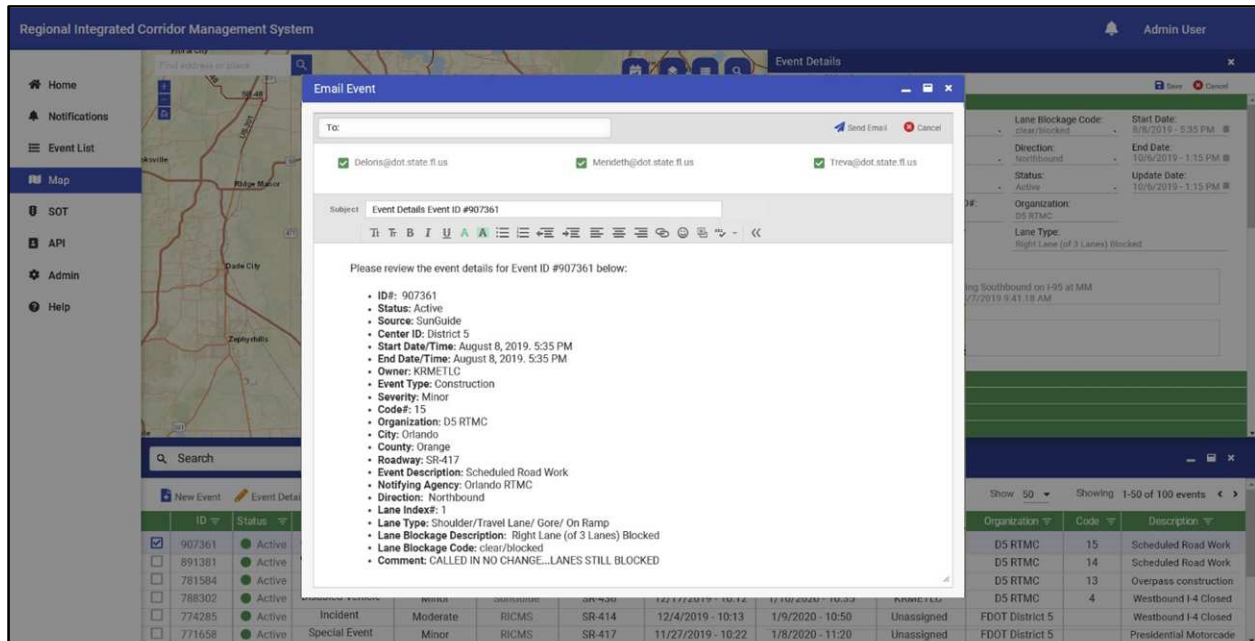


Figure 57 – Map Page Email Event

3.4.3.3.6 Event List Report

The event list report will be comprised of the data that is available on the event list at the time of the report creation. The event list report will be able to be exported in an excel for .pdf format. The Overall look and feel to be determined by FDOT staff.

3.4.3.4 Authorization (AuthZ-UI)

The authorization component of the User Interface will be the frontend to the Authorization Business Service. It will provide the necessary functions to manage roles, permissions, device groups, and devices. All authorization is tied to FDOT's Active Directory (AD). The system will tie all device groups and roles to Active Directory Groups that users can be assigned to. For each device group or role the user will be able to select the Active Directory Distinguished Name, an ID referencing the Active Directory Group that was previously setup by FDOT IT. This will in turn populate the Active Directory Group Guid, a read-only unique value that users can tie to AD to ensure they have selected the right group.

3.4.3.4.1 Device Group Management

Management of Device Groups is accomplished through a group of screens accessible from the main menu. The device group screen shown in Figure 58, lists the device groups defined in the

system. Using this screen, users with appropriate permissions can add new devices groups or edit existing device groups.

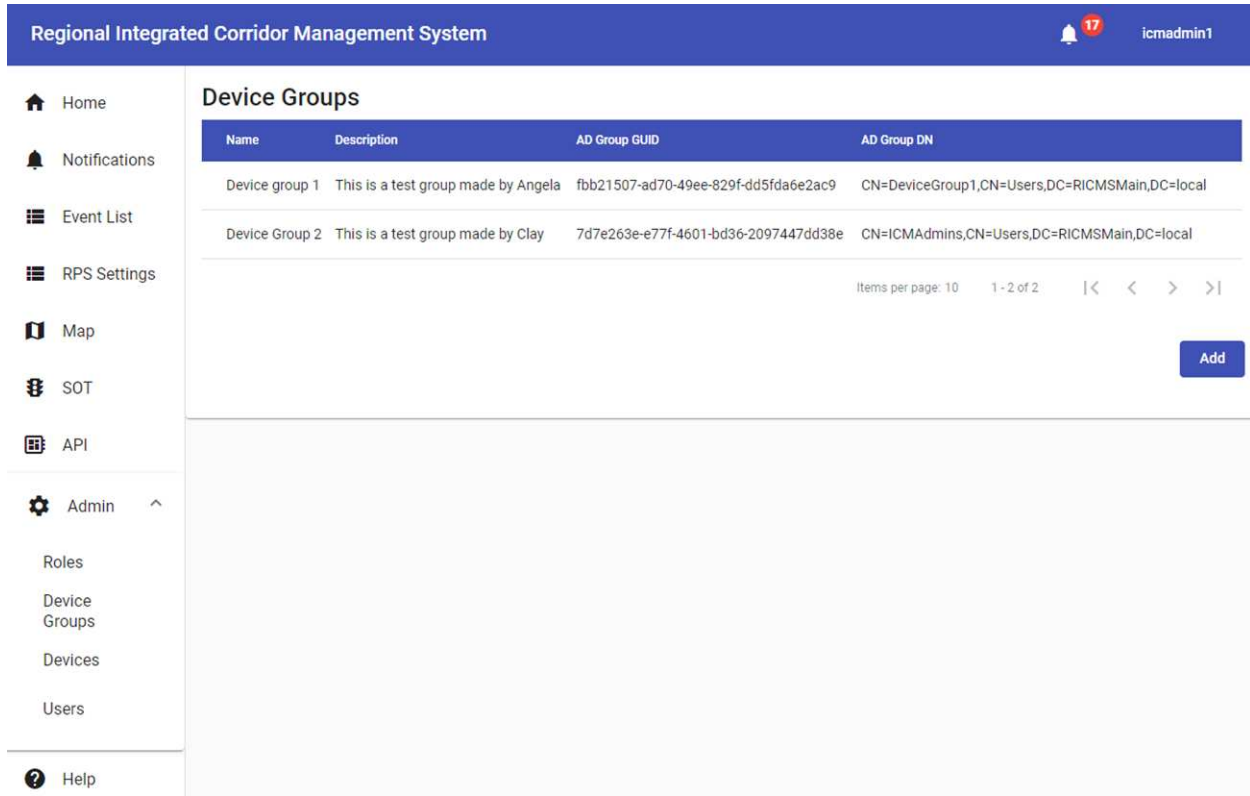


Figure 58 – Device Group List Screen

In response to a user clicking on the “Add” button, the system displays the “Add a device group” screen shown in Figure 59. A user completes the requested information and clicks the “Add”; note: the Guid field is auto-populated.

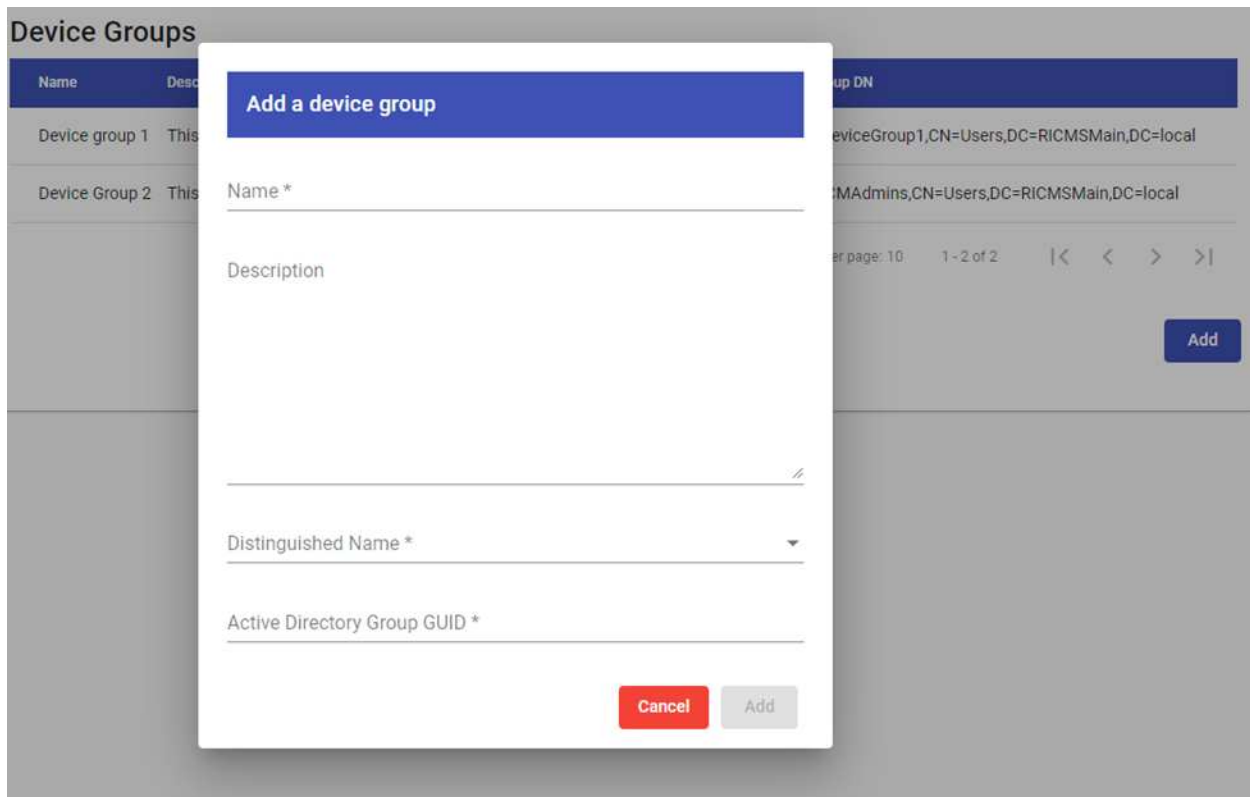


Figure 59 – Add Device Group Screen

Clicking on a row as shown in Figure 58 **Error! Reference source not found.** in the device group list will bring the user to that device group’s detail page as shown in Figure 60.

Device group 1

General Information

Name *
Device group 1

Description
This is a test group made by Angela

Distinguished Name *
CN=DeviceGroup1,CN=Users,DC=RICMSMain,DC=local

Active Directory Group GUID
fbb21507-ad70-49ee-829f-dd5fda6e2ac9

[Save](#) [Delete](#)

Contact Information

Contact Name
sam

Contact Email

Contact Number

[Save](#)

Approval Profile

Enabled	Mon	Tue	Wed	Thu	Fri	Sat	Sun	Holiday	Start Time	End Time	Status	Delay	
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	00:00	06:00	Approve	00:05	Delete

[Save](#) [Device Overrides](#)

Figure 60 – Device Group Detail Screen

The first section of the detail page allows a user to view or modify general information about the device group. The user can also delete the device group. The second section shows the contact information of the person responsible for Device Group settings. The third section provides the approval profile for a device group. The user can add, remove, or edit the timespans that make up the approval profile in-place in the table. The user can also modify the approval profiles of individual devices by clicking the device override button. The last section list devices that belong to the device group in question. By clicking edit, the user can move devices between device groups.

Figure 61 shows the functionality for moving devices between device groups, by selecting devices and using the arrow buttons. Both tables can be filtered to show only devices of certain types.

System Design Document for Regional Integrated Corridor Management System (R-ICMS)

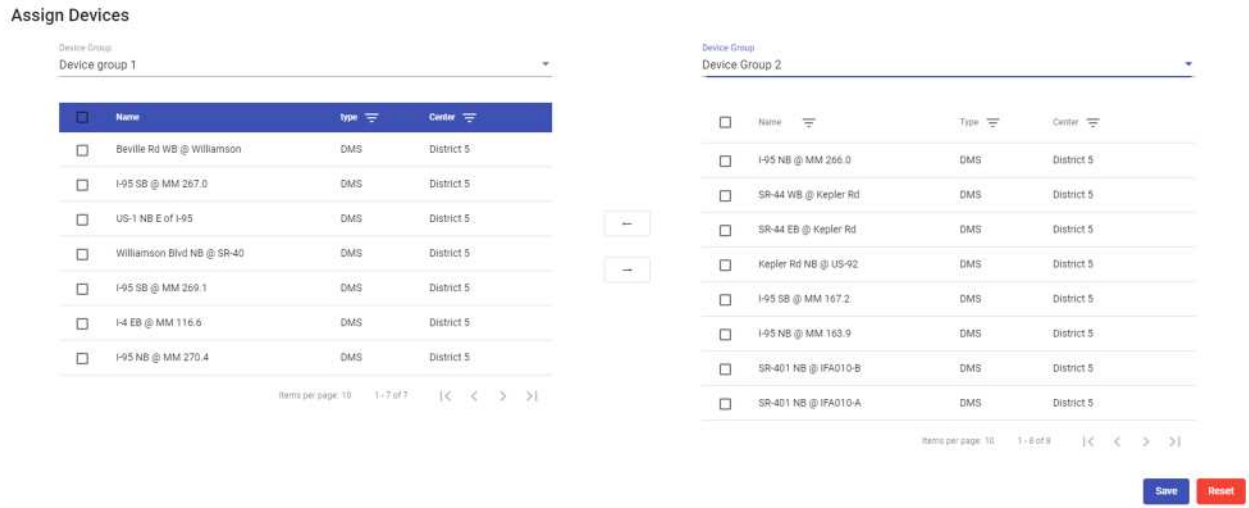


Figure 61 – Device Assignment Screen

Figure 62 shows the device list, filtered to only show the devices belonging to the device group from which the user originated. Clicking a row's edit button takes the user to that device's approval profile page.

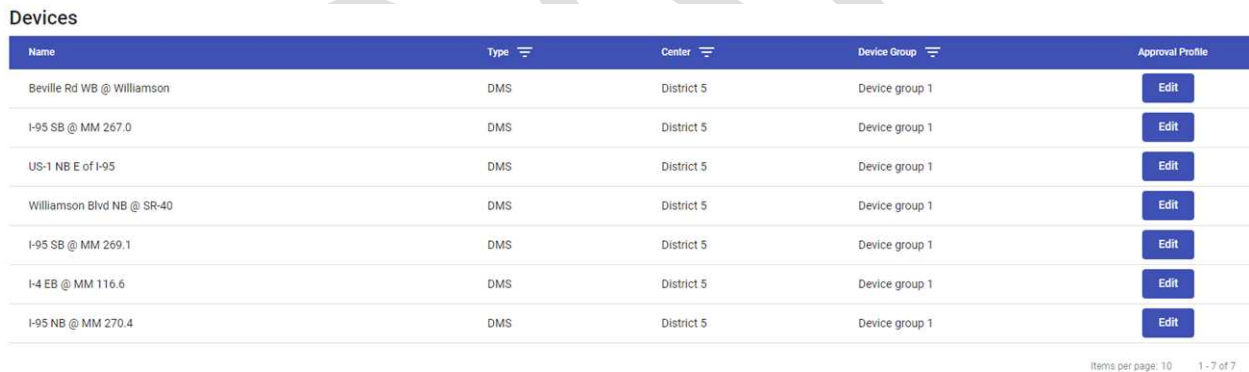


Figure 62 – Device List Screen

Figure 63 acts analogously to that of a device group's approval profile.

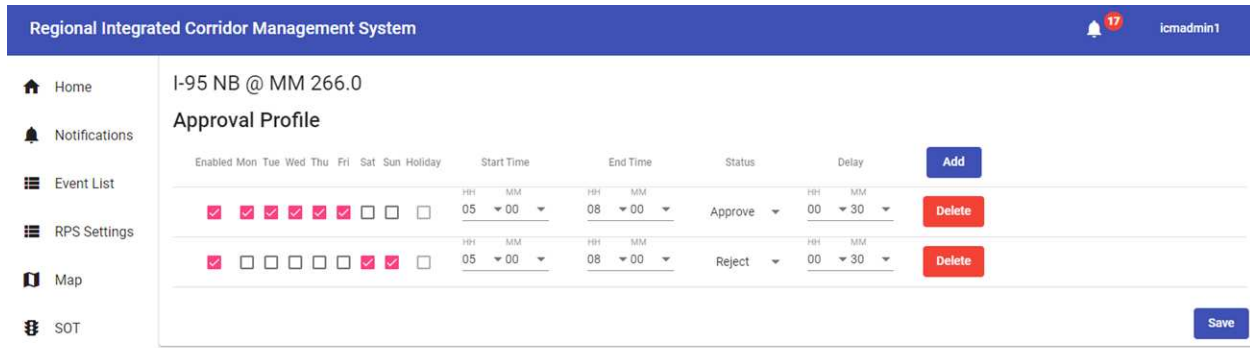


Figure 63 – Device Approval Profile Screen

3.4.3.4.2 Role Management

The roles list is very similar to the device group list. The roles in the system can be viewed, searched over, and filtered, and new roles can be added.

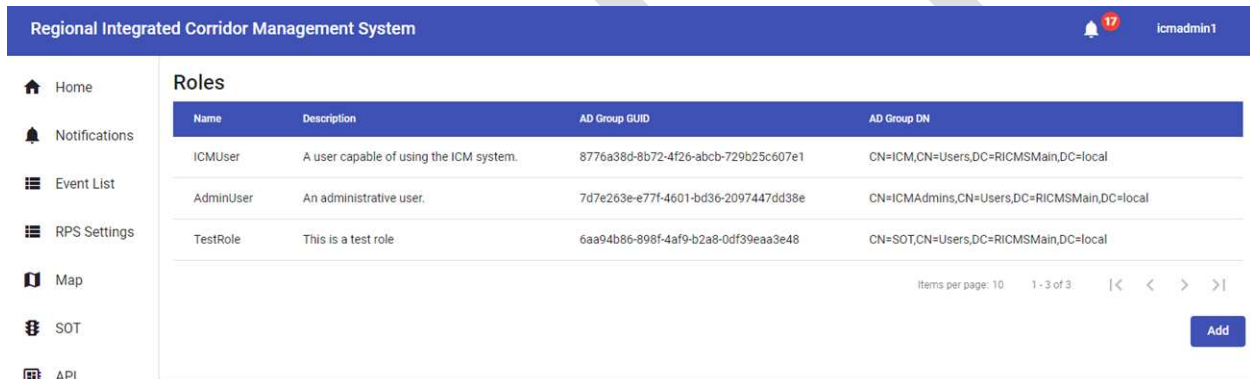


Figure 64 – Role List Screen

Adding a role follows the same process flow as adding a device group and is illustrated in Figure 65.

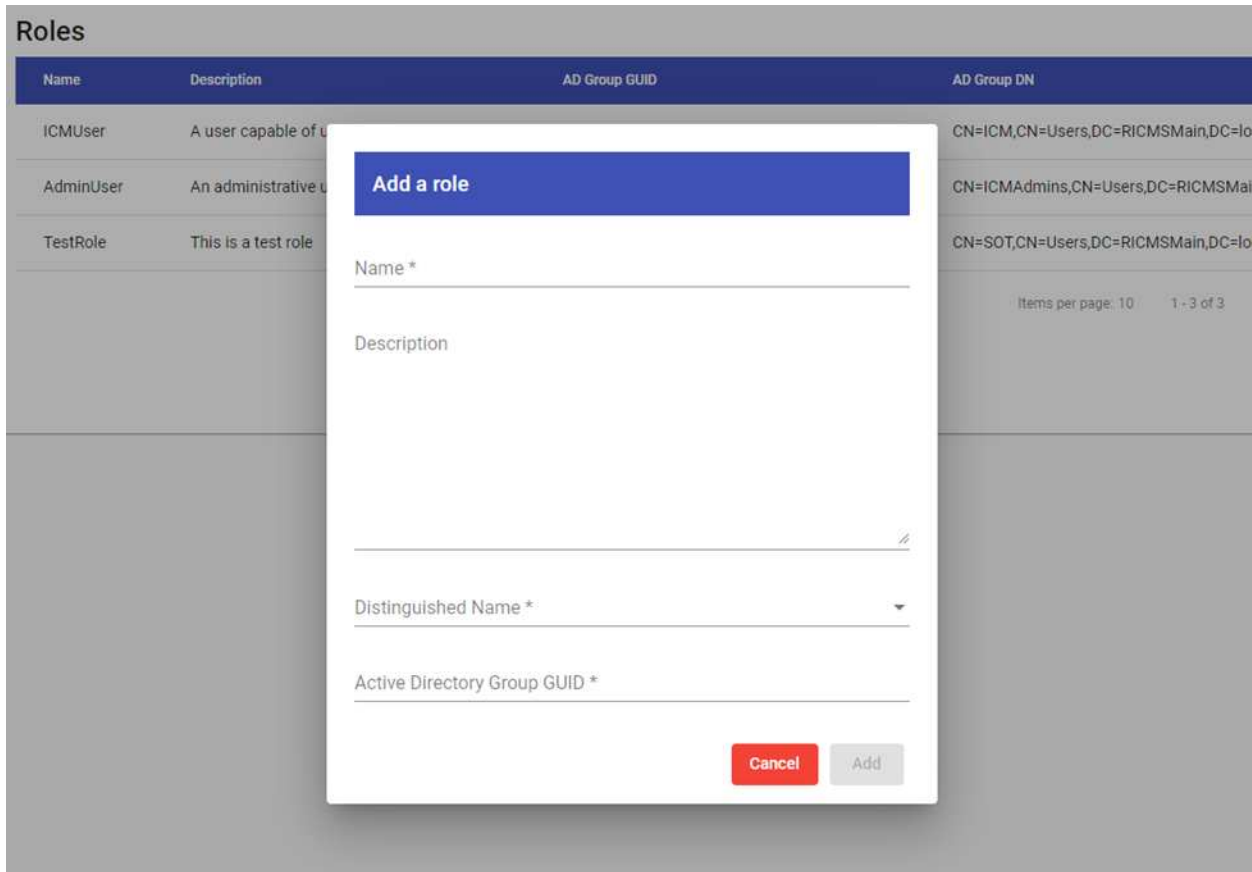


Figure 65 – Add Role Screen

Clicking on a row as in Figure 64 displays that role’s detail page.

In the top half of the Role Detail page in Figure 66, the general information associated with the role can be modified, and the role can be deleted. The bottom half shows the permissions belonging to the role. Clicking edit brings up the Modify Permissions Page in **Error! Reference source not found.** This table shows the permissions available to the system, and those that are checked belong to this particular role. By checking and unchecking rows, the user can add or remove permissions to and from the role.

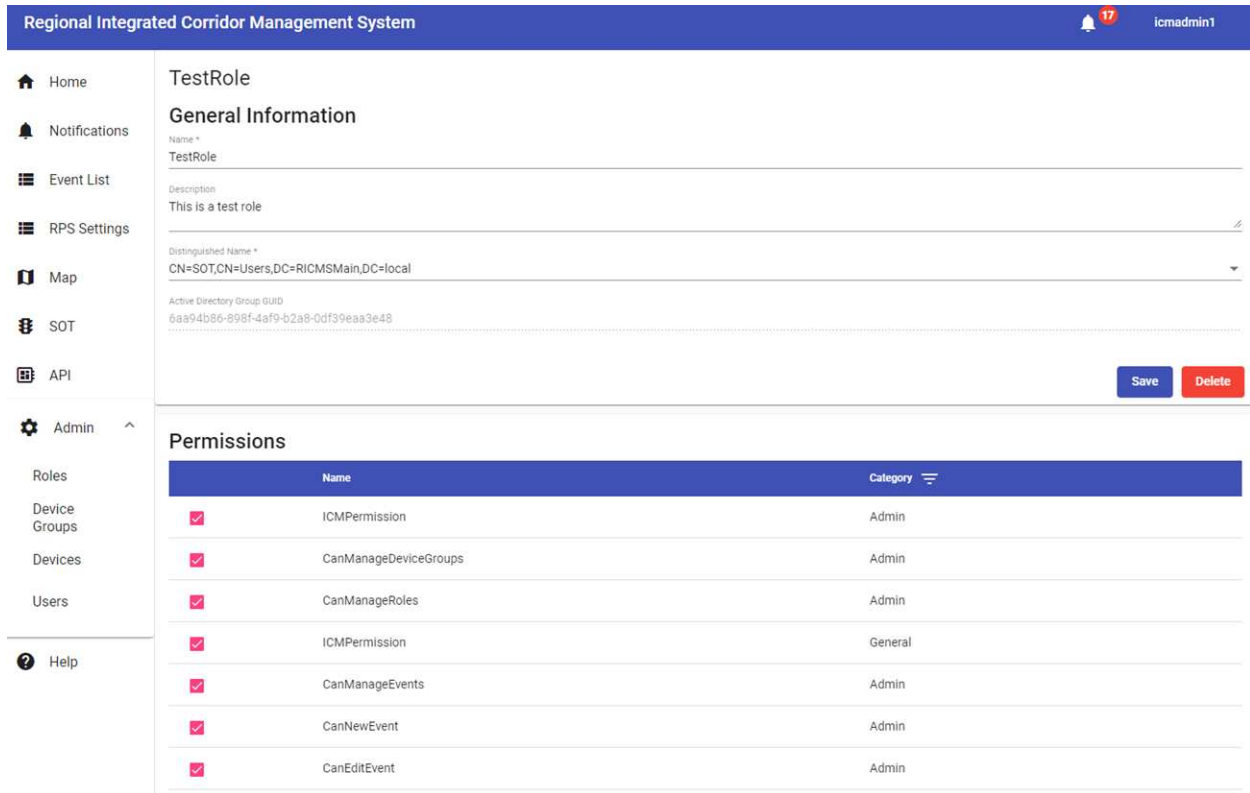


Figure 66 – Role Detail Screen

3.4.3.5 Notifications (NOT-UI)

The Notification component (also known as the information feed) of the UI will be responsible for display of user notifications. A user will be able to open and close their notification window.

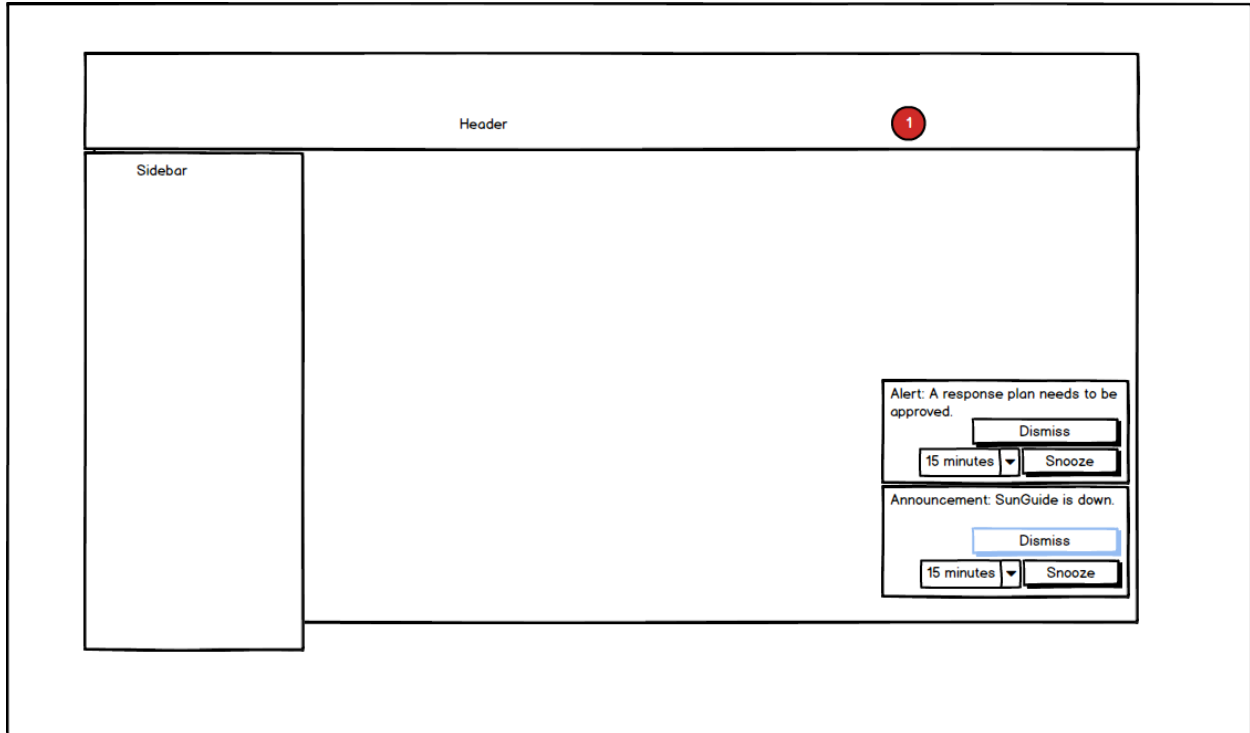


Figure 67 – Notification Popup Dismissal

There are two parts of the notification UI, the notification feed and popups. Any notification marked with critical importance will create a popup on the user's screen.

- The user can dismiss these pop-ups as shown in Figure 67, which removes them from the page.
- The user can also snooze popups as shown in Figure 68, which temporarily hides them from the page, and then after a configurable amount of time, causes them to reappear.
- After the snooze period has lapsed, the notification appears as in Figure 69
- The red indicator in the header, see figure Figure 69 shows the number of unresolved alerts the user has. By clicking on the indicator, the user can bring up the notification feed.

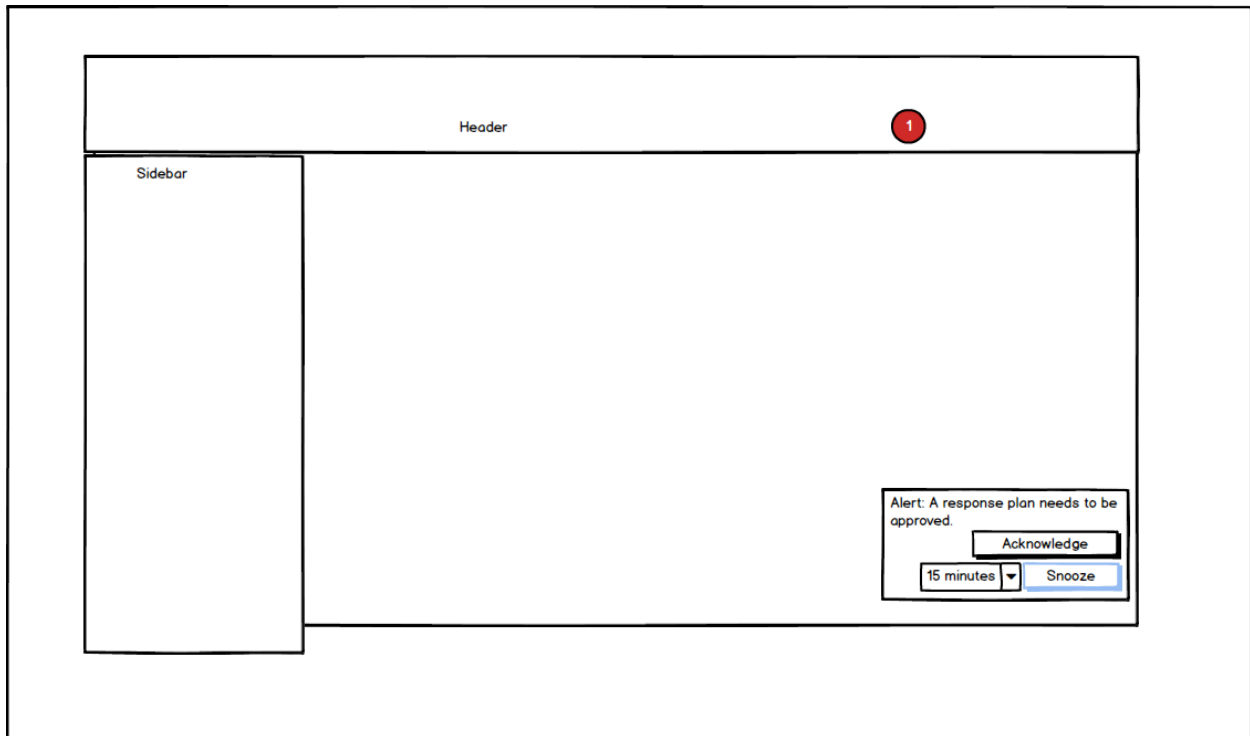


Figure 68 – Notification Popup Snoozing

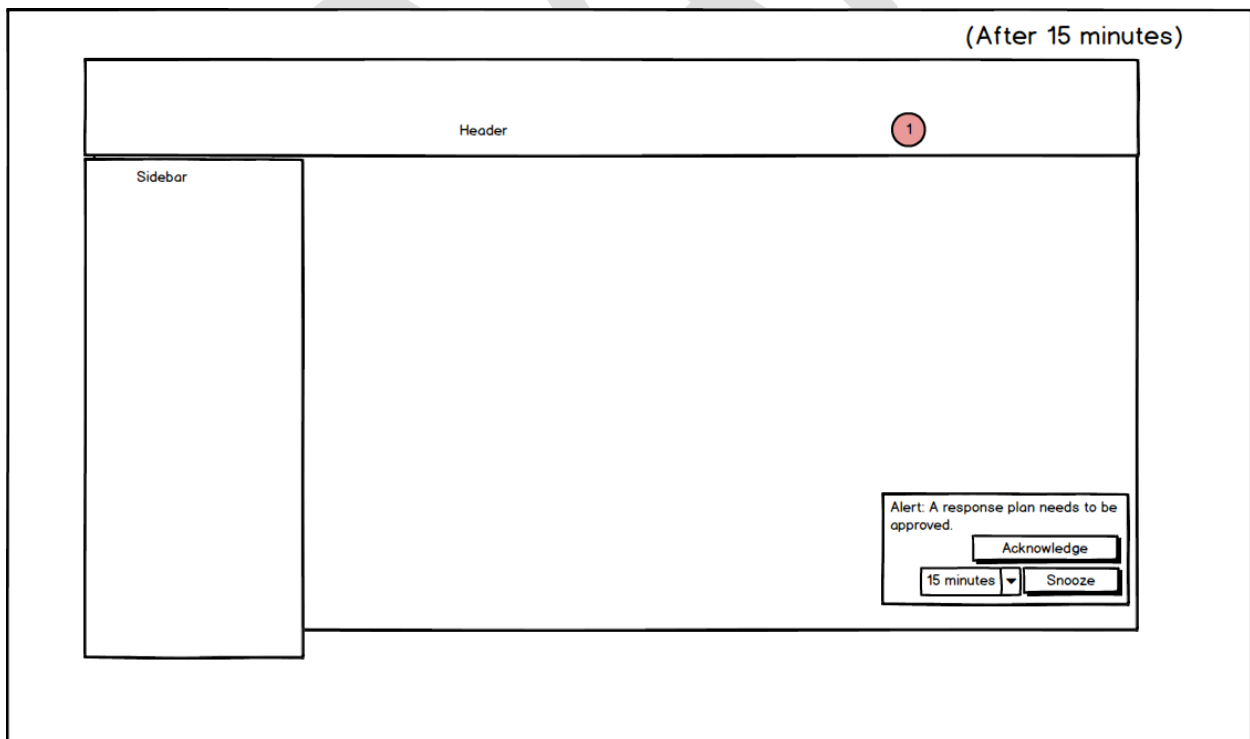


Figure 69 – Notification Indicator

The notification feed in Figure 70 has two panels. The left panel shows the user’s unresolved alerts. Notice that it matches the number in the red indicator. The right panel shows all of the user’s other notifications, with the most recent on top. The rows will be color-coded to indicate if a notification is minor or critical.

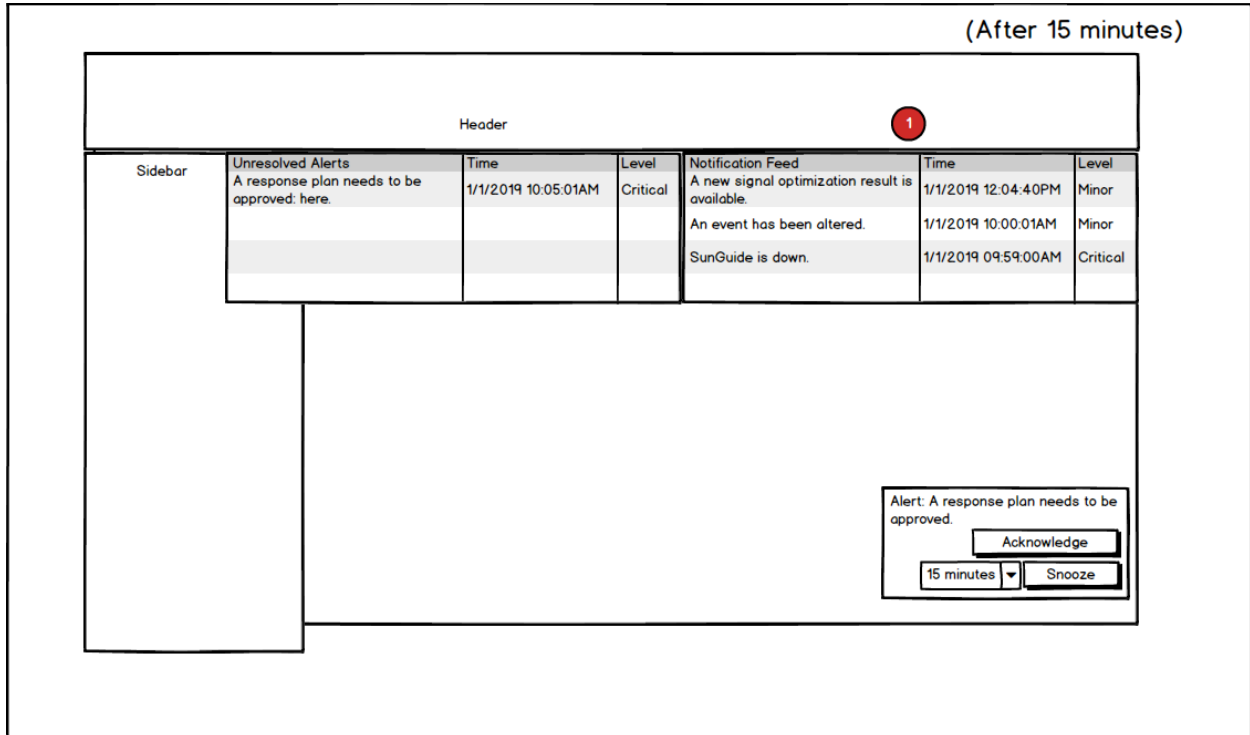


Figure 70 – Notification Feed

As new notifications come in, they appear at the top of the right panel, pushing older notifications down. The user can page through the feed, and filter on different fields.

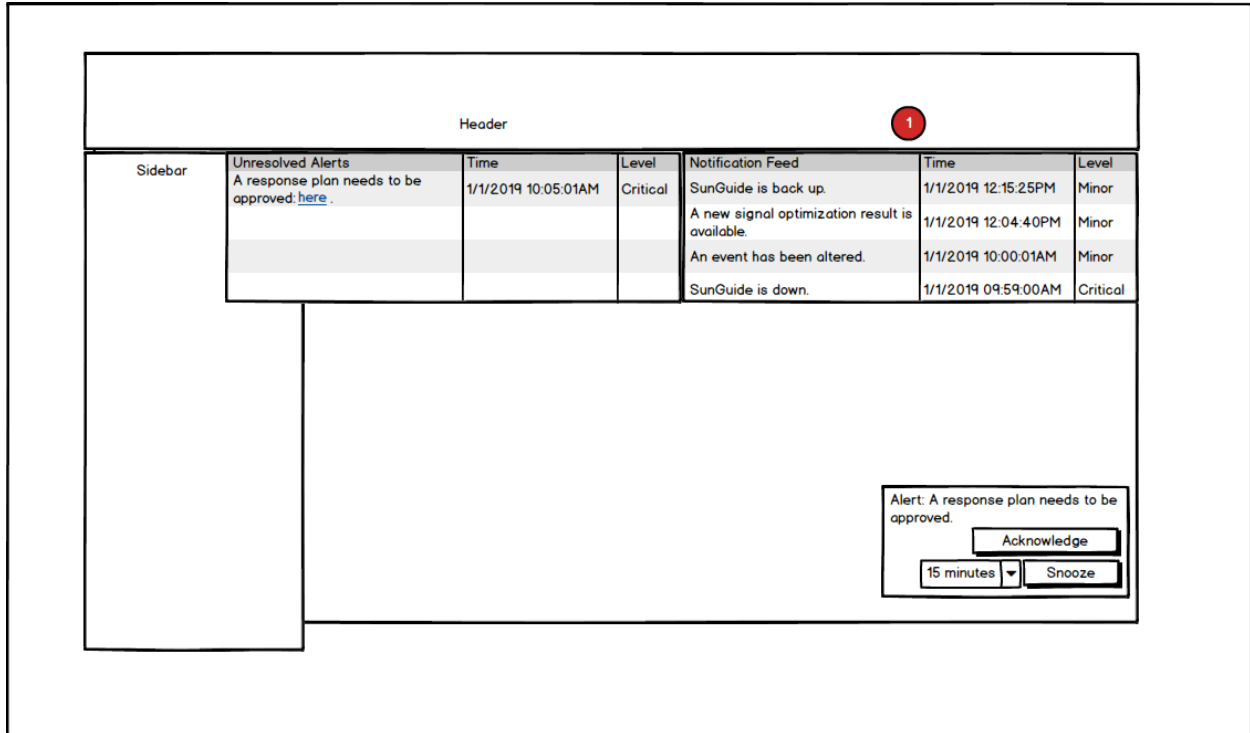


Figure 71 – Notification Feed with New Notification

Unresolved alerts often contain links to redirect the user to the appropriate page to resolve them. For example, when a response plan needs to be evaluated as shown in Figure 71, clicking the link will redirect the user to a page in which the user can approve or reject the plan (see **Figure 72**).

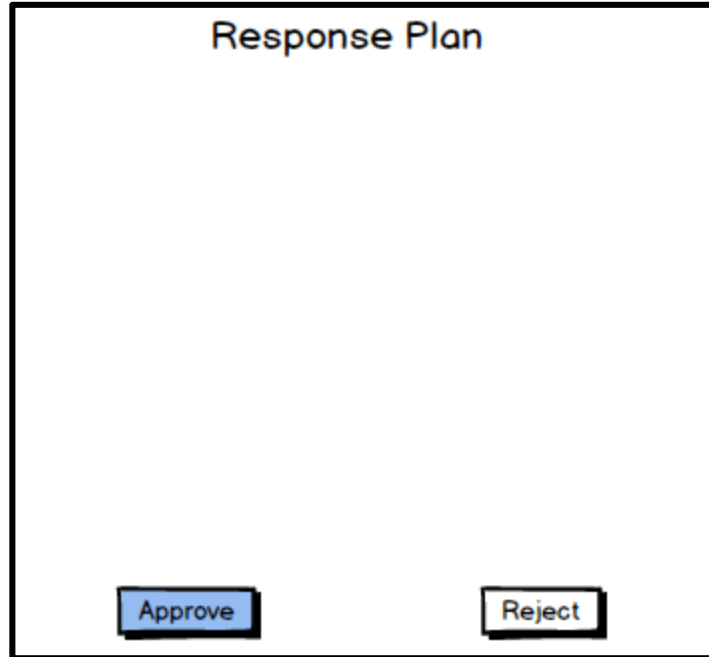


Figure 72 – Alert Resolution

Approving the response plan would resolve that alert. Notice in **Figure 73** that the alert has disappeared from the right panel, the popup has disappeared, and the indicator has decreased to 0. Once an alert has been resolved, it will transform into a normal notification and appear on the right panel.

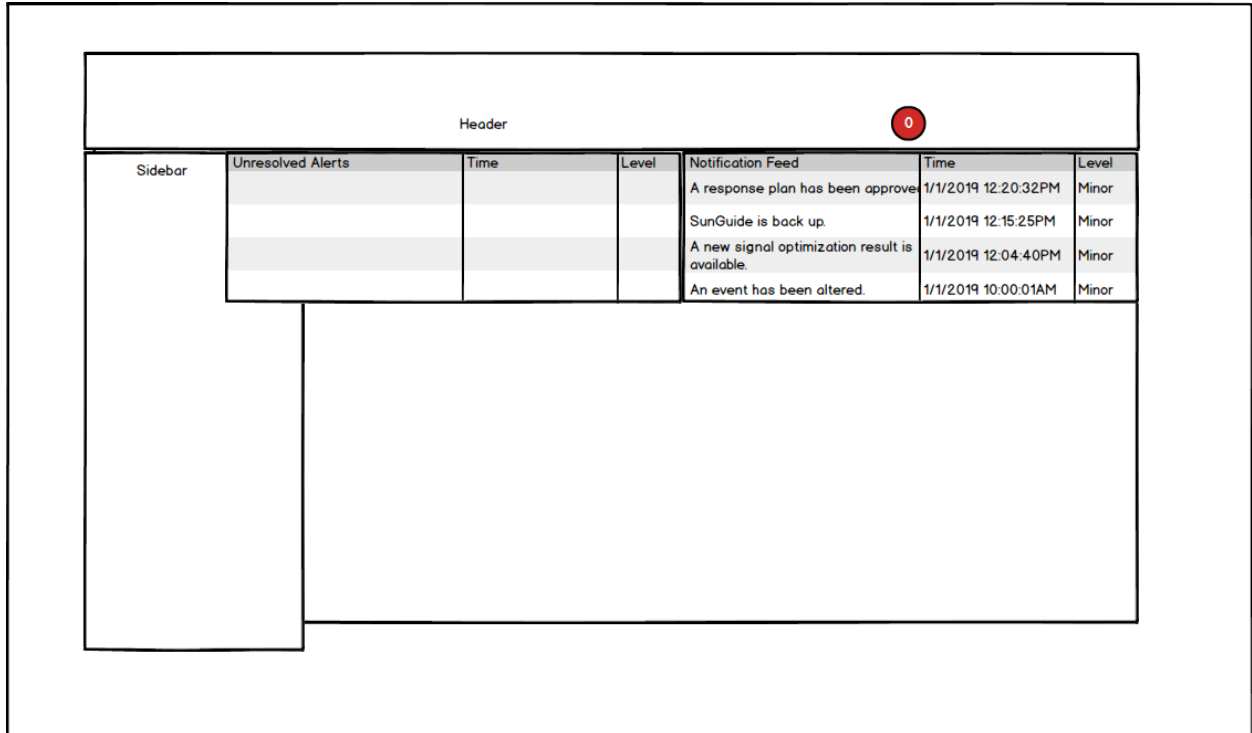


Figure 73 – Notification Feed with Resolved Alert

3.4.3.6 Reporting (RP-UI)

The Reporting component of the UI will allow for users to run defined template reports on the system. It will interface with the RPT-BS to access data in the data stores and return the data to the UI. The following figure shows the reporting architecture.

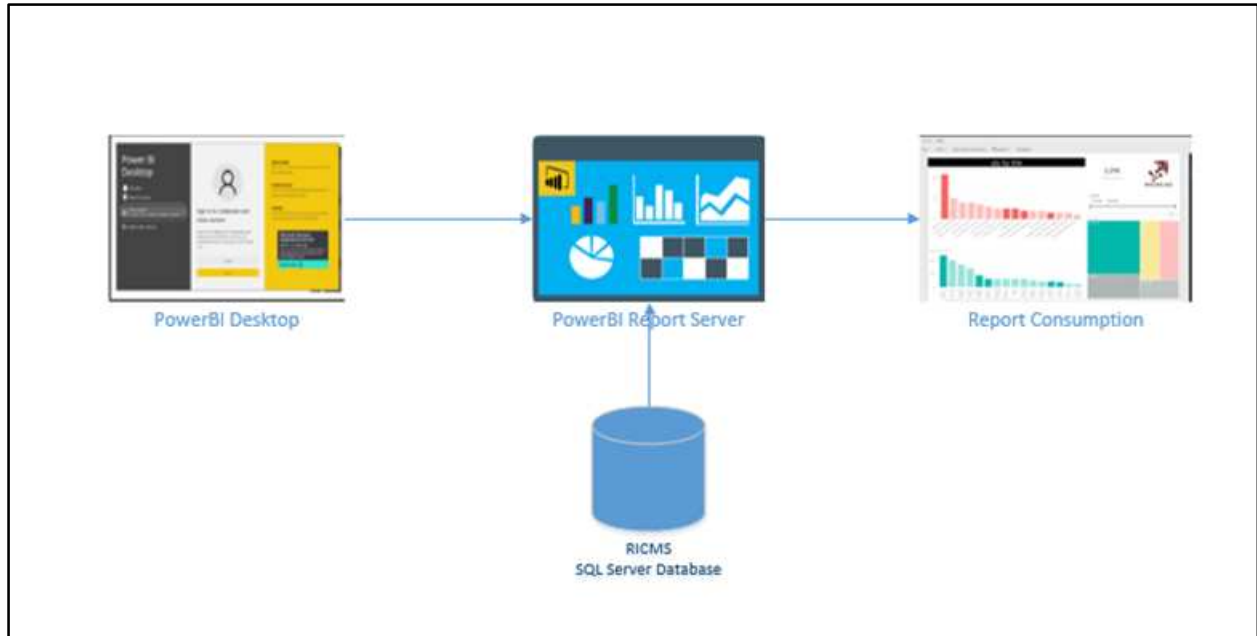


Figure 74 – Response and Repair Time SLA Parameters & LDs

The following six reports identified and selected by FDOT stakeholders will be provided for the RICMS application:

1. R-ICMS Event Summary Report

Data Source: SQL Server

Delivery Method: Dashboard/Report

a. Parameters:

- i. Date Range (overall beginning and ending date of the reporting period)
- ii. Day of Week (checkboxes for each day of week to be applied to each week in the date range)
- iii. Time Range (to be applied to each day within the date range)
- iv. Boolean parameter to include detail log, i.e. event actions (analogous to chronology in SunGuide) grouped by event and sorted by datetime
- v. Additional optional parameters for filtering by event attributes, including, but not limited to:
 1. Event ID Range: (int, int)
 2. Minimum number of travel lanes blocked: (int)
 3. Include any event with all lanes blocked: (Boolean)
 4. County (dropdown)
 5. Roadway (dropdown) (Note: this should not necessarily cut off at the county line when county is omitted)
 6. Roadway direction (dropdown)
 7. Milemarker or latitude/longitude Spatial Range – some way to specify the start and ending point along the roadway between which to include events

b. Output:

i. Report Header Section

1. Aggregation of events by a few important attributes
 - a. Time of day (or just by AM, Off peak, and PM instead of by every hour) (bar chart)
 - b. By roadway (pie chart)
 - c. By event type (pie chart)
 - d. By severity (pie chart)
 - e. By max number of lanes blocked
2. Aggregation of device approval requests (purpose to see how well we are getting things approved or if there's an agency holding us up)
 - a. Show aggregated number of approval requests, average, min, and max time for approval
 - b. Group by agency
 - i. For each hour of the day show number of approval requests, average, min, and max time for approval
 - c. Group by agency
 - i. For each day of the week show number of approval requests, average, min, and max time for approval

ii. Report Body Section

1. Grouped by event
2. Group Item (event) header contains event attributes
3. Item (event) body contains event chronology if selected from parameter

2. SOT Corridor Optimization Report (this is the one an engineer would sign and seal)

Data Source: SQL Server

Delivery Method: Direct download from SOT

a. Report Parameters:

- i. Single Optimization must be selected from a list of optimization runs having basic attributes displayed
 1. Filter parameters to narrow down the list:
 - a. Optimization ID or Range
 - b. Date Range of Optimization
 - c. Optimization Status (drop down)
 - d. User requesting the optimization (drop down)

b. Report Output:

- i. Show comprising roadway and intersections included
- ii. Show Time of Day Plan (from the clustering algorithm)

- iii. Show aggregate output results (applicability, fit improvement, output from HCS7, and output from Aimsun, etc.)
 - iv. Show SOT run output details per signal
 - 1. Optimization Inputs
 - a. Selections, global values,
 - b. Date and time ranges of volumes used (should be able to use this information to grab the input volumes from the API)
 - 2. Optimization output results (applicability, fit improvement, etc.)
 - 3. Optimized timing patterns (should follow the format of the agency timing sheets provided by Jay, Manny, and Tricia)
 - v. Show a sign and seal box with the instructions to sign and seal after making field adjustments and inputting those adjustments into the SOT before signing.
3. Response and Repair Time Performance Report
- Data Source: JIRA
- Delivery Method: Monthly Status Report
- a. Report Parameters
 - i. Date Range
 - b. Report Output
 - i. Report Header Section

Performance Metrics Definitions: (copy of table 4 from scope):

Definition	KPI	System Acceptance Test Measurement Requirement	Operations Measurement Period/Sample	Liquidated Damages*
<p>Priority 1 issue is defined as any failure that will result in: loss of ability to create response plans; inability to accurately create response plans; inability to send response plans to requested agencies; inability to collect data that would result in missing data in the archived data.</p> <p>Maintenance response time shall be measured from the time when the VENDOR receives notification of the maintenance event or failure, and ending when the VENDOR staff acknowledge the notification of the problem or acknowledge the associated alarm or alert in the network monitoring system (NMS) application.</p>	<p>Maximum response time of 15 minutes during normal operating hours of 6 AM – 7PM on weekdays.</p> <p>Maximum response time of 1 hour outside of normal operating hours.</p>	<p>NMS application will be utilized to report all events for this category during the test time period.</p>	<p>All events submitted monthly. Liquidate damage (LD) applies to individual events.</p>	<p>For every hour over the KPI, the VENDOR shall be subject to liquidated damages of 1%, with a maximum of 6% per day of the monthly support payment.</p>
<p>Repair time shall be measured from the time when the VENDOR receives notification of the maintenance event or failure and ending when the failure condition is corrected and the system is returned to normal operation.</p>	<p>Maximum repair time of 1 hour during normal operating hours of 6 AM – 7PM on weekdays.</p> <p>Maximum repair time of 4 hours outside of normal operating hours.</p>	<p>NMS application will be utilized to report all events for this category during the test time period.</p>	<p>All events submitted Monthly. LD applies to individual events.</p>	<p>For every hour over the KPI, the VENDOR shall be subject to liquidated damages of 1%, with a maximum of 6% per day of the monthly support payment.</p>
<p>Measurement Method:</p> <ol style="list-style-type: none"> System acceptance testing can be used to verify successful resolution System report to be provided by the VENDOR to indicate performance <p>Notes:</p> <ol style="list-style-type: none"> Provide a report and detail log of all Priority 1 events including: <ol style="list-style-type: none"> Loss of response plan generations is related to the loss of any system and/or hardware Exclusions include DEPARTMENT-directed postponements Report indicates Maintenance event, failure detection, notification time, and repair times for each event, and make clear those events that exceed the SLA Repair time is measured for each event Time duration between the event notification and repair for each ticket. Events will be tracked on an individual basis, and summarized by the VENDOR for monthly reporting The report will indicate all repair times and those that exceed the SLA increments, examples below: <ol style="list-style-type: none"> Event notification and response and repair occurs during normal business hours and at 3 hrs. and 30 mins. after event notification. <ol style="list-style-type: none"> SLA LD will be 3% for response, and SLA LD will be 2% for repair SLA LD will be 5% for both response and repair 				

Figure 75 – Response and Repair Time SLA Parameters & LDs

1. Aggregate performance information

- Show monthly support payment dollar amount against which to assess LD percentages
- Total number of response and repair time LD events, min, max, median percentage and LD assessment
- Total percentage and total LD assessment

ii. Report Detail Section

- List of all response and repair time LD events, and their relevant timestamps for calculating LD, and the assessed percentage and amount

4. System Availability Performance Report

Data Source: Jira

Delivery Method: Monthly Status Report

a. Report Parameters

- i. Date Range defaulting to prior month

b. Report Output:

- i. Header section:

- 1. Performance definitions and table from scope:

Table 5: Service Level Key Performance Indicators and Liquidated Damages

Service Level KPIs	Definition	KPI	Liquidated Damages*
Availability of Applications, including; DSS, IEN, DFE	An application will be considered unavailable if it is not functioning to a reasonable level of usability and ability to accomplish the ICMS operations described in section 1.6.	Maximum of 1% of downtime each month after deployment	For every hour over the KPI, the VENDOR shall be subject to liquidated damages of 1%, with a maximum of 6% per day of the monthly support payment.
	Measurement method: (Host Hardware and Applications) Host Availability (specific to each Host Hardware and Application) % = 1 – (total Host Hardware/Application downtime / (Expected time of operations – exclusions)) 1. SAT and Operations methods to be the same 2. System report to be provided by the VENDOR to indicate performance Notes: 1. Exclusions include all time when the system is not operating during preventative maintenance activities pre-approved by the DEPARTMENT, or due to damage beyond the VENDOR's control 2. DSS measurement excludes the PRE, and the EVE when the PRE is unavailable. 3. DFE measurement is related to entire system excluding components/software included 4. This addresses software applications that fail and are not available 5. Interfaces are included based in VENDOR interface availability only 6. SLA applies to each application independently, and excludes hardware availability 7. Need to identify the processes and hardware that impact the Host Hardware and Application and make sure failure of each will create an alarm message via automated Solar Winds monitoring with supporting system-generated availability reports being provided to the VENDOR 8. Identify if there are failures that cannot be identified through software failures alone 9. Identify the problems (such as external interfaces) where the VENDOR is not responsible; such alarms should be identified with explanation 10. SLA damages example: for a month (24hr x 30 days' x (100%-99%) = 8.2 hours of allowable down time, so for downtime in excess of SLA, LD are being calculated as: a. From 8.2 hr. to 9.2 hr. is 2% of monthly maintenance fee; and b. From 9.2 hr. to 10.2 hr. is 3% of monthly maintenance fee.		

Figure 76 – Service Level Key Performance Indicators and Liquidated Damages

2. Aggregate performance information

- a. Total monthly support payment against which to assess percentage penalties
- b. Total number of system availability LD events, min, max, median percentage and LD assessment
- c. Total percentage and total LD assessment

ii. Report Detail Section

1. List of all system availability LD events, and their relevant timestamps for calculating LD, and the assessed percentage and amount

5. System Response Plan Creation and Dissemination Performance Report

Data Source: SQL Server

Delivery Method: Dashboard

a. Report Parameters:

- i. Date Range
- ii. Include all events: Boolean

b. Report Output:

i. Report Header Section:

1. Performance Metrics Definitions: Delay as (a to b, b to c, d to e, e to f, and j to k but in more words)

Service Level KPIs	Definition	KPI	Liquidated Damages*
TC1 System Response Plan Creation and Dissemination Performance	<p>For traffic conditions that might cause a response plan activation, the following simplified flow occurs:</p> <ol style="list-style-type: none"> a) ERE rule triggers, ERE selects response plan set b) ERE sends response plan set to PRE c) PRE sends modelling task to the modelling software d) Modelling software sends results back to the ICMS e) PRE evaluates results, calculates MOEs, and sends results to DFE f) IEN displays results of simulation to ICM manager g) ICM manager selects plan h) IEN sends plan to affected agencies for approval i) Last approval received j) GO issued by ICM manager k) IEN sends implementation to SunGuide <p>Delay is computed by summing the delays a to b, b to c, d to e, e to f, j to k.</p> <p>For those steps that require connection to external systems (e.g. SunGuide, SMTP server, modeling engine), the step is considered complete when the attempted connection is initiated.</p>	<p>Minor: 2 minutes, Moderate: 4 minutes, and Major: 6 minutes</p>	<p>For every day that contains one or more occurrences of the ICMS exceeding the minor, moderate, or major KPIs, the VENDOR shall be subject to liquidated damages of 1%, 2%, or 3%, respectively, of the monthly support payment.</p>

Figure 77 – Performance Metrics Definitions

2. Aggregated performance information

a. Overall Average Delay

b. Average Day by time of day

ii. Report Detail Section

1. List of events showing datestamp of creation, event ID, and timeline with timestamps and durations of each

performance related activity, i.e. a, b, c, d, e, f, g, l, h, k with the durations that are part of the delay equation in bold.

6. DSS Report

Data Source: SQL Server

Delivery Method: Dashboard

a. Report Parameters:

- a. Date Range (overall beginning and ending date of the reporting period)
- b. Day of Week (checkboxes for each day of week to be applied to each week in the date range)
- c. Boolean parameter to include detail log, i.e. each simulator request

b. Report Output:

i. Report Header Section:

1. Aggregate event counts:

- a. Stacked bar chart showing the following aggregate values on the y-axis. The x-axis represents each hour of the day: (note: it may make sense to order the y-axis values with events detected on the bottom and the more specific conditions build upwards along the y-axis). Show a grand total column that aggregates across the entire X-axis

- i. Events detected
- ii. Events with travel lanes blockage or congestion
- iii. Events with [R-ICMS Diversion Route] response plan candidate(s)
- iv. Events with response plan simulated candidate(s)
- v. Events with beneficial response plan simulated candidate(s) having greater score than do-nothing
- vi. Events with approved response plan candidate(s) having all approvals from agencies needing approval
- vii. Events with response plans suggestions (sent to SunGuide)
- viii. Events with response plans activated

- ix. Events with reevaluated and modified response plans activated prior to termination of response plans for the event
 - b. (in the future, after we get more roadways covered than I-4: have the same chart but broken down by roadway instead of hour of day)
 - ii. Report Details Section:
 - 1. Group by event (show basic event attributes in the event item header),
 - a. Subgroup by response plan candidates simulated together (show the datetime of the candidate identifications), then by individual simulation (show the details of the candidate response plan
 - i. Show the priority score, and how long the simulation has to wait in the queue, and the time it took to perform the simulation.
 - ii. Show scores and results of the simulation – whatever we can get back from Aimsun including the QEH or other simulation quality metrics

Users will have the capability to add new defined template reports to the system in the future using Power BI.

3.4.3.7 Dashboards (DB-UI)

Dashboards will be comprised using a “Live Reports” concept providing the functionality of printing the output of the information as both a user interface screen and a report. Initially, the dashboards will include aggregated data from the R-ICMS Event Summary Report, the Decision Support System Report and the System Response Plan Performance Report. The following is a conceptual mockup of the dashboard interface:

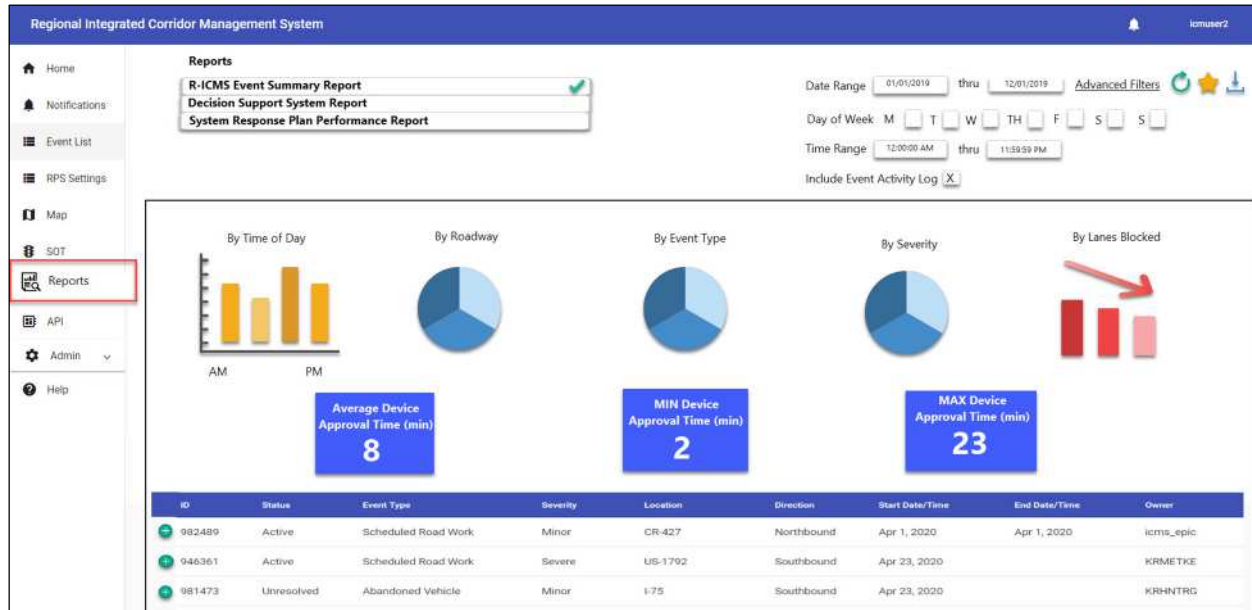


Figure 78 – Dashboard Display

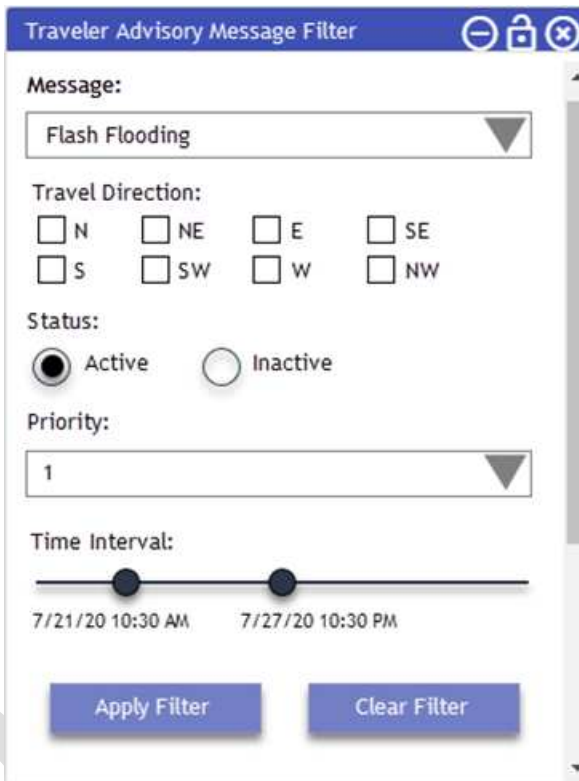
3.4.3.8 Traveler Advisory Messages (TAM-UI)

The TAM UI component will allow users to interactively visualize active TAMs and associated data from the map interface. Additionally, inactive TAMs will be available for review through user selection from the TAM data filter. Notable UI features will include:

- Toggle active TAM on/off
- View active TAM details via info window
- View active/inactive TAM record details via an info window that contains the message, start time, duration, and priority.
- Filter for a TAM record and zoom to the selected TAM feature on map
- Active TAM features will render in the map with visually distinctive symbology relative to the associated Presentation Region
- The Active TAM Presentation Region will be a polygon shape with a semi-transparent background and a slightly darker border. See the below diagrams.
- Each active TAM will render an eight-directional arrow (N, NE, E, SE, S, SW, W, NW) at the center of the feature – the affect will be that each of the directions which are currently active within the TAM/Presentation Region will be emboldened and darker to stand out. Any non-active direction arrow will be thinner and subtler, producing the affect that active directions are visually distinctive from non-active.
- Approach using Filter Widget:
 - The data table will be removed from the UI
 - A popup window will be available with TAM layer filter controls
 - Active vs Inactive

System Design Document for Regional Integrated Corridor Management System (R-ICMS)

- Start/End Times
- Direction in Affect
- Message



The screenshot shows a dialog box titled "Traveler Advisory Message Filter". It contains the following controls:

- Message:** A dropdown menu with "Flash Flooding" selected.
- Travel Direction:** A grid of checkboxes for N, NE, E, SE, S, SW, W, and NW, all of which are currently unchecked.
- Status:** Two radio buttons, "Active" (which is selected) and "Inactive".
- Priority:** A dropdown menu with "1" selected.
- Time Interval:** A horizontal slider with two black handles. Below the slider, the start and end times are displayed as "7/21/20 10:30 AM" and "7/27/20 10:30 PM".
- Buttons:** Two blue buttons at the bottom labeled "Apply Filter" and "Clear Filter".

Figure 79 – Tam Filtering

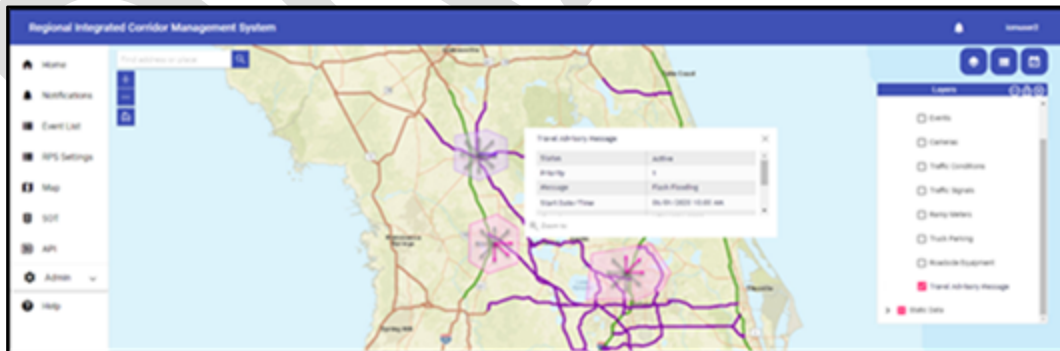


Figure 80 – TAM Display

Proposed TAMS Data/Workflow Process:

1. DFE to receive TAMS data updates
2. DFE to process TAMS data into defined schema, which will be readable by the GeoEvent Service
3. DFE to save TAMS data to shared folder, which will be accessible by the GeoEvent Service

4. GeoEvent Service will read/process TAMS data files into a GeoEvent Feature Service
5. The GeoEvent Feature Service will be added as a published service and included into the Map UI as a selectable Map Layer

3.4.3.9 DIVAS (DIV-UI)

The R-ICMS project team proposes to provide CCTV video through integration with FDOT's DIVAS. CCTV IDs from SunGuide will be combined with the associated DIVAS URL to provide CCTV video within the R-ICMS application. User access to CCTV video will be provided through the following R-ICMS components:

- SG Event details page
 - SG Event details area accessible from the map page
 - Camera info window from the map page
1. When the user clicks on a link to view a SunGuide camera video stream the UI will do the following: If not already authenticated, authenticate to the DIVAS VDS API
 - a. Request
 - i. URL: <https://DIVAS.cloud/VDS-API/oauth2/token>
 - ii. Headers
 1. Content-Type: application/x-www-form-urlencoded
 - iii. Form Data
 1. username=<username>
 2. password=<password>
 3. grant_type=password
 - b. Response
 - i. JSON

```
{
  "access_token": <token>,
  "token_type": "bearer",
  "expires_in": <timestamp>
}
```
 2. Retrieve the camera data for the SunGuide camera
 - a. Request
 - i. URL: <http://divas.cloud/VDS-API/VdsCameras/GetCamerasByNetworkIdAndSourceId/District%205/<SunGuide cameraId>>
 - ii. Headers
 1. Accept: application/json
 2. Authorization: Bearer <token>
 - b. Response
 - i. JSON

```
{  
  "cameraId": <cameraId (DIVAS)>,  
  "name": <name>,  
  "description": <description>,  
  "cameraType": <cameraType>,  
  "sourceId": <sourceId (SunGuide cameraId)>,  
  "center": <center>,  
  "agency": <agency (SunGuide centerId)>,  
  "county": <county>,  
  "highway": <highway>,  
  "locationName": <locationName>,  
  "latitude": <latitude>,  
  "longitude": <longitude>,  
  "direction": <direction>,  
  "mileMarker": <mileMarker>,  
  "videoRtspUri": <videoRtspUri>,  
  "videoHlsUri": <videoHlsUri>,  
  "snapshotAddress": <snapshotAddress>,  
  "isBlocked": <isBlocked>  
}
```

3. Extract the value of videoHlsUrl from the DIVAS camera response JSON.
4. Launch an HTTP Live Streaming (HLS) client to play the video stream for the HLS URL.

3.4.3.10 Response Plan Selection

The Response Plan Selection and Device Approval UI will provide the capabilities for the Corridor Managers to select/enact response plans and for the Device Managers to approve/deny their managed devices for a response plan.

3.4.3.10.1 *Diversion Route and Device Statuses*

The stage of an Evaluation is what influences the “Status” column in the Diversion Route and Device tables. The following list details all the possible stages of an Evaluation:

- Initial Pre-Simulation – The initial processing of the event and available response plans.
- Awaiting Simulation Results – Awaiting simulation results from the Simulation Engine.
- All Simulation Results Received – All simulation results have been received from the Simulation Engine.
- Awaiting Manager Selection – Available Response Plans have been sent to the Corridor Manager and is awaiting their selection.
- Manager Selection Received – A Manager has selected a Response Plan for Device Approval.
- Awaiting Device Approval – Awaiting device approval

- All Devices Responded – All devices of the Diversion Route have been responded to (approved or denied).
- Awaiting Manager Review – All devices have been responded to and is awaiting manager activation
- Selected Plan For Activation – A Manager has selected the plan for activation and a Response Plan Activation Request is sent to SunGuide.
- Rescheduled – The Evaluation was rescheduled
- Done – All processing has completed

The “Status” column of the Diversion Route table will reflect the stage of the Evaluation. As seen in Figure 82, the stage of the Evaluation is displayed in the title of the page, “Event 900405 – Awaiting Manager Selection”. Subsequently, the statuses of the Diversion Routes’ are “Awaiting Selection”. However, when a Diversion Route is selected for device approval, only the selected diversion route will display the status for the remainder Evaluation. This can be seen in Figure 83.

The “Status” column of the Device table is determined by the stage of the Evaluation and the approval status of the device. As seen in Figure 83, the stage of the Evaluation is “Awaiting Device Approval”. As a result, the statuses of the Devices are “Awaiting Approval”. When a Device Manager approves a device, the approved device’s status will change to “Approved” as seen in Figure 86. A device can also have a status of “Denied” when the device is denied and a status of “Overridden” when the device has not been approved but the Corridor Manager has activated the response plan.

3.4.3.10.2 Actions

The Response Plan Selection component provides functionality for both Corridor Managers and Device Managers (see Figure 85). Actions are enabled/disabled based on a user’s permissions and the stage of the Evaluation. The following table describes all the available actions of the Response Plan Selection Component:

Table 46 – Response Plan User Interface Actions

Button Name	Location	What it does	Enabled when
Approve All	Above Device Table	Approves all of the current user’s managed devices	Evaluation stage is “Awaiting Device Approval” and user has device permissions.
Deny All	Above Device Table	Denies all of the current user’s managed devices	Evaluation stage is “Awaiting Device Approval” and user has device permissions.
Approve (green check)	In the “Action” column of the device table.	Approves a device	Evaluation stage is “Awaiting Device Approval”, user has device permissions, and user is in the correct Device Group.
Deny (red X)	In the “Action” column of the device table.	Denies a device	Evaluation stage is “Awaiting Device Approval”, user has device permissions, and user

System Design Document for Regional Integrated Corridor Management System (R-ICMS)

			is in the correct Device Group.
View On Map	Above Device and Diversion Route table	Focuses the map on the selected object (diversion route or device)	A row in a table is selected.
Poll For Approval	Above Diversion Route Table	Selects a response plan for device approval	Evaluation stage is "Awaiting Manager Selection" or "Awaiting Device Approval" and users has response plan permissions.
Submit Response Plan	Above Diversion Route Table	Sends an activation request for the response plan.	Evaluation stage is "Awaiting Device Approval" or "Awaiting Manager Review" and users has response plan permissions.
View Event	Above Diversion Route Table	Navigates to the event details screen.	Users has view event permissions.

DRAFT

System Design Document for Regional Integrated Corridor Management System (R-ICMS)

ID	Event Status	Event Type	Severity	Location	Direction	Start Date/Time	End Date/Time	Owner	Diversion Route Status
900405	Active	Scheduled Road Work	Minor	I-75	Southbound	04/14/2020 15:30		some_user	Awaiting Manager Selection
900410	Active	Crash	Major	SR-46	Eastbound	04/14/2020 15:30		some_user	Awaiting Device Approval
900415	Not Active	Crash	Minor	I-75	Westbound	04/14/2020 15:30	04/14/2020 15:35	some_user	Activated
900417	Not Active	Crash	Minor	I-75	Eastbound	04/14/2020 15:30	04/14/2020 15:35	some_user	

Figure 81 – Event List with Diversion Route Status

Once a user clicks the “Diversion Route Details” button it will navigate you to Figure 82. This component provides functionality for both Corridor Managers and Device Managers. Actions are enabled/disabled based on a user’s role (Corridor Manager vs Device Manager) and the stage of the Response Plan.

The screenshot displays the 'Regional Integrated Corridor Management System' interface. At the top, the title bar reads 'Regional Integrated Corridor Management System' and 'CorridorManager1'. The main content area is titled 'Event 900405 - Awaiting Manager Selection'. On the left is a navigation menu with icons for Home, Notifications, Event List, Map, SOT, API, Admin, and Help. The central part of the interface features a map showing a road network with a red star marking the event location. To the right of the map is a table listing device messages:

Device Type	Name	Message/Plan ID	Status	Action
DMS	DeviceA	SOME_MESSAGE	Awaiting Route Selection	
DMS	DeviceB	SOME_MESSAGE	Awaiting Route Selection	
Signal	DeviceC	1	Awaiting Route Selection	
Signal	DeviceD	2	Awaiting Route Selection	
Signal	DeviceE	3	Awaiting Route Selection	

Below the map and message table is a table of diversion routes:

Diversion Route	Score	Model Results (TSD)	Status
10 IBA-L-to-98	1.8	10	Awaiting Selection
10 IBA-R-to-98	4.3	15	Awaiting Selection
10 IBA-R-to-99	3.5	20	Awaiting Selection
Do Nothing	11	30	Awaiting Selection

Buttons for 'Approve All', 'Deny All', and 'View On Map' are located above the message table. Buttons for 'Poll For Approval', 'Activate Response Plan', 'View Event', and 'View On Map' are located below the message table.

Figure 82 - Response Plan Selection: Awaiting Manager Selection

In Figure 82, a Corridor Manager can review the different Diversion Routes and select one by clicking the “Poll For Approval” button. This will transition the Response Plan into the “Awaiting Device Approval” stage as seen in Figure 83.

The screenshot displays the 'Regional Integrated Corridor Management System' interface. At the top, the title bar reads 'Regional Integrated Corridor Management System' and 'CorridorManager1'. The main content area is titled 'Event 900405 - Awaiting Device Approval'. On the left, a navigation menu includes Home, Notifications, Event List, Map, SOT, API, Admin, and Help. The central map shows a road network with a red star marking the event location. To the right of the map is a table with the following data:

Device Type	Name	Message/Plan ID	Status	Action
DMS	DeviceA	SOME_MESSAGE	Awaiting Approval	
DMS	DeviceB	SOME_MESSAGE	Awaiting Approval	
Signal	DeviceC	1	Awaiting Approval	
Signal	DeviceD	3	Awaiting Approval	
Signal	DeviceE	3	Awaiting Approval	

Below the map and table, there are buttons for 'Approve All', 'Deny All', and 'View On Map'. At the bottom of the interface, there is a 'Diversion Route' table and buttons for 'Poll For Approval', 'Activate Response Plan', 'View Event', and 'View On Map'.

Diversion Route	Score	Model Results (TBD)	Status
10 IBA-L-to-98	9.8	10	Awaiting Device Approval
10 IBA-R-to-98	4.3	15	
10 IBA-R-to-99	3.5	20	
Do Nothing	11	30	

Figure 83 - Response Plan Selection: Awaiting Device Approval

At this point, a Corridor Manager can await device approval, or they can submit the Response Plan and override device approval as seen in Figure 84.



Figure 84 - Response Plan Selection: Devices Overridden

If a Corridor Manager decides to not override the device approval, then they must await the Device Managers' approval. A Device Manager can approve or deny any of their managed devices using the actions available in the Device Table as seen in Figure 85.

The screenshot displays the R-ICMS interface for an event titled "Event 900405 - Awaiting Device Approval". The interface is divided into several sections:

- Navigation Panel (Left):** Includes icons for Home, Notifications, Event List, Map, SOT, API, Admin, and Help.
- Map (Center):** Shows a road network with a red star indicating the event location. Two "DMS" labels are visible on the map.
- Device Management Table (Top Right):** A table with columns: Device Type, Name, Message/Plan ID, Status, and Action.

Device Type	Name	Message/Plan ID	Status	Action
DMS	DeviceA	SOME_MESSAGE	Awaiting Approval	✓ ✗
DMS	DeviceB	SOME_MESSAGE	Awaiting Approval	✓ ✗
Signal	DeviceC	1	Awaiting Approval	
Signal	DeviceD	3	Awaiting Approval	
Signal	DeviceE	3	Awaiting Approval	✓ ✗
- Diversion Route Table (Bottom):** A table with columns: Diversion Route, Score, Model Results (TBD), and Status.

Diversion Route	Score	Model Results (TBD)	Status
10 IBA-L-10-98	4.8	10	Awaiting Device Approval
10 IBA-R-10-98	4.3	15	
10 IBA-R-10-99	3.5	20	
Do Nothing	11	30	

Figure 85 - Response Plan Selection: Awaiting Approval

A Device Manager can approve/deny individual devices using the buttons in the “Actions” column. The approve/deny buttons are only available for devices the Device Manager has access to. The Device Manager can also approve/deny all their managed devices by using the “Approve All” or “Deny All” buttons. When a Device Manager approves a device, it changes the Status to “Approved” as seen in Figure 86.



Figure 86 - Response Plan Selection: Devices Approved

Once all devices are approved, the Response Plan will transition into the “Awaiting Activation” stage as seen in Figure 87. The Corridor Manager will be notified that all devices have been approved and they will now be able to activate the Response Plan without overriding device approval.

The screenshot displays the R-ICMS interface for an event titled "Event 900405 - Awaiting Activation". The interface is divided into several sections:

- Header:** "Regional Integrated Corridor Management System" and "CorridorManager1".
- Left Navigation:** Home, Notifications, Event List, Map, SOT, API, Admin, Help.
- Map:** Shows a road network with a red star indicating the event location. Two "DMS" labels are visible on the map.
- Device Table:**

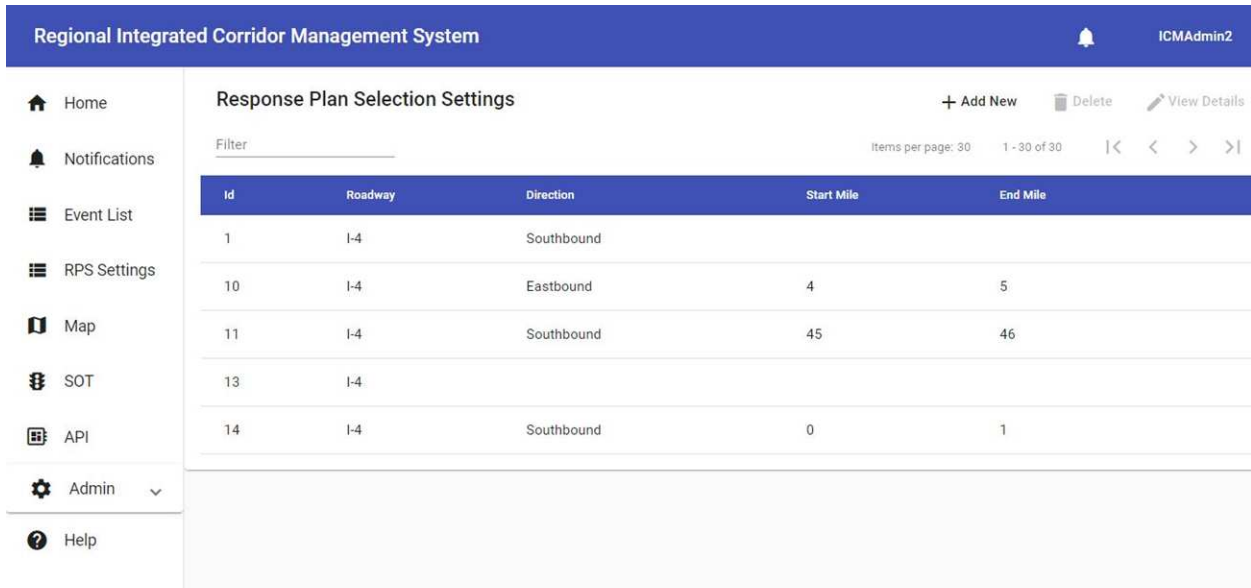
Device Type	Name	Message/Plan Id	Status	Action
DMS	DeviceA	SOME_MESSAGE	Approved	
DMS	DeviceB	SOME_MESSAGE	Approved	
Signal	DeviceC	1	Approved	
Signal	DeviceD	2	Approved	
Signal	DeviceE	3	Approved	
- Buttons:** "Approve All", "Deny All", "View On Map", "Pull For Approval", "Activate Response Plan", "View Event", "View On Map".
- Diversion Route Table:**

Diversion Route	Score	Model Results (TBD)	Status
10 IBA-L-10-98	9.8	10	Awaiting Activation
10 IBA-R-10-98	4.3	15	
10 IBA-R-10-99	3.5	20	
Do Nothing	11	30	

Figure 87 - Response Plan Selection: Awaiting Activation

3.4.3.10.3 Response Plan Selection Settings

The Response Plan Selection Settings UI will provide the functionality for configuring Response Plan selection settings for a roadway section. The Response Plan Selection Settings List displays all the existing settings and allows a user to add, delete, or modify a Selection Setting.



Regional Integrated Corridor Management System

ICMAdmin2

Response Plan Selection Settings

+ Add New Delete View Details

Filter

Items per page: 30 1 - 30 of 30

Id	Roadway	Direction	Start Mile	End Mile
1	I-4	Southbound		
10	I-4	Eastbound	4	5
11	I-4	Southbound	45	46
13	I-4			
14	I-4	Southbound	0	1

Figure 88 – Response Plan Selection: Settings List

RPS Settings

Applicability

Roadway: I-4 Direction: Southbound

Start Mile: 45 End Mile: 46

Queue Thresholds

Congestion Speed (MPH): 5 Congestion Speed Relative to Historical Norm: 0 Uncongested Buffer (Miles): % 0

Filters

Critical Signals

Percent Required: 0 Minimum Number Required: % 5

DMS

Percent Required: 0 Minimum Number Required: % 0

Maximum Volume Over Capacity

Minor Events: 0 Intermediate Events: % 0 Major Events: % 0

Benefit Threshold

Initial: 0 Recurring: 0

Cancel Save

Figure 89 – Response Plan Selection: Setting Creation/Modification

The following table describes the editable fields that are displayed in **Figure 89**.

Table 47 – Response Plan Selection Applicability Fields

Field Name	Section	Description
Roadway	Applicability	The roadway the setting applies to.
Direction	Applicability	The direction of the roadway the setting applies to.
Start Mile	Applicability	The start mile of the roadway the setting applies to.
End Mile	Applicability	The end mile of the roadway the setting applies to.
Congestion Speed (MPH)	Queue Thresholds	The speed of congestion.

System Design Document for Regional Integrated Corridor Management System (R-ICMS)

Congestion Speed Relative to Historical Norm	Queue Thresholds	The current speed compared to the historical norm for determining congestion.
Uncongested Buffer (Miles)	Queue Thresholds	The length of uncongested links to determine the end of the congestion queue.
Percent Required	Filters – Critical Signals/DMS	The percent of devices required for a response plan to be considered.
Minimum Number Required	Filters – Critical Signals/DMS	The minimum number of devices required for a response plan to be considered.
Minor Events	Filters - Maximum Volume Over Capacity	The maximum percent volume over capacity threshold for severity 1 events.
Intermediate Events	Filters - Maximum Volume Over Capacity	The maximum percent volume over capacity threshold for severity 2 events.
Major Events	Filters - Maximum Volume Over Capacity	The maximum percent volume over capacity threshold for severity 3 events.
Initial	Filters – Benefit Threshold	The value of simulated response plan score must meet or exceed during the initial evaluation.
Recurring	Filters – Benefit Threshold	The value of simulated response plan score must meet or exceed after the initial evaluation.
Max Simulations	Filters - Miscellaneous	The max number of simulations the Simulation Engine will run per response plan.
Max Suggested Plans	Filters - Miscellaneous	The max number of suggested plans.
No Suggested Plan	Reevaluation Delays (Minutes)	The reevaluation time in minutes when no plan was selected.
Suggested But Not Activated Plan	Reevaluation Delays (Minutes)	The reevaluation time in minutes when a plan was selected but not activated.
Activated Plan	Reevaluation Delays (Minutes)	The reevaluation time in minutes when a plan was activated.

3.4.3.11 Signal Optimization Timing (SOT-UI)

Accessible from the main UI, the SOT-UI provides screens and functionality necessary to optimize traffic signal coordination timing plans. The SOT-UI allows users to:

- Build and manage signal corridors with scheduled coordinated timing plans
- Use existing signal timing plans, schedules, lane geometry, restrictions, and traffic demand data
- Create ideal timing plan schedules by clustering traffic demand data
- Optimize signal corridor timing plan cycle lengths, splits, offsets, lead/lag and flashing-yellow-arrow phasing
- Simulate new versus existing signal corridor timing plans
- Fine-tune splits and offsets, including re-simulation
- Request and manage approval of new signal corridor timing plans
- Export new signal corridor timing plans and reports

Signal corridors can also consist of a single intersection, which is optimized individually.

Signal corridors are integrated with the map UI, so that managed signal corridors may be viewed on the map.

3.4.3.11.1 Corridor Configuration

Management of SOT corridors is accomplished through a group of screens accessible from the main menu. Figure 90 shows the SOT list of corridors configured in the system. Corridor details may be viewed by clicking a row in the list. Corridor names are automatically generated, based on the roadway, direction, starting and ending intersections. Corridor ID, activation times of day and week, status, creation time and owner may also be used to uniquely identify a configuration. Most fields in the list are sortable and some can be filtered by user-entered text or categorical choices, such as a selection of “Signal Agencies”. Additionally, users may filter for corridors containing a specific traffic signal.

The following functionality is accessible from the SOT list:

- “+ Add” button allows the user to create a new corridor
- “View Details” button loads the selected corridor details view
- “View on Map” button loads a split view of the SOT list and the map zoomed in on the selected corridor
- “Clone” button makes a copy of the selected corridor
- “Archive” button sets the corridor status to “Archived”, which is filtered out of view by default
- “Help” button loads the SOT user manual

System Design Document for Regional Integrated Corridor Management System (R-ICMS)

Id	Corridor	SR-426	Signal	Start	End	Applicability	Overall Delay	Auto-Clustered	Status	Last Modified	Created By	Signal Agencies
1	SR-426: Dean Rd to Old	US-17-92	1770: Howell Branch Rd / Hall Rd	08:00	18:00	60%		No	Configuring	4/6/20 10:05 PM	KWRIDL	ORL
2	SR-426: Dean Rd to Old	Corridor 3	1775: Trinity Prep Ln	06:00	20:00	55%		No	Optimization Success	4/5/20 10:05 PM	MARTIP23	ORL
3	SR-426: Dean Rd to Old Howell Br Rd		1780: Tuskawilla Rd	Sa/Su	00:00	24:00	100%	Yes	Simulating	4/4/20 10:05 PM	BVILLA	ORL
4	US-17-92: E Marks St to Virginia Dr		1785: Clayton Crossing Way	08:00	18:00	60%	7mi ▶ 3.3min	Yes	Simulation Success	4/3/20 10:05 PM	KWORL32	ORL
5	US-17-92: E Marks St to Virginia Dr		1790: SR-417 SB ramp	08:00	15:00	55%	9mi ▶ 7.2min	No	Awaiting Approval	4/2/20 10:05 PM	ICMAdmin1	SEM/ORA/ORL
6	SR-426: Dean Rd to Old Howell Br Rd		1795: SR-417 NB ramp	10:00	18:00	60%	14mi ▶ 13.2min	No	Approved	4/1/20 10:05 PM	MasterUser2	SEM/ORA/ORL
7	US-17-92: E Marks St to Virginia Dr		1880: Dean Rd / Chaddsford Cir	08:00	20:00	70%	4.4mi ▶ 4.6min	No	Denied	3/30/20 10:05 PM	SEMCTYUSR	SEM/ORA/ORL
8	US-17-92: E Alexander Hamilton Ln to George Washington St		M/T/W/Th/F	00:00	18:00	60%	18mi ▶ 12min	Yes	Deployed	3/29/20 10:05 PM	KWMS22FD	SEM/ORA/ORL
9	US-17-92: E Marks St to Virginia Dr		M/T/W/Th/F	00:00	18:00	60%	12mi ▶ 11.2min	Yes	Retired	3/29/20 10:05 PM	FDOTUSR1	SEM/ORA/ORL

Figure 90 – SOT corridor list

SOT corridors may have the following statuses:

- Configuring, corridor configuration in progress
- Optimizing, optimization in progress (edits not allowed)
- Optimization Converged, optimization completed by converging
- Optimization Not Converged, optimization completed without converging
- Optimization Error, optimization completed with error
- Simulating, simulation in progress (edits not allowed)
- Simulation Completed, simulation completed successfully
- Simulation Error, simulation completed with error
- Modifying, split & offset modifications in progress
- Awaiting Approval, approval requests are pending
- Approved, all signals timing plans approved
- Denied, any signal timing plan denied
- Deployed, marked as deployed (deployment is a manual offline process)
- Retired, marked as retired (retirement is a manual offline process)
- Archived, marked as archived (filtered out of view by default)

If a SOT corridors optimization and simulation have completed, additional data is shown, such as “Overall Delay” and the status of corridor approval requests in the “Signal Agencies” column.

Color codes for approval status are:

- black/normal = corridor approval has not been requested
- green/italic = all signals for an agency were approved
- red/bold = any signal for an agency was denied

- grey/underline = request pending

3.4.3.11.2 Optimization Configuration & Results

Figure 91 shows the new corridor configuration screen, which is divided into a set of sequential steps the user must complete in order. The figure depicts the “Step 1: Corridor” view, where the user may select a roadway followed by a set of any contiguous signals.

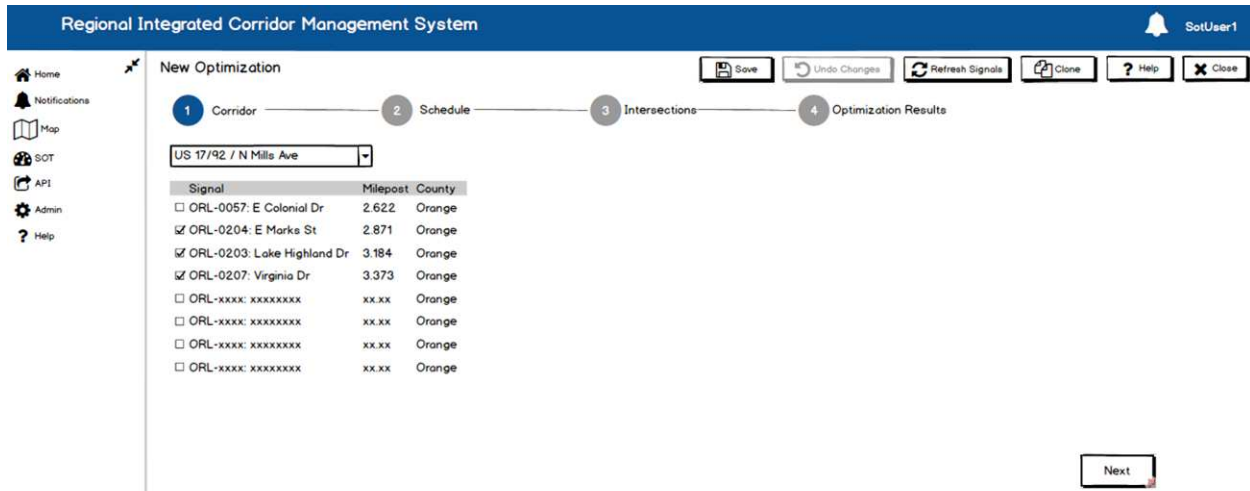


Figure 91 – SOT Step 1: Corridor

Figure 92 depicts the “Step 2: Schedule” view, where the user may set corridor activation days and times, historical traffic volume date range, periodic/recurring optimization settings, and the timing pattern schedule. In this view, the user can click the “Auto-calculate Schedule” button, which runs a background process to cluster traffic demand and generate the initial “For Optimization” timing plan schedule. Alternatively, the user can view the existing timing pattern schedule for any signal and click a button to “Use this Schedule”. When the button is clicked, a copy of the schedule is made loaded into the “For Optimization” schedule tab, as shown in Figure 93. Users can also create a new schedule manually using the “Add/Edit Time Cluster” button. On the “For Optimization” tab, the user sets optimization settings for each time cluster by clicking on the associated cog icon. Configurable settings are depicted in Figure 93.

System Design Document for Regional Integrated Corridor Management System (R-ICMS)

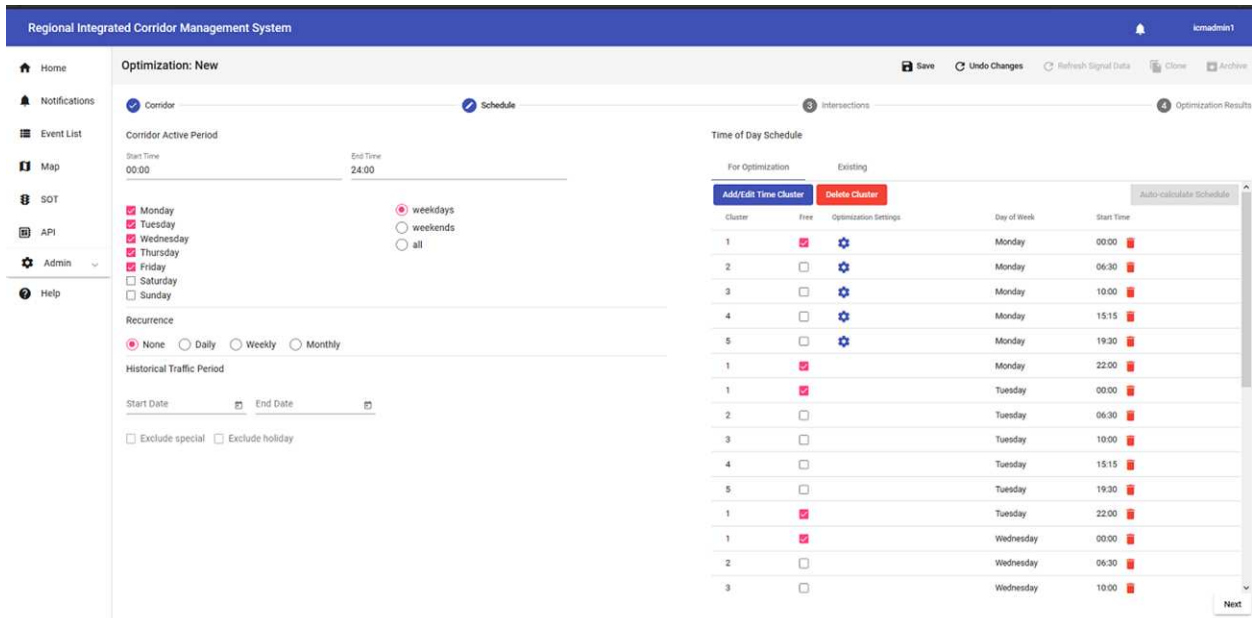


Figure 92 – SOT Step 2: Schedule

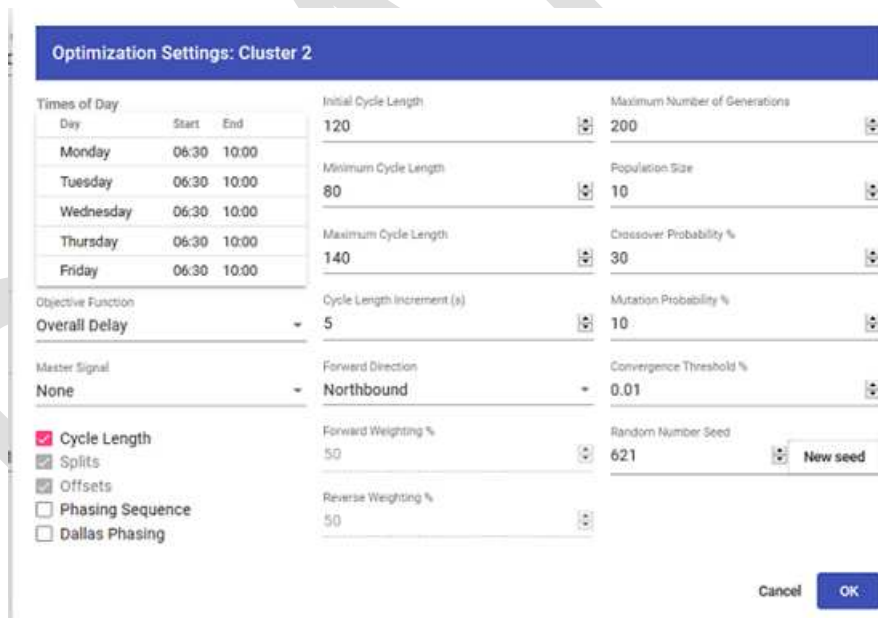


Figure 93 - SOT Step 2: Schedule – Cluster Settings

Figure 94 depicts the “Step 3: Intersections” view, where a user may cycle through intersections to view lane and phasing configuration and basic timing parameters. The user may also cycle through each analysis time-cluster per signal to view and set inputs for signal coordination optimization. These settings are loaded with real data from signal controllers or default values set by the SOT services. Users may edit the following fields: minimum green, passage time,

System Design Document for Regional Integrated Corridor Management System (R-ICMS)

exclusive pedestrian time, initial movement splits, lead/lag phasing, recall mode, offset, reference phase, and reference point.

Optimization: New

Corridor: ORL-0203: E Marks St, ORL-0202: Lake Highland Dr, ORL-0207: Virginia Dr

Movement	Phase	Yellow Change (s)	Red Clear (s)	Min Green (s)	Passage (s)	Ped. Walk (s)	Ped. Clear (s)	Dual Entry
EBL				5	3	7	20	<input checked="" type="checkbox"/>
EBT	8	3.5	3					<input type="checkbox"/>
WBL				5	3	7	20	<input checked="" type="checkbox"/>
WBT	4	3.5	3					<input type="checkbox"/>
NBL				6	3	7	10	<input checked="" type="checkbox"/>
NBT	6	4.2	2					<input type="checkbox"/>
SBL				6	3	7	13	<input checked="" type="checkbox"/>
SBT	2	4.2	2					<input type="checkbox"/>

Force Off: Fixed

Dallas Phasing: E/W, N/S, E/W, N/S

Time Cluster 1 (FREE): EBL, EBT, WBL, WBT, NBL, NBT, SBL, SBT

Cycle Length: 120, Offset: 0, Reference Phase: Begin Green

Times of Day:

Day	Start	End
Monday	06:30	10:00
Tuesday	06:30	10:00
Wednesday	06:30	10:00
Thursday	06:30	10:00
Friday	06:30	10:00

Figure 94 – SOT Step 3: Intersections

Clicking on the lane diagram in the figure above, yields a dialog box showing additional intersection geometry data, as shown in Figure 95.

Lane Configuration Details

Roadway: US-17/92 / Mills Ave
Intersection: ORL-0203: E Marks St

SB Approach Speed limit: 35 Grade: 0%

	Width(ft)	Storage(ft)	Detector(ft)
SBL	12	145	40
SBT	12	0	40
SBR			
SBU			

EB Approach Speed limit: 25 Grade: 0%

	Width(ft)	Storage(ft)	Detector(ft)
EBL	12	95	40
EBT	12	0	40
EBR			
EBU			

WB Approach Speed limit: 25 Grade: 0%

	Width(ft)	Storage(ft)	Detector(ft)
WBL			
WBT	12	0	40
WBR			
WBU			

NB Approach Speed limit: 35 Grade: 0%

	Width(ft)	Storage(ft)	Detector(ft)
NBL	12	95	40
NBT	12	0	40
NBR			
NBU			

OK

Figure 95 – SOT Step 3: Intersections Geometry Details

Figure 96 depicts read-only view of overlap phases, which are loaded from the SunGuide™ ATMS traffic signal subsystem Traffic Management Data Dictionary (TMDD) feed and accessed by clicking the “Overlap Phases” icon in step 3. The SOT application will not use overlaps in the optimization or simulation process.

Overlap #	Phases Included
1	2:9

Figure 96 – SOT Step 3: Intersections Overlap Phases

Figure 97 depicts the read-only view of time-of-day restrictions for an intersection as shown by clicking on the “Restrictions” icon in Step 3.

Approach Restrictions				
Roadway: US-17/92				
Intersection: ORL-0207: Virginia Dr				
Direction	Day of Week	Start Time	End Time	Restriction
SB	M/T/W/Th/F	11:00	15:00	Can't lag left
EB	M/T/W/Th/F/Sa/Su	00:00	24:00	Split phase side street
WB	M/T/W/Th/F/Sa/Su	00:00	24:00	Split phasing required
SB	M/T/W/Th/F/Sa/Su	00:00	24:00	Can't Run Concurrent Lefts
NB	M/T/W/Th/F/Sa/Su	00:00	24:00	Can't Run Concurrent Lefts
SB	M/T/W/Th/F/Sa/Su	00:00	24:00	Dallas phasing disallowed
NB	M/T/W/Th/F/Sa/Su	00:00	24:00	Dallas phasing disallowed

OK

Figure 97 – SOT Step 3: Intersections (D – restriction)

Impacts of intersection restrictions on the SOT application are described below in Table 48. Restrictions are loaded from the Signalized Intersection Inventory Application (SIIA) where they are provided per-lane, however SOT will apply these restrictions per-approach, and those marked with ** will be also be applied to opposing approaches.

Table 48 - SOT Intersection Restrictions

Approach Restriction	Impact on SOT UI	Notes
"Can't Lag Left"	Uncheck and disable lag left for the approach	
"Can't Overlap Right"	None	Phasing is read-only and loaded from SIIA. If future SOT development allows edits to movements, then additional validation logic would be required.
**"Can't Run Concurrent Lefts"	Ensure phase 1 or 5 are lagged, user can select which one, but	

System Design Document for Regional Integrated Corridor Management System (R-ICMS)

	form validation will fail if neither box is checked	
**"Dallas phasing disallowed"	Uncheck and disable all Dallas phasing	
"Exclusive phases for pedestrians"	Require exclusive pedestrian time to be entered and > 0	FDOT will provide a lower limit on the number of seconds to be required if this restriction exists. This input reserves cycle time where vehicle phases are all red, except potentially an overlap, such as a protected right turn.
"No Right Turn On Red"	None	Not an input for HCS7, but traffic volume data should reflect this.
"Permitted-L disallowed"	None	Phasing is read-only and loaded from SIIA. If future SOT development allows edits to movements, then additional validation logic would be required.
"Protected-L disallowed"	None	Phasing is read-only and loaded from SIIA. If future SOT development allows edits to movements, then additional validation logic would be required.
"Restriction"	None	
"Right Turn on Red (RTOR) only right most turn bay"	None	Not an input for HCS7, but traffic volume data should reflect this.
**"Simultaneous gap disallowed"	Uncheck and disable simultaneous gap for the approach and opposing direction	
**"Split phase side street"	Check and disable split phasing	
"U-turn disallowed"	None	Not an input for HCS7, but traffic volume data should reflect this.

Figure 98 depicts the read-only view of traffic demand for an intersection and time cluster as shown by clicking on the "Traffic Volume" icon in Step 3. Historical traffic demand data is queried for the time period configured in step 2 and filtered to include the time periods

belonging to a cluster. Traffic volume is shown for the peak (80th percentile) 15-minute interval within the time-cluster and the off-peak, or average of below-peak time intervals.

The values on the traffic volume screen are used by the signal optimization back end (HCS7) to adjust expected intersection saturation flow rates and capacity:

Table 49 – SOT Intersection Traffic Volume Defaults

Factor Name	Description	Data Source	Default Value
Sat Flow Rate	Base Saturation Flow Rate	Default	1900 for Through 1700 for Left 1500 for Right
Start Up Lost	The amount of time it takes for vehicle to react to a green light and for traffic to begin flowing, the default is 2 seconds		2 Seconds
Extension of Green	The amount of yellow time at the end of a green phase which drivers continue to use		2 Seconds
Heaviest Lane	for movements with multiple lanes this is the highest vehicle count in a single lane	ITSIQA	0
% Heavy Vehicles	Percentage of Heavy Vehciles	ITSIQA	0
Right on Red/Hr	How many right turns on red happen per hour	ITSIQA	0
Pedestrians/Hr	How many pedestrian crossings per hour	ITSIQA	0
Bike/Hr	How many bikes per hour	ITSIQA	0
Bus/Hr	How many buses per hour	ITSIQA	0
% Arrival on Green	Percentage of vehicles arriving at the intersection to green lights.	ITSIQA	0
Average Queue	Average queue length	Unknown	0

Traffic Volume													
Roadway:	US-17/92 / Mills Ave		Date Range:	5/4/20 to 5/22/20									
Intersection:	US-17/92 : E Marks St		Time Cluster:	2									
			Period:	15-min									
	Peak	Off-Peak											
Movement	Vehicles/Hr	Heaviest Lane	% Shared Lane	% Heavy Vehicle	Right on Red/Hr	Pedestrian/Hr	Bike/Hr	Bus/Hr	Sat Flow Rate	Start Up Lost	Extension Of Green	% Arrive On Green	Average Queue
EBU	0	0		0%		0	0	0	1500	2	2	0%	
EBL	0	0	0%	0%		0	0	0	1700	2	2	0%	
EBR	0	0	0%	0%	0	0	0	0	1500	2	2	0%	
EBT	0	0		0%		0	0	0	1900	2	2	0%	
WBU	0	0		0%		0	0	0	1500	2	2	0%	
WBL	0	0	0%	0%		0	0	0	1700	2	2	0%	
WBR	0	0	0%	0%	0	0	0	0	1500	2	2	0%	
WBT	0	0		0%		0	0	0	1900	2	2	0%	
NBU	0	0		0%		0	0	0	1500	2	2	0%	
NBL	0	0	0%	0%		0	0	0	1700	2	2	0%	
NBR	0	0	0%	0%	0	0	0	0	1500	2	2	0%	
NBT	0	0		0%		0	0	0	1900	2	2	0%	
SBU	0	0		0%		0	0	0	1500	2	2	0%	
SBL	0	0	0%	0%		0	0	0	1700	2	2	0%	
SBR	0	0	0%	0%	0	0	0	0	1500	2	2	0%	
SBT	0	0		0%		0	0	0	1900	2	2	0%	

OK

Figure 98 – SOT Step 3: Intersection (E - traffic volume)

Once the intersection and time-cluster settings have been finalized, users can click on “Step 4: Optimization Results” and click the “Run” button to initiate a set of HCS7 optimizations via the RICMS function-as-a-service (FaaS) portal. The run button always saves the configuration before requesting the optimization, and the user can save a configuration at any time after completing step 1 via the toolbar “save” button. When an optimization is saved, it will be re-displayed with an optimization ID and an automatically generated name. When the run button is clicked, the

user will be redirected to the corridor list view where the optimization will be shown with a status of “Running”.

When a SOT corridor optimization and initial simulations complete, authorized users receive a notification by popup or email. The notification links directly the SOT “Step 4: Optimization Results” screen which shows the result details.

The figures listed below show the SOT result details:

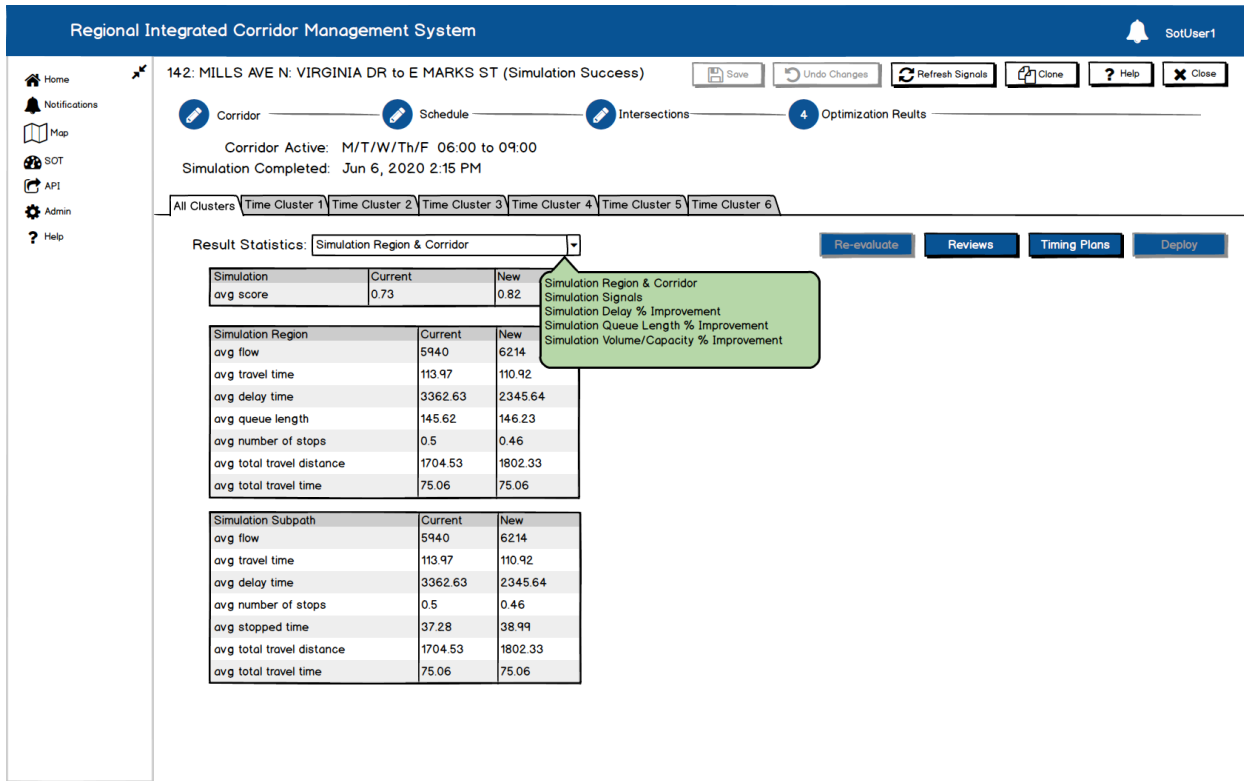


Figure 99 - SOT All Clusters – Simulation Region & Corridor statistics

Figure 99 – The "All Clusters" tab shows a variety of simulation statistics, selectable by a drop-down box. A button-toolbar allows users to request and view timing plans, approvals, and signatures, export data, and mark corridors as deployed or retired. The “Simulation Region & Corridor” statistics are averaged for all simulations within the corridor active period.

System Design Document for Regional Integrated Corridor Management System (R-ICMS)

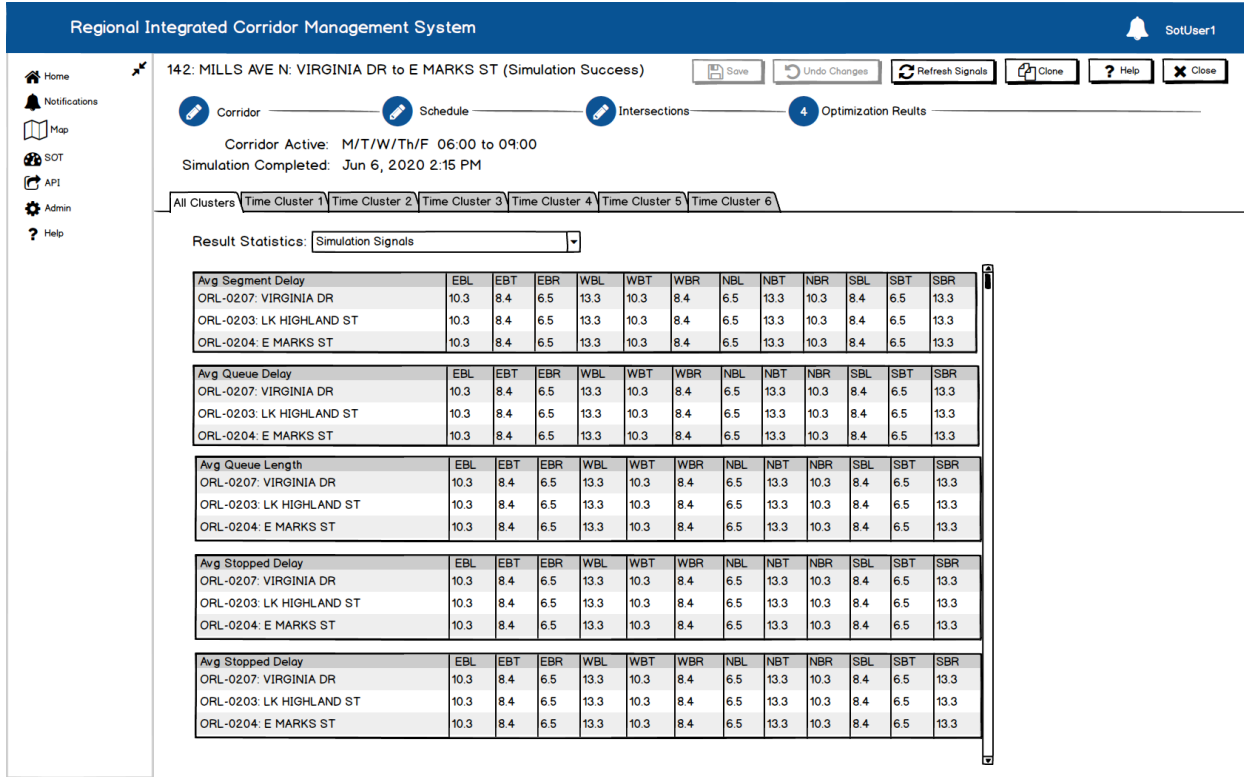


Figure 100 - SOT All Clusters – Simulation Signals statistics

Figure 100 – The "All Clusters" tab shows the "Simulation Signals" statistics averaged for all simulations within the corridor active period.



Figure 101 - SOT All Clusters – Simulation Delay % Improvement heatmap

Figure 101 – The "All Clusters" tab shows the "Simulation Delay % Improvement" statistics averaged for all simulations within the corridor active period. This is shown as a heat map of the percent difference between values for new and existing timing plans per-intersection and per-approach over time. Additional heat maps not pictured include "Simulation Queue Length % Improvement" and "Simulation Volume/Capacity % Improvement". Time gaps in the corridor activation period, from one day to the next, are represented as a small amount of whitespace. Mouse-over tooltips show the intersection/approach, percent change, day and time period, and time cluster assignment.

System Design Document for Regional Integrated Corridor Management System (R-ICMS)

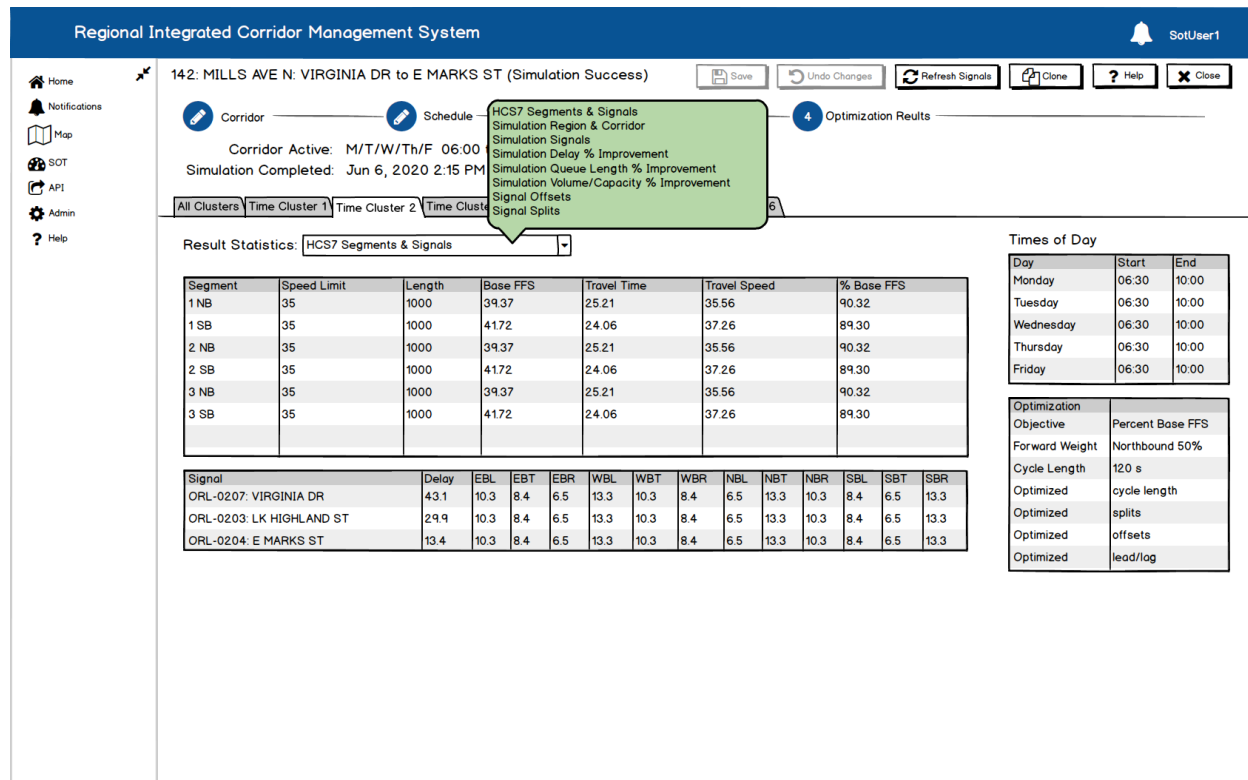


Figure 102 - SOT Time Cluster Tabs– HCS7 Segments & Signal statistics

Figure 102 – The "Time Cluster 2" tab shows a variety of HCS7 and simulation statistics, selectable by a drop-down box. These statistics are supplied by a single HCS7 optimization and Aimsun simulation that represents time periods in a cluster with similar traffic demand. "Times of Day" shows the time periods assigned to the cluster, below which the optimization objectives and cycle length are shown. "HCS7 Segments & Signals" shows results from the optimization, and while not pictured, the simulation statistics provided on the "All Clusters" tab are also available for the time cluster.

System Design Document for Regional Integrated Corridor Management System (R-ICMS)

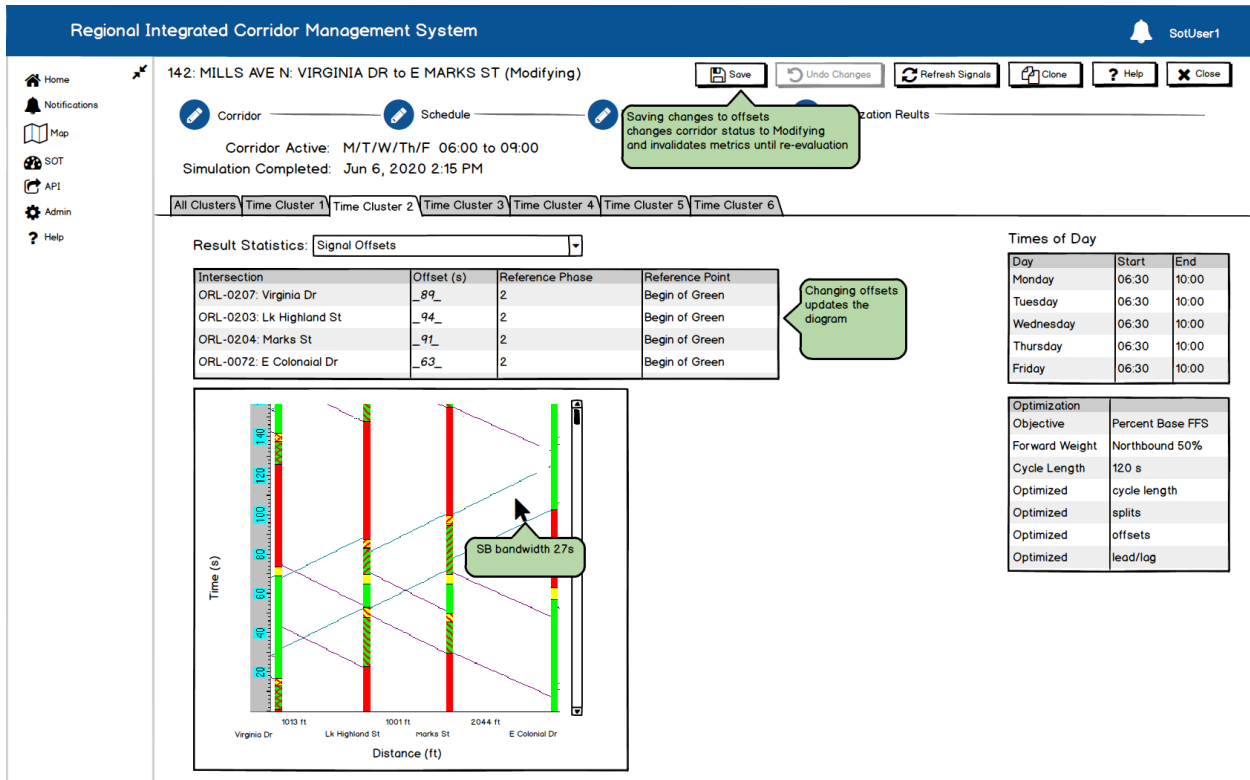


Figure 103 – SOT Time Cluster 2 – Signal Offsets

Figure 103 – The "Time Cluster 2" tab shows "Signal Offsets" as a time-space diagram with the expected progression bandwidth of vehicles moving through all intersections. Two cycles will be shown, along with progression bands in both main street directions. Offsets in the table are editable, which causes the time-space diagram to be redrawn.

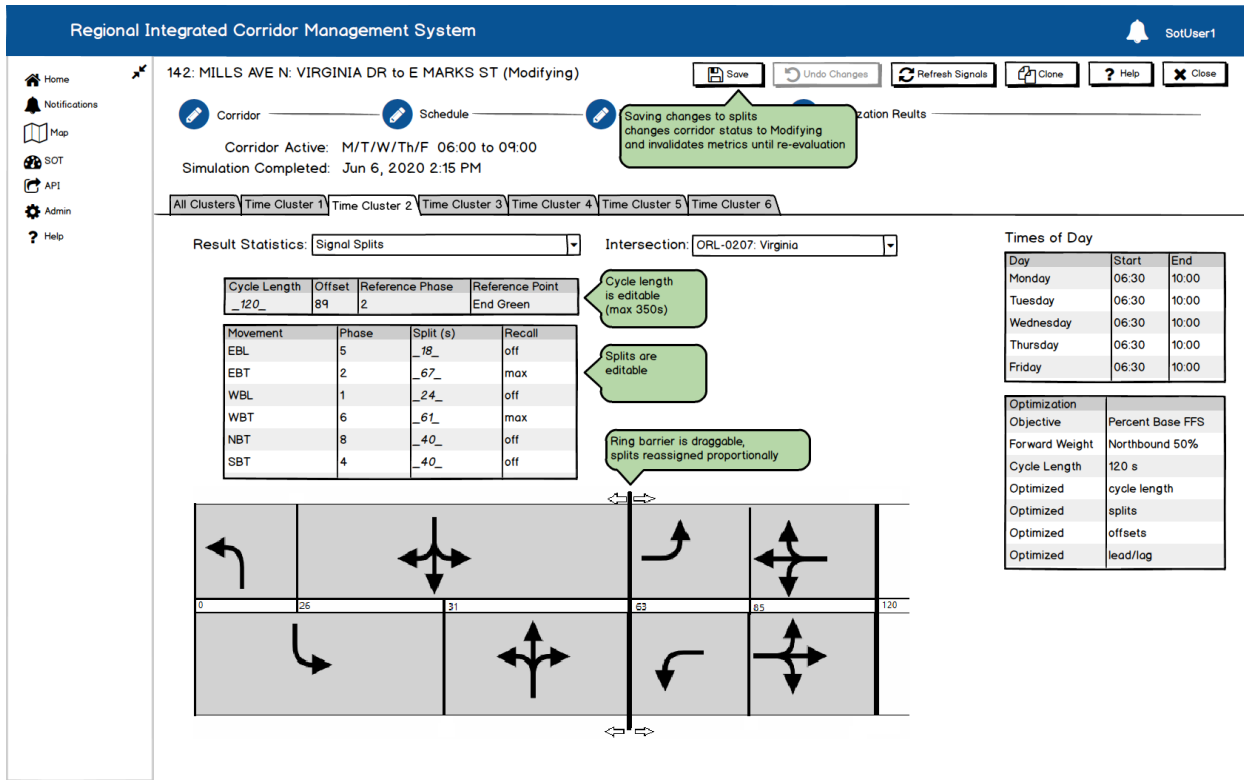


Figure 104 - SOT Time Cluster Tabs – Signal Splits

Figure 104 – The "Time Cluster 2" tab shows "Signal Splits" as a phasing diagram. Users can edit cycle length or individual phase splits. Main street through phases, typically 2 and 6, are not editable, any unallocated green time is assigned to these phases automatically. The diagram is redrawn if values are edited. Users can also drag the ring-barrier bar to reallocate splits proportionally.

- Modifications to offsets and splits may be reverted using the "Undo Changes" button or saved using the "Save" button. If modifications are saved:
 - The corridor status is changed to "Modifying"
 - Any review requests, whether pending or completed, are canceled
 - SOT simulation result details are be blanked out until the user runs a re-evaluate

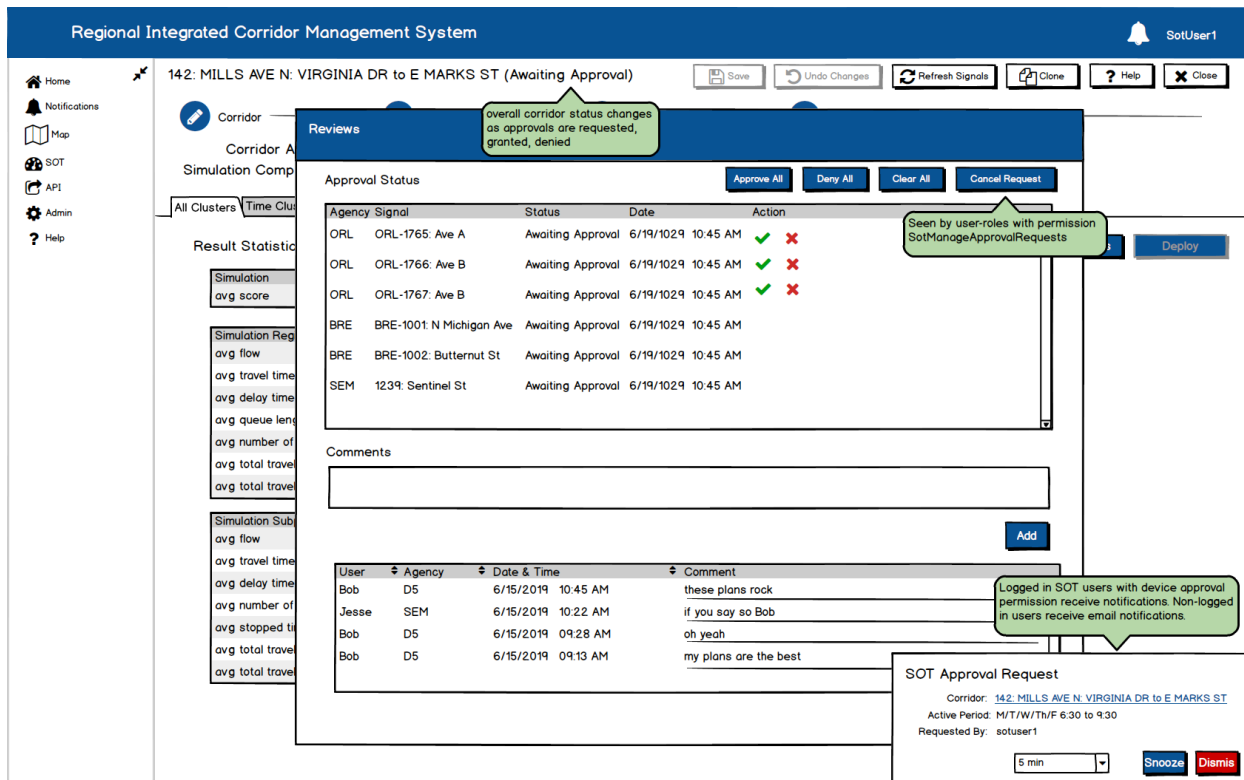


Figure 105 - SOT All Clusters – Reviews & Approvals

Figure 105 – The "All Clusters" tab button "Reviews" launches a dialog in which users can request approval for signal timing plans. Users with device approval permissions receive alerts by popup or email, which are automatically resolved when all devices are approved or denied, or the approval request is canceled. Users can also add comments to the timing plan set. Clicking an Action button (Approve, Deny, etc.) updates the Status and Date, and these changes to status and comments are saved when a user clicks OK.

System Design Document for Regional Integrated Corridor Management System (R-ICMS)

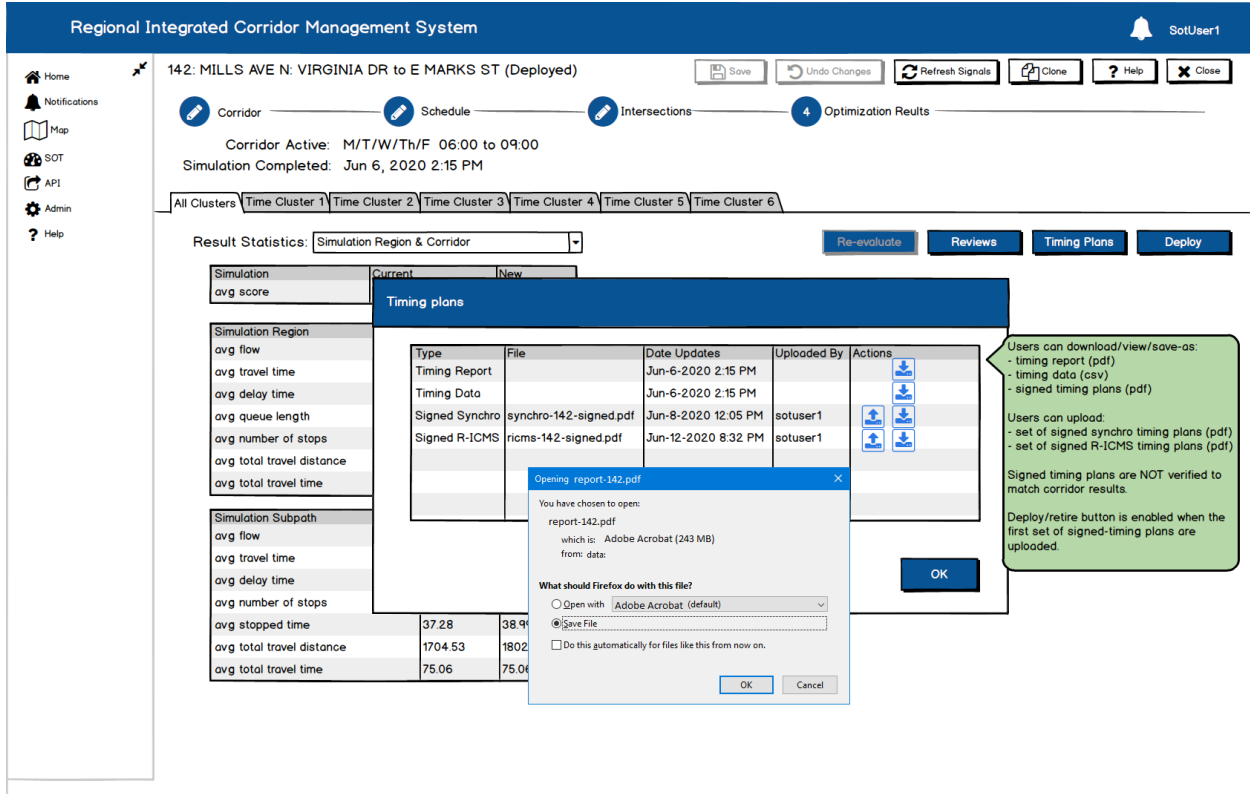


Figure 106 - SOT All Clusters – Timing Plans

Figure 106 – The "All Clusters" tab button "Timing Plans" launches a dialog which allows users to download reports, data, or signed plans, as well as upload signed plans from Synchro or R-ICMS. Signed timing plans are NOT verified to match corridor results. The "Deploy" button is enabled when the first set of signed-timing plans are uploaded.

System Design Document for Regional Integrated Corridor Management System (R-ICMS)

Regional Integrated Corridor Management System

142: MILLS AVE N: VIRGINIA DR to E MARKS ST (Simulation Success)

Corridor | Schedule | Intersections | Optimization Results

Corridor Active: M/T/W/Th/F 06:00 to 09:00
Simulation Completed: Jun 6, 2020 2:15 PM

Corridor Conflicts

US-17/92: SR-50 to Virginia Dr
Active: M/T/W/Th/F/Sa/Su (00:00 - 24:00)

The deployed corridors listed below must be retired or cross-coordinated to deploy this configuration.

ID	Roadway	From	To	Days of Week	Start Time	End Time	Intersection	Can Cross-Coordinate
1	Marks St	Magnolia Ave	N Fern Creek Ave	M/T/W/Th/F	06:00	09:30	Marks St	Y
4	SR-50	Highland Ave	Maguire Blvd	M/T/W/Th/F/Sa/Su	00:00	24:00	US-17/92	Y
8	US-17-92	Marks St	Virginia Dr	M/T/W/Th/F	00:00	24:00	Marks St	
8	US-17-92	Marks St	Virginia Dr	M/T/W/Th/F	00:00	24:00	Lake Highland Dr	
8	US-17-92	Marks St	Virginia Dr	M/T/W/Th/F	00:00	24:00	Virginia Dr	

Seen by user-roles with permission SotManageCorridors

OK

Figure 107 - SOT All Clusters – Deploy Corridor

Figure 107 – The "All Clusters" tab button "Deploy" marks a corridor as deployed. If signals are in other deployed corridors with overlapping days and times, these conflicts are displayed, and deployment is blocked until all are resolved. To resolve conflicts, other corridors must be retired by using the "Retire" button in place of the deploy button.

The general workflow for SOT corridor management is:

1. Configure signal corridor input
2. Optimize timing plans and simulate new versus existing
3. Receive alert email or popup to review results
4. Optionally modify offsets and/or splits and re-evaluate
5. Download timing reports and data for review in Synchro and deployment to signal controllers
6. Request timing plan approval
7. Monitor approvals and comments
8. Deploy timing plans to signal controllers in-the-field
9. Tweak timing plans in-the-field and review ATSPM measures before/after deployment
10. Update the SOT corridor timing plans to match the in-the-field
11. Download timing report to sign and seal
12. Upload signed sealed timing report
13. Deploy corridor, optionally retire conflicting corridors
14. Retired corridors and those which are not approved or deployed may be “Archived”

Figure 108 shows a popup alert received by SOT users after an optimization and simulation have completed. If the corridor is deployed, all logged-in users with permission to edit corridors receive this alert. If the corridor is not deployed, only the user who created the corridor configuration receives the alert. Emailed alerts are issued to offline users. Agency users with permission to approve/reject signal timing receive similar alerts after a user with corridor edit privileges requests agency review of a corridor configuration.

Clicking the “Results” link dismisses the popup and redirects the user to the corridor results view. This alert looks the same for deployed and test corridors, and whether it was sent to one or more users across one or more agencies. This alert looks the same whether shown on the map page or from another RICMS view. Dismissing a popup does not resolve the alert, which is still shown in the “Alerts and Notification” information feed.

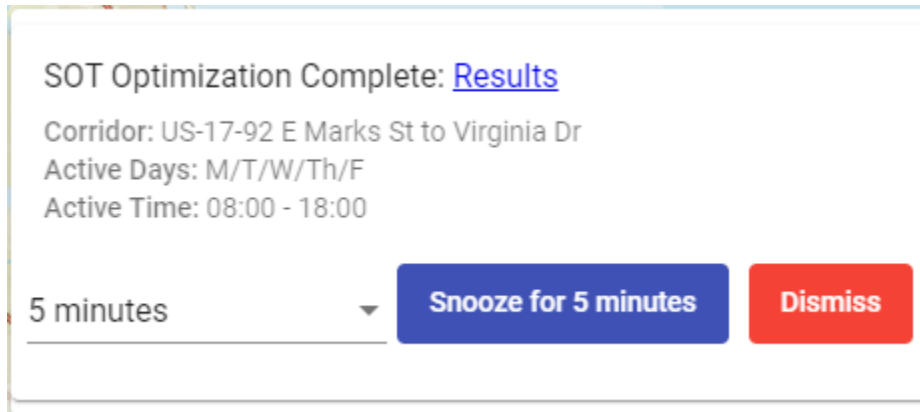


Figure 108 - SOT alert

SOT alert emails and popups and information feed messages are sent in scenarios depicted below in **Figure 109**. Individual signal device approvals will add messages to the information feed but will not trigger pop-ups or emails. All alerts are automatically included in the information feed.

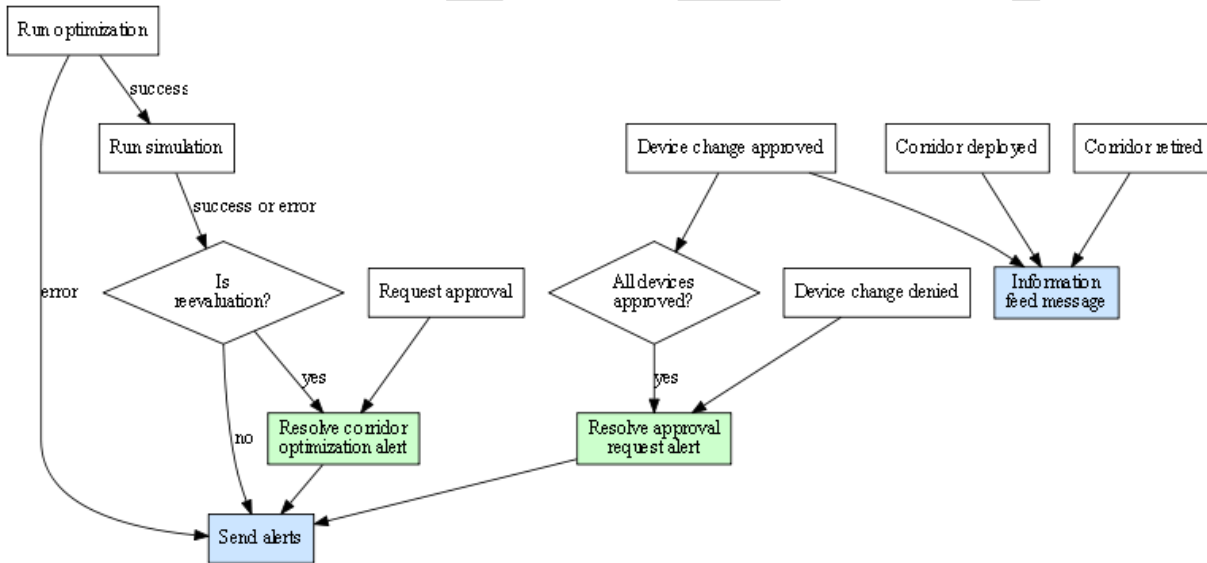


Figure 109 - SOT alert and information message flow-chart

Figure 110, **Figure 111**, and **Figure 112** below show the information feed with examples of SOT alerts, some of which are resolved at various stages of the corridor management lifecycle.

Alerts and Notifications		
Type	Message	Created Date
	SOT Devices Approved by SOTUSER2, Corridor 142: MILLS AVE: E MARKS ST to VIRGINIA DR, Active Days: M/T/W/Th/F, Active Time: 8:00 to 18:00 Results	6/5/20, 11:59 AM
	SOT Review Requested by SOTUSER1, Corridor 142: MILLS AVE: E MARKS ST to VIRGINIA DR, Active Days: M/T/W/Th/F, Active Time: 8:00 to 18:00, Results	6/5/20, 11:59 AM
	SOT Optimization Completed, Corridor 142: MILLS AVE: E MARKS ST to VIRGINIA DR, Active Days: M/T/W/Th/F, Active Time: 8:00 to 18:00, Results	6/5/20, 11:59 AM

Items per page: 10 1 - 10 of 20

Figure 110 - SOT information feed –review requested

Alerts and Notifications		
Type	Message	Created Date
	SOT Devices Denied by SOTUSER3, Corridor 142: MILLS AVE: E MARKS ST to VIRGINIA DR, Active Days: M/T/W/Th/F, Active Time: 8:00 to 18:00, Results	6/5/20, 11:59 AM
	SOT Devices Approved by SOTUSER2, Corridor 142: MILLS AVE: E MARKS ST to VIRGINIA DR, Active Days: M/T/W/Th/F, Active Time: 8:00 to 18:00 Results	6/5/20, 11:59 AM
	SOT Review Requested by SOTUSER1, Corridor 142: MILLS AVE: E MARKS ST to VIRGINIA DR, Active Days: M/T/W/Th/F, Active Time: 8:00 to 18:00, Results Resolution Message: Approval denied for device (6/5/20, 1:13 PM)	6/5/20, 11:59 AM
	SOT Optimization Completed, Corridor 142: MILLS AVE: E MARKS ST to VIRGINIA DR, Active Days: M/T/W/Th/F, Active Time: 8:00 to 18:00, Results	6/5/20, 11:59 AM

Items per page: 10 1 - 10 of 20

Figure 111 - SOT information feed – review denied

Alerts and Notifications		
Type	Message	Created Date
	SOT Corridor Deployed, Corridor 142: MILLS AVE: E MARKS ST to VIRGINIA DR, Active Days: M/T/W/Th/F, Active Time: 8:00 to 18:00, Results	6/5/20, 1:06 PM
	SOT Corridor Approved, Corridor 142: MILLS AVE: E MARKS ST to VIRGINIA DR, Active Days: M/T/W/Th/F, Active Time: 8:00 to 18:00, Results Resolution Message: Corridor deployed (6/5/20, 1:13 PM)	6/5/20, 12:06 PM
	SOT Devices Approved by SOTUSER2, Corridor 142: MILLS AVE: E MARKS ST to VIRGINIA DR, Active Days: M/T/W/Th/F, Active Time: 8:00 to 18:00, Results	6/5/20, 12:04 PM
	SOT Review(M) Requested by SOTUSER1, Corridor 142: MILLS AVE: E MARKS ST to VIRGINIA DR, Active Days: M/T/W/Th/F, Active Time: 8:00 to 18:00, Results Resolution Message: Approval granted for all devices (6/5/20, 1:13 PM)	6/5/20, 12:01 PM
	SOT Devices Denied by SOTUSER3, Corridor 142: MILLS AVE: E MARKS ST to VIRGINIA DR, Active Days: M/T/W/Th/F, Active Time: 8:00 to 18:00, Results	6/5/20, 11:59 AM
	SOT Devices Approved by SOTUSER2, Corridor 142: MILLS AVE: E MARKS ST to VIRGINIA DR, Active Days: M/T/W/Th/F, Active Time: 8:00 to 18:00, Results	6/5/20, 11:59 AM
	SOT Review Requested by SOTUSER1, Corridor 142: MILLS AVE: E MARKS ST to VIRGINIA DR, Active Days: M/T/W/Th/F, Active Time: 8:00 to 18:00, Results Resolution Message: Approval denied for device (6/5/20, 1:13 PM)	6/5/20, 11:59 AM
	SOT Optimization Completed, Corridor 142: MILLS AVE: E MARKS ST to VIRGINIA DR, Active Days: M/T/W/Th/F, Active Time: 8:00 to 18:00, Results	6/5/20, 11:59 AM

Figure 112 - SOT information feed – re-evaluation approved and deployed

3.4.4 Common Core

Common core components, consisting of libraries, will be available for use in other R-ICMS components developed for the R-ICMS system. They include functionalities that services need in order to function properly in the defined R-ICMS environment. Shared libraries will ensure that services will utilize a common method of interacting.

3.4.4.1 Kubernetes

Kubernetes is a container orchestrator that schedules application containers on the available hardware, connects them to each other and the outside world, and manages some runtime configuration of these applications. It also detects application containers that have exited and will automatically restart them, providing crash tolerance. It will be responsible for managing all drivers, data services, business services, the gateway, and large parts of the pipelines.

The configuration of the cluster state is performed through multiple manifest files in Yet Another Markup Language (YAML) format. These document what the initial deployment conditions are and can be used to bring all Kubernetes managed services up from scratch or to repair an existing cluster to this state. Updates to the cluster can also be done by modifying these files and then applying the new configuration. In the case that a new software version is deployed, Kubernetes will automatically perform a rolling upgrade.

These manifests also specify names for services that are backed by application containers or point to resources outside of the cluster. By default, these services are only visible to applications running inside the cluster, but a special kind of service will be used to allow information from outside the cluster to be passed to the gateway, which will then route the requests through the cluster to the appropriate service for handling.

Kubernetes will also be used to override default configurations in cases it makes sense, such as connection strings to outside resources.

3.4.4.2 Authorization (AuthZ-CC)

The Authorization library allows a service to authenticate tokens and to authorize that a user token contains the required permissions and devices needed to perform a particular action.

The structure of the user JWT is discussed in Section 3.4.2.1. The authorization process is as follows:

1. First, the format of the JWT is verified: the token must have a valid issuer and audience and must not be expired.
2. Next, the integrity of the token is ensured by checking that the token has been signed with the designated signing key residing in the in-memory cache.
3. Finally, any permissions and devices that are required to perform the given action are verified to be present in the token.

If any part of this process fails, the service will log the abnormality and return an Unauthorized HTTP response.

Besides handling user tokens, the AuthZ-CC also authenticates requests from one service to another service. This adds a further defense to prevent unauthorized non-R-ICMS services from making calls to sensitive R-ICMS service endpoints.

3.4.4.3 Logging (Log-CC)

Services will be responsible for providing information for the system logs. The Log-CC component will allow each service to use a common interface to provide log data. There will be small libraries that wrap and configure standard logging libraries for each language used. Log levels will be defined in the system and have six levels of severity:

- **Fatal:** Used when the application has encountered an unrecoverable error that prevents the application from running at all, meaning the only viable option is to crash the application and restart it from scratch.
- **Error:** Used when something that prevents the current request or job from being processed, but can be corrected between requests or jobs, such as a problem with network or database connectivity.
- **Warning:** Used when a problem occurs that is recoverable during the same request, allowing it to be fulfilled. An example of this would be using a default configuration value where one was explicitly expected or needing to retry a connection to a resource.

- **Information:** Used to state the success of expected events during the normal operation of the program.
- **Debug:** Used to state actions that are going to be tried. In tandem with the information level messages, this can be used to confirm that an expected path was taken through the application.
- **Verbose:** Used for logging large data, e.g. entire messages or data structures, or for logs that may occur many times per second, e.g. when in a loop that may run several times. This may also contain tracing information, such as which functions have been called.

Log levels will be changeable for specific components without necessitating a restart, ensuring the system can be monitored at different levels at different times depending on user need. There will be a service that aggregates the log level controls of individual components to facilitate this.

The logging library we use will be made available to the parts of the application that need it by way of constructor-based dependency injection. Any controller that needs access to a logger will declare its need by specifying the logging interface in its constructor and the dependency injector will provide the logger upon instantiation of the class.

The log messages will be logged in JSON, a machine-readable format, to facilitate automatic aggregation and analysis. This avoids transforming data that is structured within the application to a plain text log and then parsing it back into structured data. The outermost fields of the JSON message will be the same to facilitate searches on those common values. The fields are:

- **Timestamp:** an ISO 8601 date time including time zone.
- **Level:** one of the above severity levels.
- **MessageTemplate:** the log message without specific values substituted in.
- **Properties:** all the context that could be used to fill out the MessageTemplate. Any message specific values will appear here.

The logs will be shipped from whichever machine they are generated on to an Elasticsearch cluster via the use of Filebeats. The Filebeat will be configured to read the logs of any Docker container on the system and send them to Elasticsearch.

All logs generated by the system will be aggregated in Elasticsearch. It is designed for storage and search of semi-structured data such as logs. We will use it in conjunction with Kibana, a data visualization tool, to generate charts and dashboards from the logged information. Elasticsearch will also use the aggregated logs as the basis for system monitoring.

3.4.4.4 Monitoring (Mon-CC)

Internal and external software components will be monitored to ensure the system is running properly. When these software components enter undesired states or report issues, the Monitoring-CC component will alert appropriate system users using the Notification Common Core component. This allows the system to continuously keep track of the status of the system in order to have the earliest warning of failures, defects, or problems.

Table 50 – Monitoring Alerts

Alert Name	What is being monitored	Resolvable	Alert Message	Resolution Message
Unsuccessful Data Retrieval	Unsuccessful data retrieval from data source	True	The {driver-name} has not successfully retrieved data for a configured time period.	The {driver-name} is successfully retrieving data.
Unavailable System Component	Kubernetes/Docker Swarm services' availability	True	The {service-name} is unavailable or under-replicated.	The {service-name} is now available.
Invalid Data	Invalid data retrieval	True	Received invalid data from the {data-source}.	Valid data is now being received from {data-source}
Critical Errors	Critical log messages.	False	The {service-name} reported the following critical error: {error}	N/A

Table 50 – Monitoring Alerts shows the different types of conditions the system will monitor and produce alerts for. If the result is resolvable the system will detect the resolved condition and resolve the alert, giving notification to users that the alert has been resolved. Critical errors are marked as unresolvable. The system will continue to alert on critical alerts at a configurable interval until the alert has been resolved.

The Mon-CC component is composed of Elasticsearch, Watchers, and the Monitoring Service. Elasticsearch is a distributed search/analytics engine and acts as the primary data store for RICMS application logs. A Watcher is a feature of Elasticsearch that is used to monitor the incoming logs. The Watchers send information to the Monitoring Service for further processing when configured conditions are met. The Monitoring Service then processes the information sent by the Watcher and determines if an alert should be created.

Figure 113 – Unresolvable Alert Monitoring Sequence Diagram shows the monitoring process for an unresolvable alert. An unresolvable alert notifies authorized users of an event without additional update information being provided. There are no external actions (passive or active) that can be taken to change the status of these alerts.

R-ICMS Unresolvable Alert Monitoring Sequence Diagram

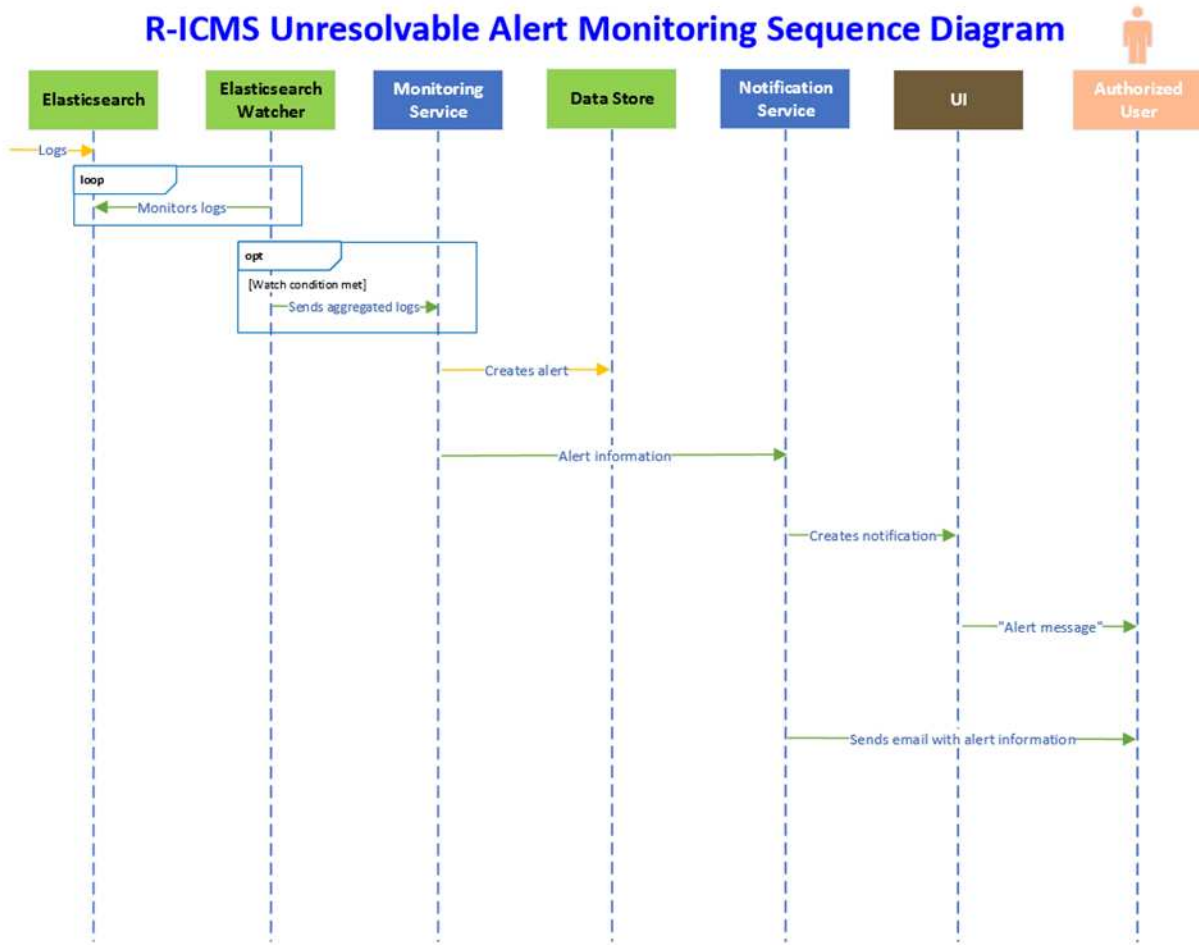


Figure 113 – Unresolvable Alert Monitoring Sequence Diagram

Figure 114 – Resolvable Alert Monitoring Sequence Diagram shows the monitoring process for a resolvable alert. A resolvable alert is an alert whose status can change based on external actions. After a resolvable action (passive or active) takes place, the alert becomes resolved and the authorized users are notified of the resolution.

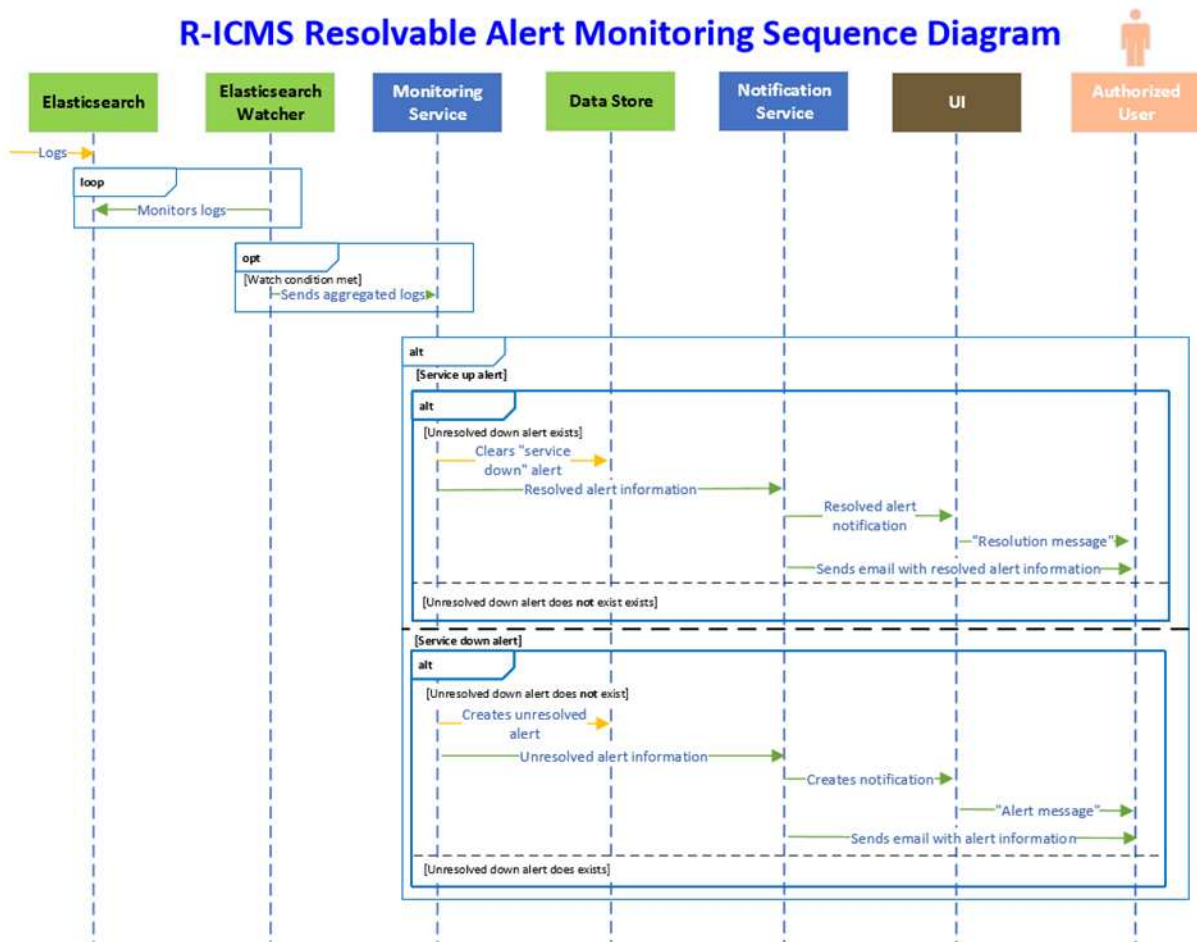


Figure 114 – Resolvable Alert Monitoring Sequence Diagram

3.4.4.5 Notification (Not-CC)

The Not-CC provides functionality for handling notifications. This common notification library interacts with the NOT-BS, allowing a service to create both alerts and announcements. In the case of alerts, the NOT-BS returns a unique notification ID that can later be used by the Not-CC to resolve that alert.

3.4.4.6 Gateway (GW-CC)

As shown in the high level architecture in Section 3.3, a Gateway (GW-CC) will sit between user interfaces and the services they can access. The Gateway will provide a single point of contact for user systems and provide the load leveling for the external user load on the data and business services. The Gateway will distribute messages from users to the appropriate service and messages from the services to the appropriate user.

3.5 System Resources

To support the R-ICMS in its proposed configuration, the following hardware needs have been identified. All servers will be hosted in the FDOT D5 hosting environment and will follow FDOT D5 standards for hosted systems. As specified in the contract, these standards include:

- Agency for State Technology (AST) Chapter 74-2 F.A.C., Information Technology Security, also known as the Florida Cybersecurity Standards (FCS)
- FCS, 74-2 F.A.C. Section 74-2.002 (4)

3.5.1 Hardware Configuration

Components within the business and data layers are built to run in a container. The container dynamically occupies a portion of resources on a server allocated to that layer. The dynamic allocation is administered using the Kubernetes orchestration tool as described in Section 3.7.2 Containerized Service Orchestration.

Network infrastructure is provided by the Department.

Table 51 – Hardware Configuration describes the servers and the layers to which they are assigned.

Table 51 – Hardware Configuration

Server details	# machines	Virtual / Physical	Configuration	Layer
SQL DB Server – 1 GIS DB Server - 1	2	V	128 GB / 8 Core / 4 TB RAID 10	Data Store
ArcGIS Server – 1 GeoEvent Server -1	2	V	128 GB / 16 Core / 1 TB RAID 10	Data Store
GIS App Server – 1 Business Services Server – 1 Gateway Server - 1	3	V	64 GB / 8 Core / 1 TB RAID 10	Data/Business Services
Web Servers – 2 Application Servers - 2	4	V	128 GB / 16 Core / 4 TB RAID 10	Data/Business Services
Kafka Cluster	3	P	2 x 8 Core , 128GB RAM w/ 2 x 1TB RAID 1, 4 x 2TB RAID 1 individual	Pipelines
Cloudera Manager	3	P	2 x 8 Core , 128GB RAM w/ 2 x 1TB RAID 1, 2 x 1TB RAID 1, 2 x 1TB individual	Data Store
Cloudera Data Nodes	9	P	2 x 8 Core , 128GB RAM w/ 2 x 1TB RAID 1, 6 x 2TB Individual	Data Store
Edge nodes for client sessions / users	3	P	2 x 8 Core , 128GB RAM w/ 2 x 1TB RAID 1, 1 x 1TB Individual	User Interface
Edge nodes for ETL	3	P	2 x 8 Core , 256GB RAM w/ 2 x 1TB RAID 1, 6 x 2TB Individual	Data Store

Server details	# machines	Virtual / Physical	Configuration	Layer
Elastic search Master nodes	2	P	2 x 8 Core, 128GB RAM w/ 2 x 1TB RAID 1, 1 x 1TB individual	Data Store
Elastic Search Workers	3	P	2 x 8 Core , 128GB RAM w/ 2 x 1TB RAID 1, 2 x 8TB RAID 1 individual	Data Store
NoSQL	3	P	2 x 8 Core , 256GB RAM w/ 2 x 1TB RAID 1, 6 x 2TB individual	Data Store

3.5.2 Physical Deployment: Servers and Network

This diagram shows the deployment of physical servers hosting R-ICMS services within private virtual networks, including routing through an internet firewall for any external server access.

DRAFT

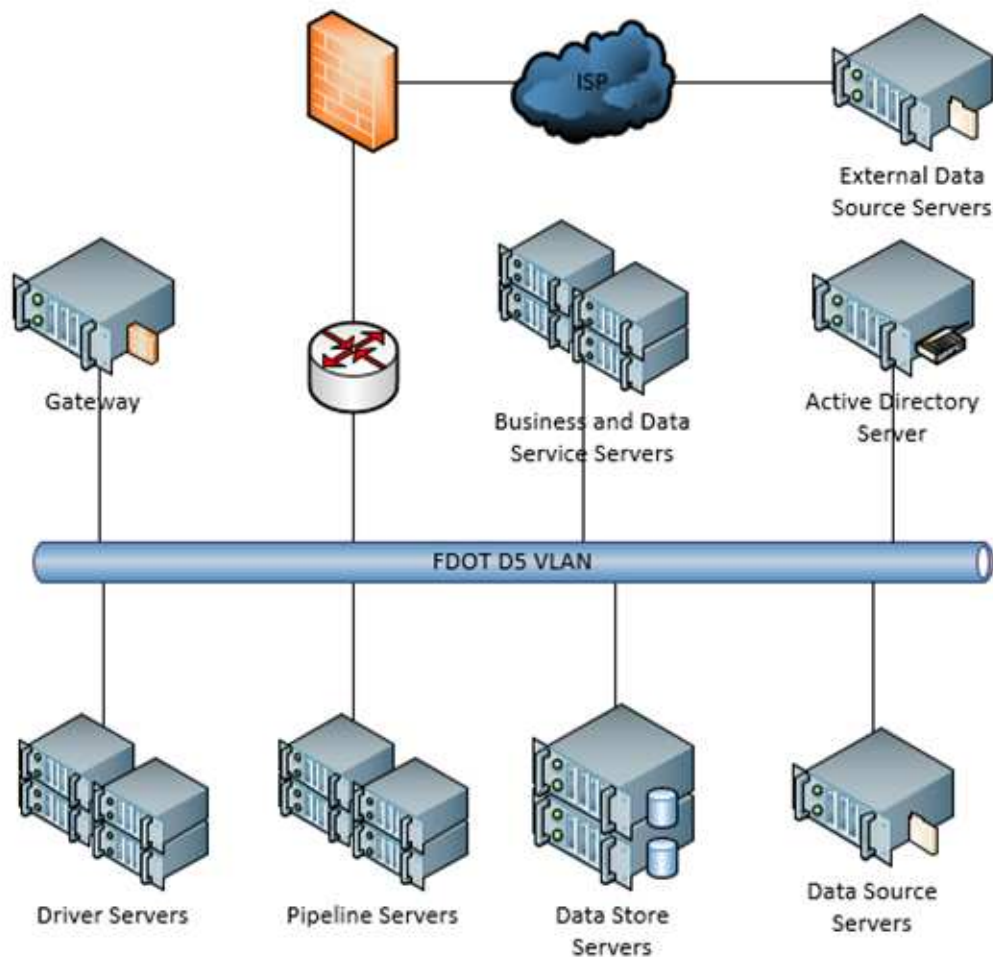


Figure 115 – Physical Deployment: Servers and Network Diagram

3.6 Concept of Execution

The section contains the diagrams and descriptions showing the dynamic relationships among the software modules including how the modules will interact during system operation. The sequence diagrams use standard Unified Modeling Language (UML), as constrained by using Visio. In the sequence diagrams, a yellow arrow is used to imply that the content of a message is stored, as well being transmitted in the message. The initial sequence diagrams provide a more detailed understanding of the authentication and authorization steps in the interactions of system components. The lack of those detailed steps in the remaining diagrams is not intended to imply that they do not occur; rather, the lack of detail is only intended to make the diagrams cleaner and therefore clearer for the specific interactions being focused on. Likewise, the logging sequence diagrams show more details for the logging of data than other diagrams. Similarly, the intent is not that the logging will not occur during other interactions, but that the other sequence diagrams can be clearer by not spelling out all of the logging in those interactions. The sequence diagrams shown in these graphics primarily address the “happy path” interactions. “Unhappy

paths” will typically take the form of error messages. Additional “unhappy paths” implementation will be addressed in the appropriate detailed design section.

Other diagrams are more informal in nature and intended primarily to clarify the information presented.

A list of components used in the diagrams is included in Section 3.4 along with descriptions of the components. In some cases, generic components are referenced when any component of that type can be involved in the interaction.

The coloring scheme of Figure 2 is used to depict the layer within the high-level architecture that each represented component resides.

3.6.1 Security

Security is essential to the functionality of the R-ICMS. This section contains the relevant diagrams concerning the security processes needed to ensure data is accessed in a secure manner. The following diagrams are intended to show how the security processes will work throughout the system such that following diagrams will not have to specifically reference security features.

Exchanges of credentials/tokens and other data between users and the gateway are performed over HTTPS to prevent sniffing.

3.6.1.1 Login

Figure 116 – Login Sequence Diagram shows the system login process. User login requests are authenticated via Active Directory/ Lightweight Directory Access Protocol (LDAP), and active sessions are represented internally as signed tokens containing user permissions.

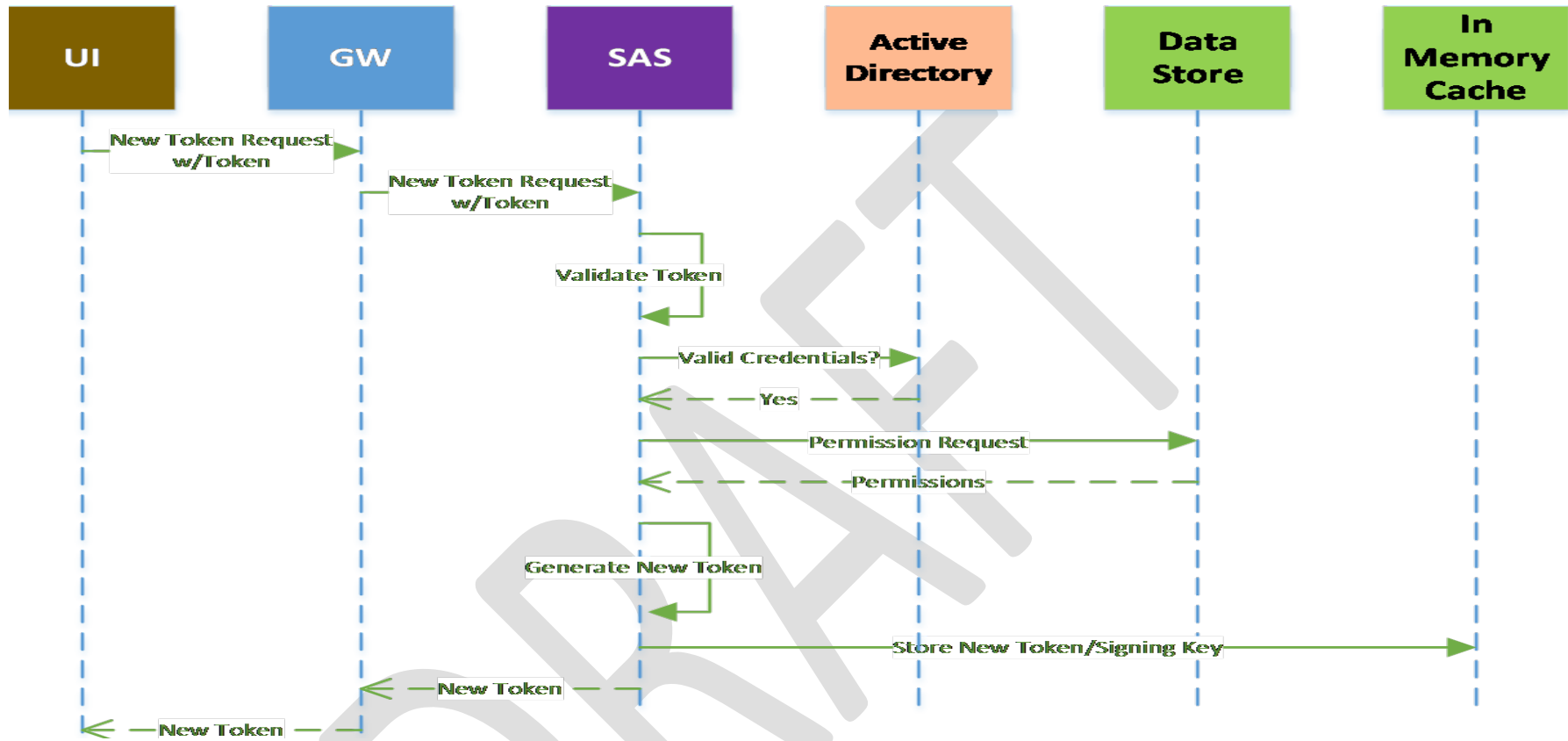


Figure 116 – Login Sequence Diagram

3.6.1.2 User Data Request

Figure 117 – User Data Request Sequence Diagram shows the interaction for any data request or setting up a streaming data subscription using a token. This includes verification of the token and user permissions for that request. System services use the authorization library, a common core component, to validate security tokens using the associated signing key from an in-memory cache. The authorization library is also used to validate that permissions required by a system request match the user permissions within the token.

DRAFT

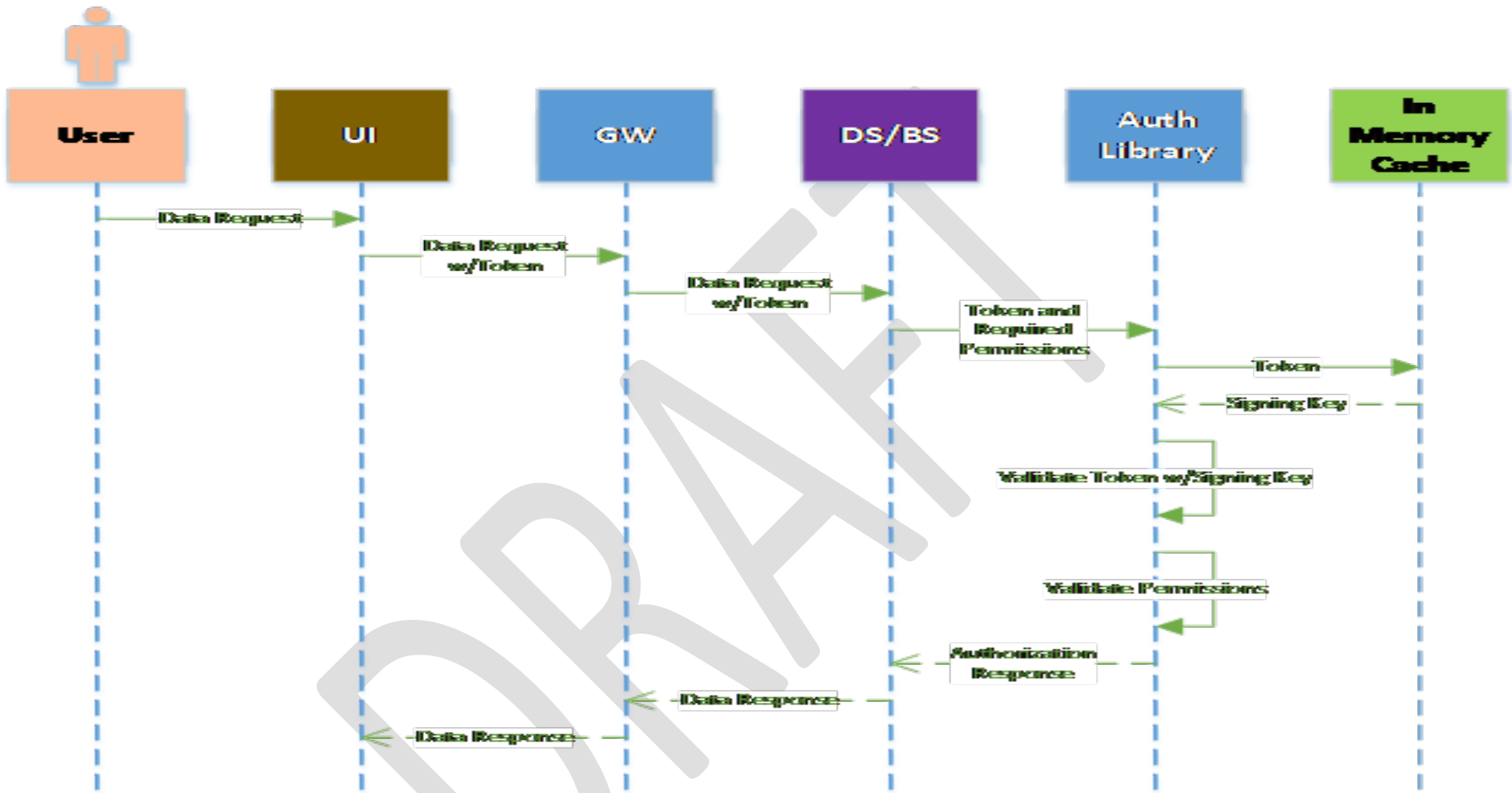


Figure 117 – User Data Request Sequence Diagram

3.6.1.3 User Authorization Update

Figure 118 – User Authorization Update Sequence Diagram shows how an active user session may be extended beyond the token expiration time by renewing a token. The user interface is responsible for automatically renewing a token before it expires without any user interaction.

DRAFT

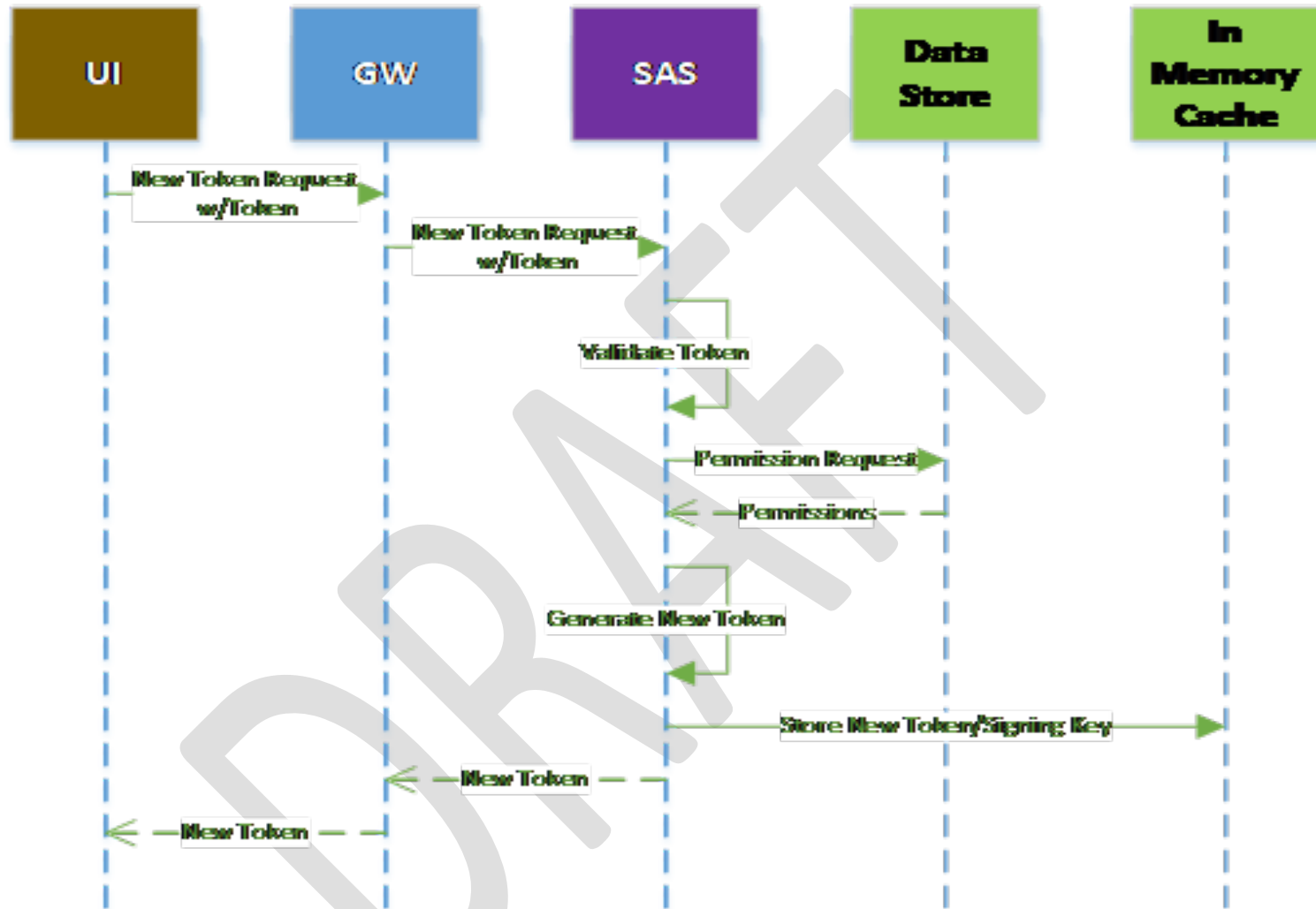


Figure 118 – User Authorization Update Sequence Diagram

3.6.1.4 User Personalization

Figure 119 – User Personalization Sequence Diagram shows how user may personalize view and filter settings, which are persisted across login sessions via the User Personalization business service.

DRAFT

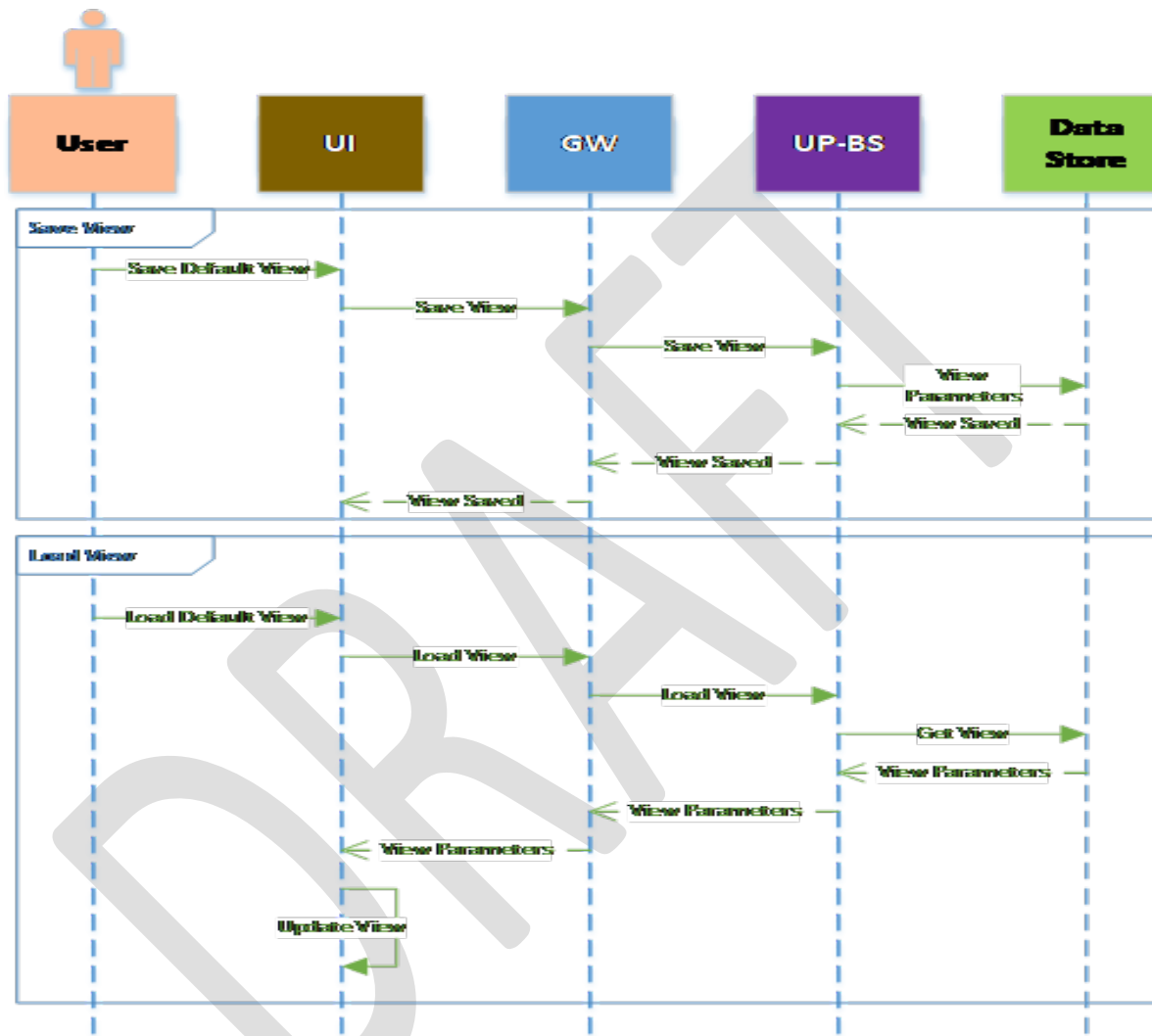


Figure 119 – User Personalization Sequence Diagram

3.6.1.5 GIS Use Cases

Figure 38 – GIS Use Cases shows the use cases supported by the GIS. The tables following the use case diagram, Table 52 – Use Case View Map Details, Table 53 – View GIS Layers Use Case Details, Table 54 – View Details Use Case Details, and Table 56 – Receive Dynamic Data Use Case Details, provide details for the illustrated use cases.

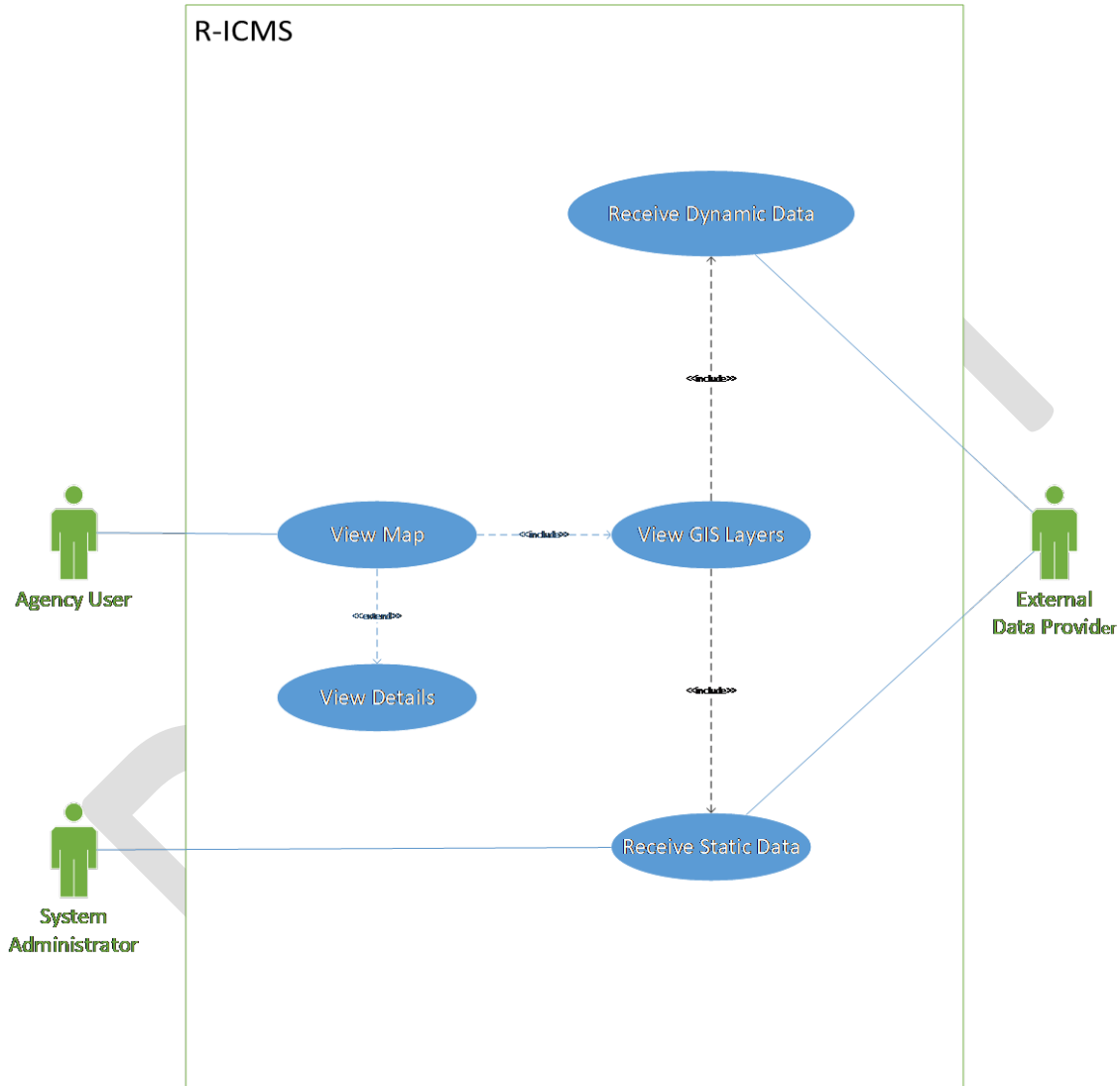


Figure 120 – GIS Use Cases

Table 52 – Use Case View Map Details

Use Case Information	
Use Case ID	UC.GIS.1
Use Case Name	View Map
Summary	User is able to view the map, map navigation and legend
Actors	
Primary	System User
Secondary	None
Pre-conditions	The user must be logged into the system and have privileges to access the map feature
Main Scenario	
Trigger	User Interaction
Typical Events	<ol style="list-style-type: none"> 1. The use case begins when the user selects the function to view the map 2. The system displays the Map 3. Use case ends
Includes	View Map Details, View GIS Layers
Input	None
Output	Map View
Alternate Scenarios	
Variants	<p>Refresh the map</p> <ol style="list-style-type: none"> 1. During the basic flow, [system displays the map], the user may select to refresh the page 2. The system will refresh the contents displayed on the map and the use case continues in the basic flow at [system displays the map]
	<p>Zoom In and Out</p> <ol style="list-style-type: none"> 1. During the basic flow, [system displays the map], the user may select to change the zoom (in/out) 2. The system will adjust the view to the user selected zoom level and the use case continues in the basic flow at [system displays the map]
	<p>Pan</p> <ol style="list-style-type: none"> 1. During the basic flow, [system displays the map], the user may select to pan the map to view a different area by using the click and drag or map overview methods 2. The system will adjust the view to the user selected pan location and the use case continues in the basic flow at [system displays the map]
	<p>Home View</p> <ol style="list-style-type: none"> 1. During the basic flow, [system displays the map], the user may select to return to the default view 2. The system will adjust the view to the system default zoom and location view and the use case continues in the basic flow at [system displays the map]
	<p>Search</p> <ol style="list-style-type: none"> 1. During the basic flow, [system displays the map], the user may search using the parameters configured for the respective layers/features 2. The system will perform the search and display the results on the map and the use case continues in the basic flow at [system displays the map]
Extends / Uses	View GIS Layers
Post-conditions	Map is displayed
Frequency	User driven
Exceptions	User cancels the use case. The use case ends
Related Use Cases	None

Table 53 – View GIS Layers Use Case Details

Use Case Information	
Use Case ID	UC.GIS.2
Use Case Name	View GIS Layers
Summary	User selects/deselects items in the map layers to customize view
Actors	
Primary	System User
Secondary	None
Pre-conditions	The user must be logged into the system and have privileges to use the map feature
Main Scenario	
Trigger	User Interaction
Typical Events	<ol style="list-style-type: none"> 1. The use case begins when the user selects the function to view map layers 2. The system displays the available layers 3. User selects/deselects layers 4. Map displays user selected layers 5. The use case ends
Includes	None
Input	Selection of desired layers
Output	Map is displayed with user selected layers
Alternate Scenarios	
Variants	<p>View Legend</p> <ol style="list-style-type: none"> 1. During the basic flow, [system displays the available layers], the user may select to view the map legend 2. The system will display the legend and the use case continues in the basic flow at [system displays the available layers] <p>View Sublayers</p> <ol style="list-style-type: none"> 1. During the basic flow, [system displays the available layers], the user may select to view the map sublayers 2. The system will display the sublayers that exist for each map layer and the use case continues in the basic flow at [system displays the available layers]
Extends / Uses	Events, Weather, Traffic, Device, Parking Data
Post-conditions	Map is displayed with user selected layers
Frequency	User driven
Exceptions	User cancels the use case. The use case ends
Related Use Cases	None

Table 54 – View Details Use Case Details

Use Case Information	
Use Case ID	UC.GIS.3
Use Case Name	View Details
Summary	User selects item on map to view/edit
Actors	
Primary	System User
Secondary	None
Pre-conditions	1. The user must be logged into the system and have privileges to view the map 2. The user has selected a layer to display items on a map
Main Scenario	
Trigger	User Interaction
Typical Events	1. The use case begins when the user selects an item on the map 2. The system displays item details 3. Use case ends
Includes	None
Input	None
Output	The map displays the detail of the user selected item
Alternate Scenarios	
Variants	Close details
Extends / Uses	None
Post-conditions	None
Frequency	User driven
Exceptions	No details available. User cancels the use case. The use case ends
Related Use Cases	None

Table 55 – Receive Static Data Use Case Details

Use Case Information	
Use Case ID	UC.GIS.4
Use Case Name	Receive Static Data
Summary	System receives static data to populate GIS layers
Actors	
Primary	System, System Administrator
Secondary	External Data Provider
Pre-conditions	Static data file is available in pre-defined location on the external source Schedule exists for static file updates Authentication credentials exist for external data source
Main Scenario	
Trigger	New static data file is identified OR A scheduled event is triggered
Typical Events	<ol style="list-style-type: none"> 1. The use case begins when the system executes a script to connect to the external data source 2. The system credentials are authenticated and returns the file details 3. The system will log the file activity and verify that updates exist since the previous file update 4. The system will ingest the updated data 5. The system will update the data in the GIS server 6. The system will refresh the map data on the UI display within 3 seconds 7. The use case ends
Includes	None
Input	FTP file
Output	Updated GIS Map, Log
Alternate Scenarios	
Variants	<p>Manual Update</p> <ol style="list-style-type: none"> 1. The use case begins when the system administrator copies the file to the R-ICMS system 2. The system will log the file activity and identify duplicates or changes in the data and appends the data to the system 3. The system administrator publishes the updates through GIS tools 4. The system will refresh the map on the UI display within 3 seconds 5. The use case ends
Extends / Uses	None
Post-conditions	Updated GIS Map, Log history
Frequency	
Exceptions	User cancels the use case. The use case ends
	No update available
Related Use Cases	None

Table 56 – Receive Dynamic Data Use Case Details

Use Case Information	
Use Case ID	UC.GIS.5
Use Case Name	Receive Dynamic Data
Summary	System receives dynamic data to populate GIS layers
Actors	
Primary	System
Secondary	External Data Provider
Pre-conditions	Dynamic data file is available in pre-defined location on the external source Authentication credentials exist for external data source
Main Scenario	
Trigger	New dynamic data file is identified
Typical Events	<ol style="list-style-type: none"> 1. The use case begins when the system executes a script to connect to the external data source 2. The system credentials are authenticated and returns the file details 3. The system will verify that updates exist since the previous file update and log history of activity 4. The system will ingest the updated data and refresh the map within 3 seconds for any GIS data 5. The use case ends
Includes	None
Input	FTP file
Output	Updated GIS Map, Log
Alternate Scenarios	
Variants	
Extends / Uses	None
Post-conditions	Updated GIS Map, Log history
Frequency	
Exceptions	User cancels the use case. The use case ends
	No update available
Related Use Cases	None

3.6.2 Situational Awareness

Situational Awareness pertains to updates coming in from external sources that need to be shared with relevant data consumers including (but not limited to) the ME, business rules engine, and GeoEvent service, which filters and forwards info to agency users based on their map or other UI views. The following diagrams depict the flow of data when updates come in from ITSQA as well as how the map updates the views for pre-located (e.g. DMS) as well as dynamically-located (e.g. Events) data.

3.6.2.1 ITSQA Ingestion

Figure 121 – ITSQA Ingestion Sequence Diagram shows the asynchronous flow of updates through the ICMS system and to the user interface after an update subscription has been established. In this example, unbounded traffic data is ingested from an ITSQA pipeline to a traffic data service. Users subscribed to a traffic data service feed receive updates as data arrives.

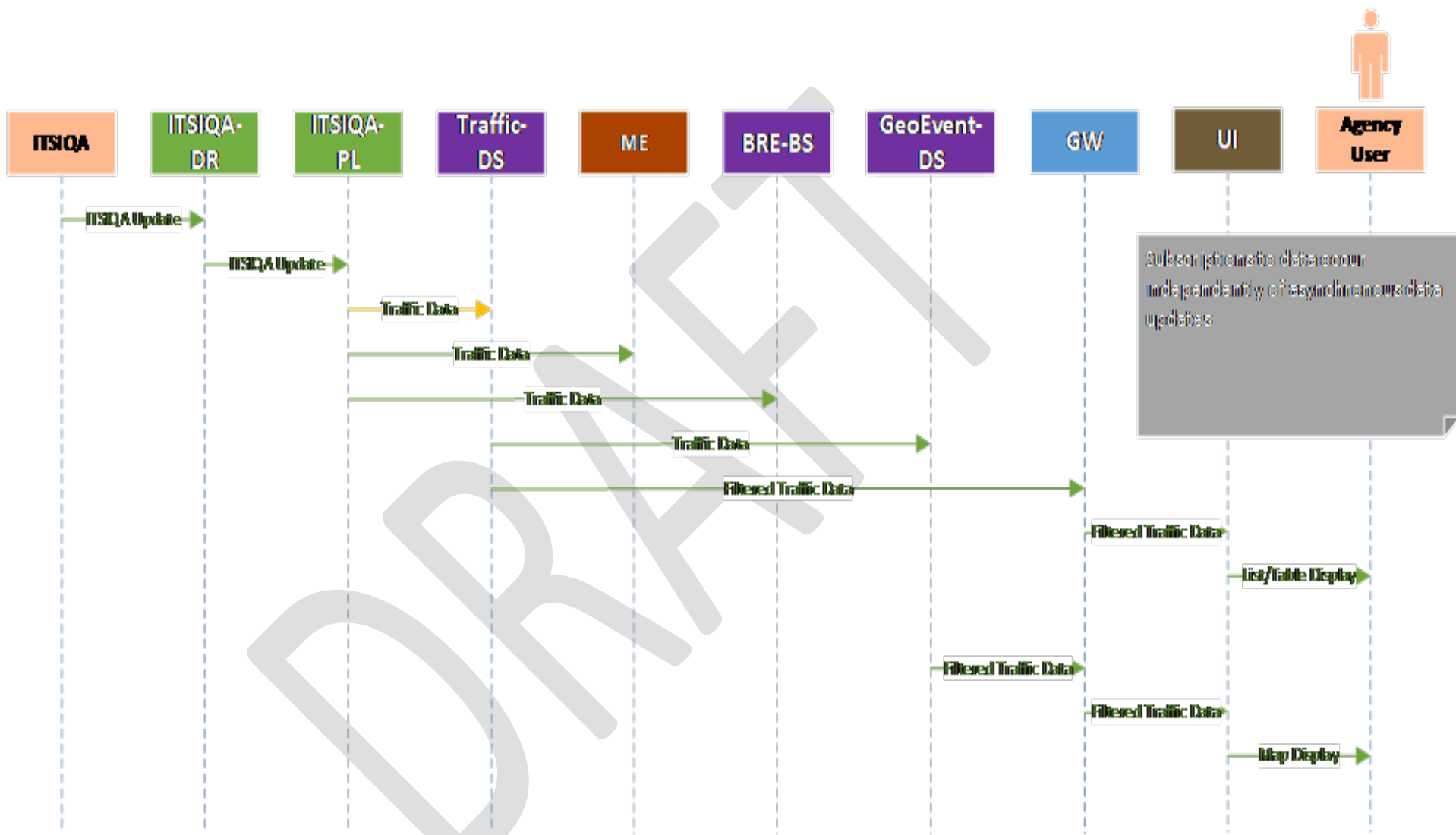


Figure 121 – ITSQA Ingestion Sequence Diagram

3.6.2.2 Map Update for Pre-Located Data

Figure 122 depicts the initiation of a subscription for pre-located feature updates. This includes features that have a static position that have dynamic status, such as DMS. A single subscription request is made to the GeoEvent Data Service and it will push any updates it receives from the pipeline that match the current filters, such as geographical extent, to the user interface, which will then update the feature in its map.

DRAFT

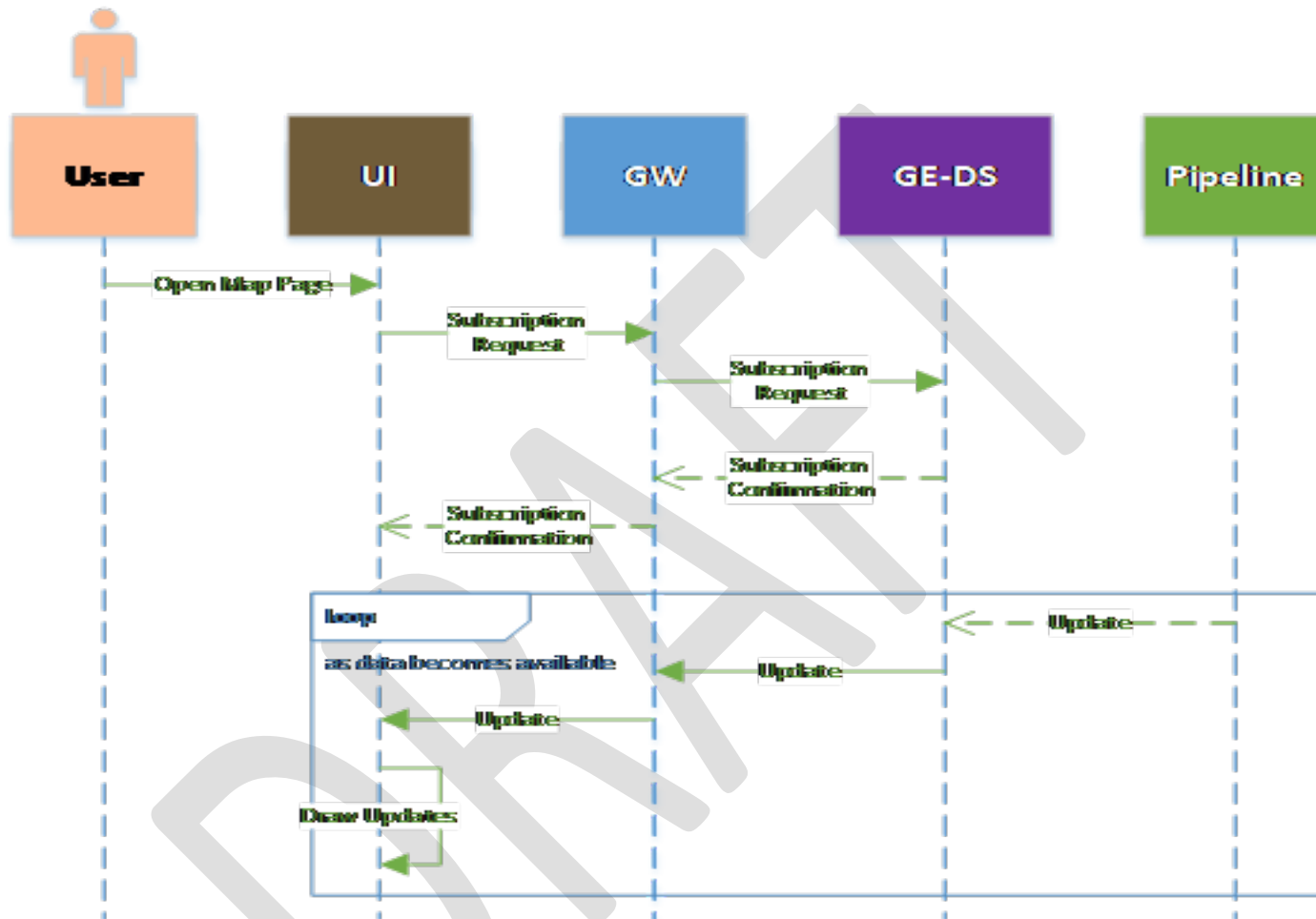


Figure 122 – Map Update for Pre-Located Data Sequence Diagram

3.6.2.3 Map Update for Dynamically Located Data

Figure 123 – Map Update for Dynamically Located Data Sequence Diagram is similar to Figure 122 – Map Update for Pre-Located Data Sequence Diagram but generalizes subscriptions for dynamically located data. Dynamically located data does not come from the GeoEvent Data Service and may come from multiple other Data Services.

DRAFT

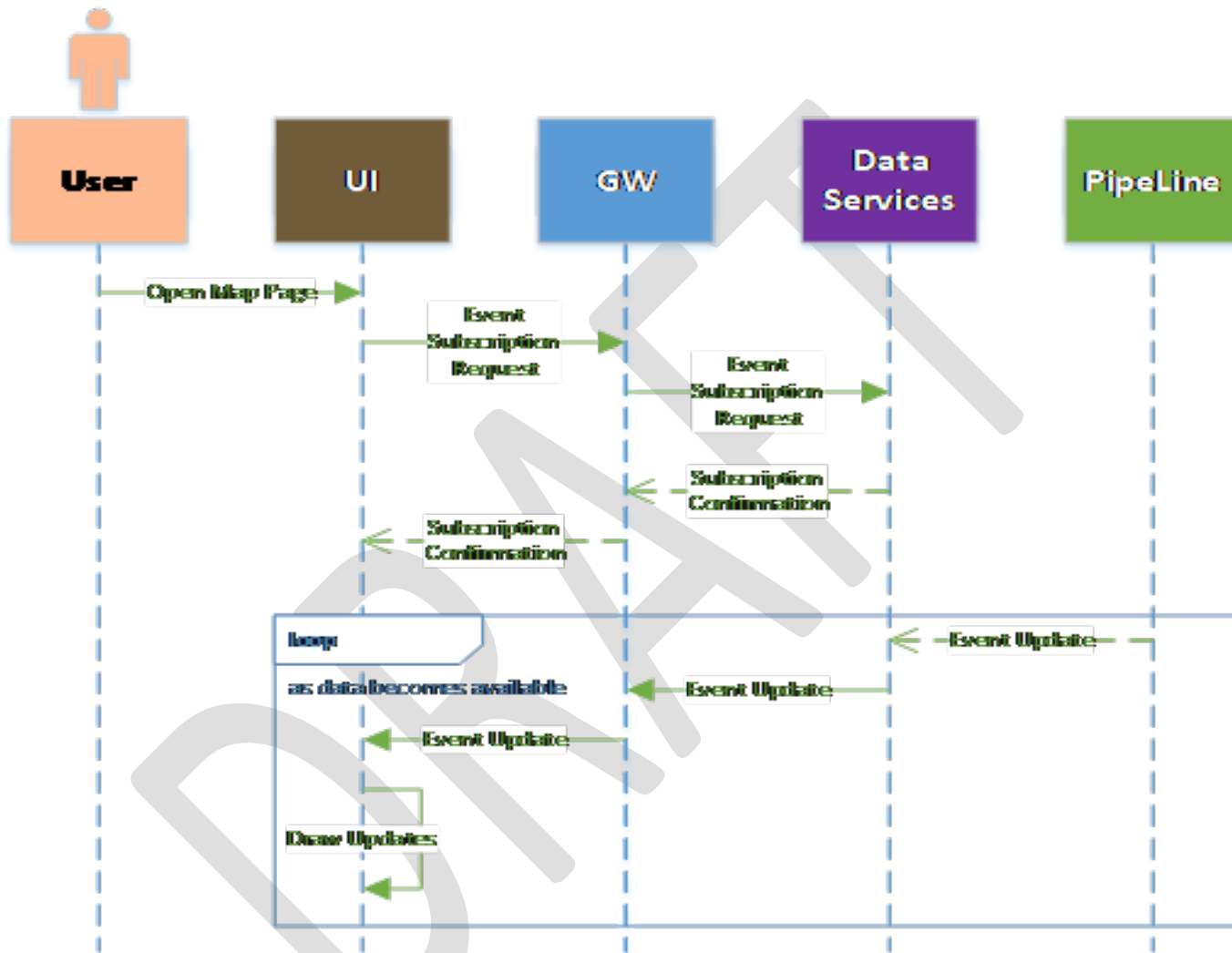


Figure 123 – Map Update for Dynamically Located Data Sequence Diagram

3.6.3 Event Diagrams

The initial implementation of the R-ICMS will generate response plans when events are generated from SunGuide or within R-ICMS. This section depicts the diagrams related to the processing of these events. It describes the main functionality contained within the DSS for processing events and initiating response plans. It describes the high-level functionality for response plan selection, response plan approval, response plan activation, and response plan reevaluation.

3.6.3.1 SunGuide / R-ICMS Event

The diagram in Figure 124 – SunGuide / R-ICMS Event Flow Diagram is a flow-chart-like diagram that shows the relationship between SunGuide and R-ICMS events. SunGuide’s Incident Detection System will notify an operator when the R-ICMS system activated a response plan. If the event that triggered the response originated in SunGuide, then identifiers for the event in both systems are known and linked automatically. If the trigger event originated in R-ICMS, the SunGuide operator will be notified and have the option to link the event to a known SunGuide event.

System Design Document for Regional Integrated Corridor Management System (R-ICMS)

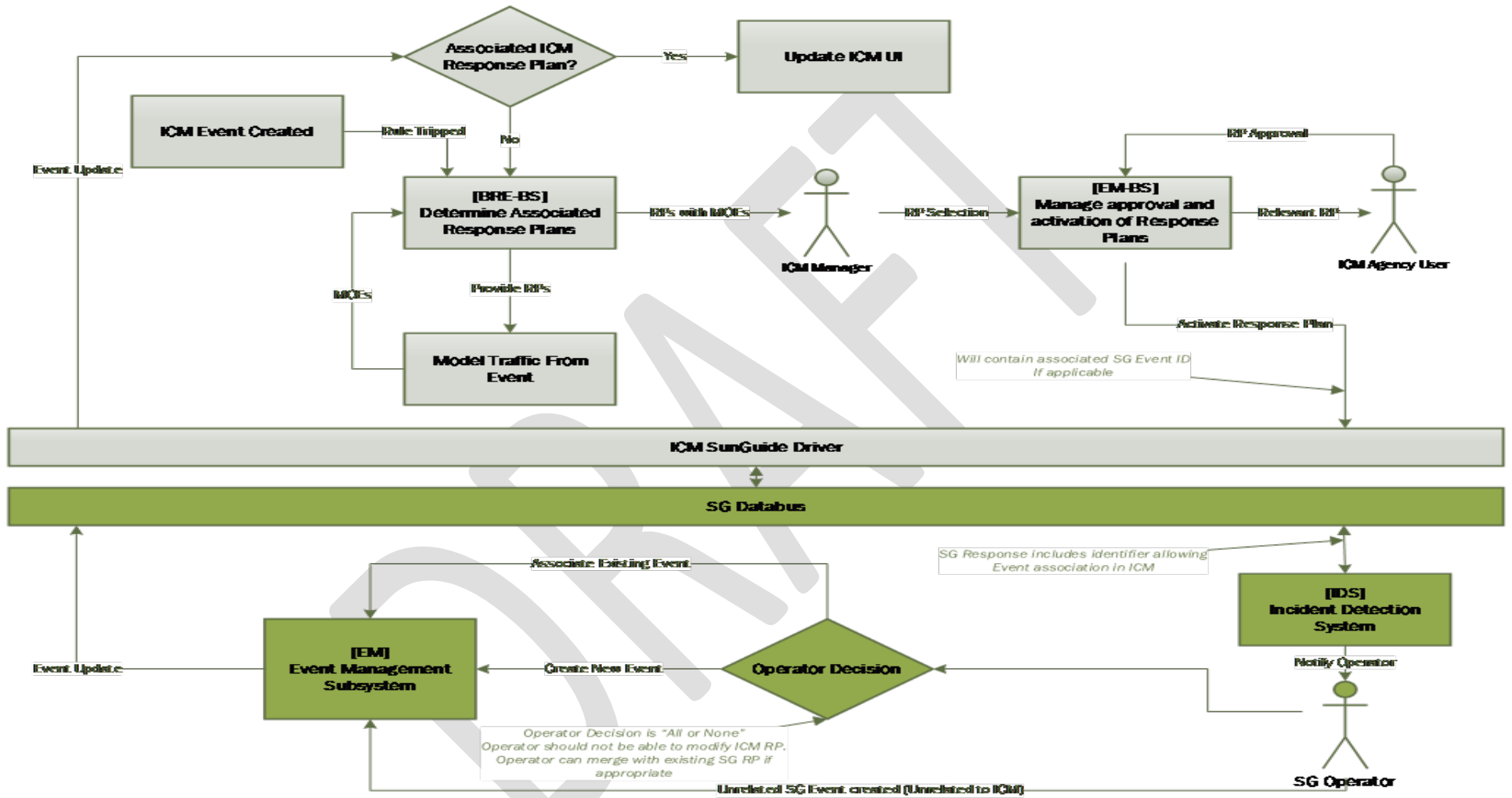


Figure 124 – SunGuide / R-ICMS Event Flow Diagram

SunGuide / R-ICMS Event Flow

Figure 125 – SunGuide / R-ICMS Event Flow Sequence Diagram shows Event Management by the business service (EM-BS), which subscribes to SunGuide event data feeds, creates and correlates R-ICMS events to SunGuide events, and publishes all modifications through the Event Management data service (EM-DS). Various system components subscribe to real-time data updates from the EM-DS, including the business rules engine, map view, event list view, and event detail view.

DRAFT

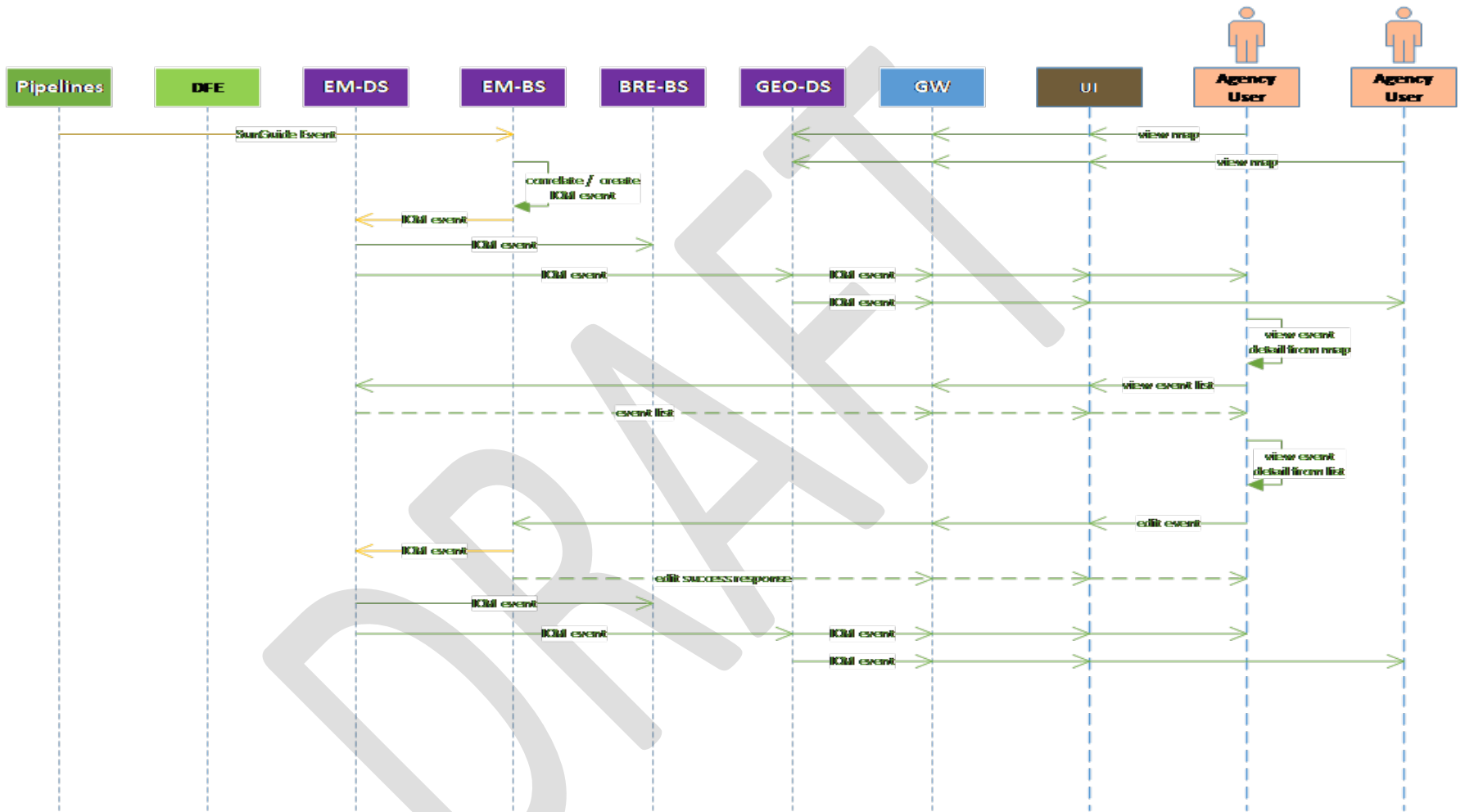


Figure 125 – SunGuide / R-ICMS Event Flow Sequence Diagram

3.6.3.2 Response Plan Selection

Figure 126 – Response Plan Selection Sequence Diagram shows the process for background triggering and ranking of alternative response plans caused by external data feeds and periodic runs of the business rules engine and surfacing those results to a Corridor Manager user.

DRAFT

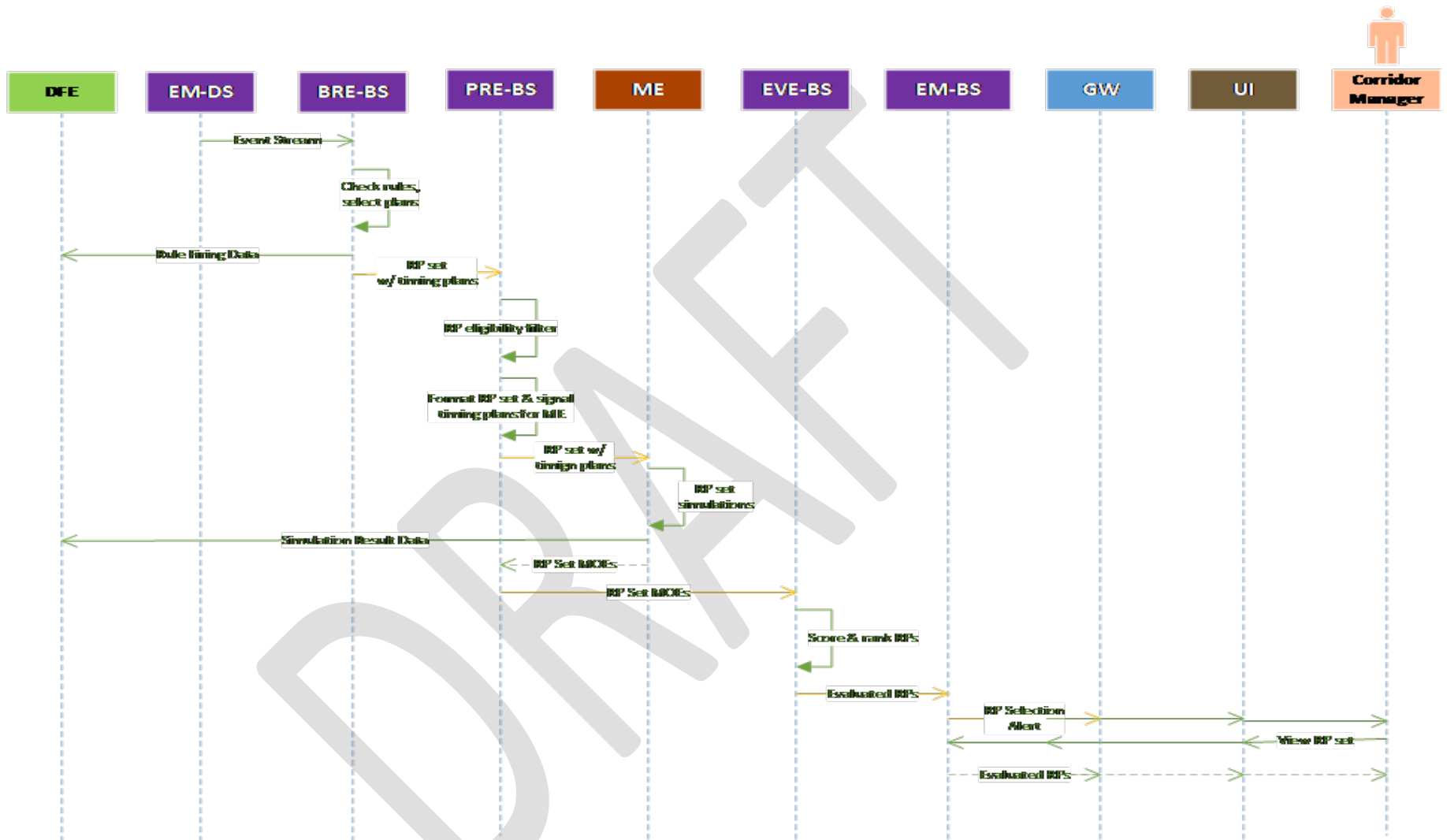


Figure 126 – Response Plan Selection Sequence Diagram

3.6.3.3 Response Plan Approval

Figure 127 – Response Plan Approval Sequence Diagram shows the process of a corridor manager user choosing a response plan from those provided by the RP Selection flow, R-ICMS soliciting for and receiving the approval of all involved agencies, and the corridor manager initiating activation of the approved response plan.

DRAFT

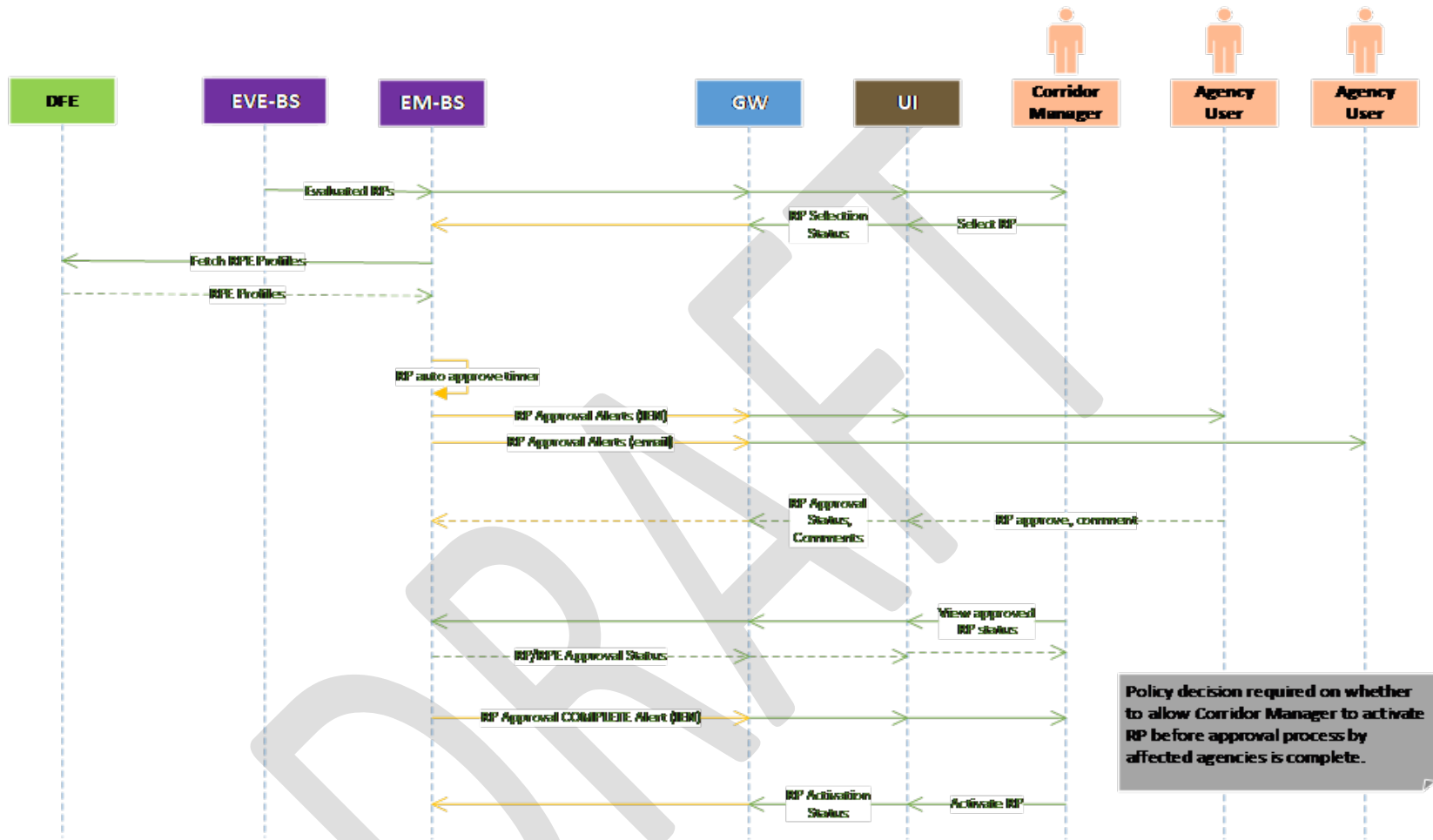


Figure 127 – Response Plan Approval Sequence Diagram

3.6.3.4 Response Plan Activation

Figure 128 – Response Plan Activation Sequence Diagram shows how a response plan is enacted after it has been activated. This involves both automatic and manual changes to the involved infrastructure.

DRAFT

System Design Document for Regional Integrated Corridor Management System (R-ICMS)

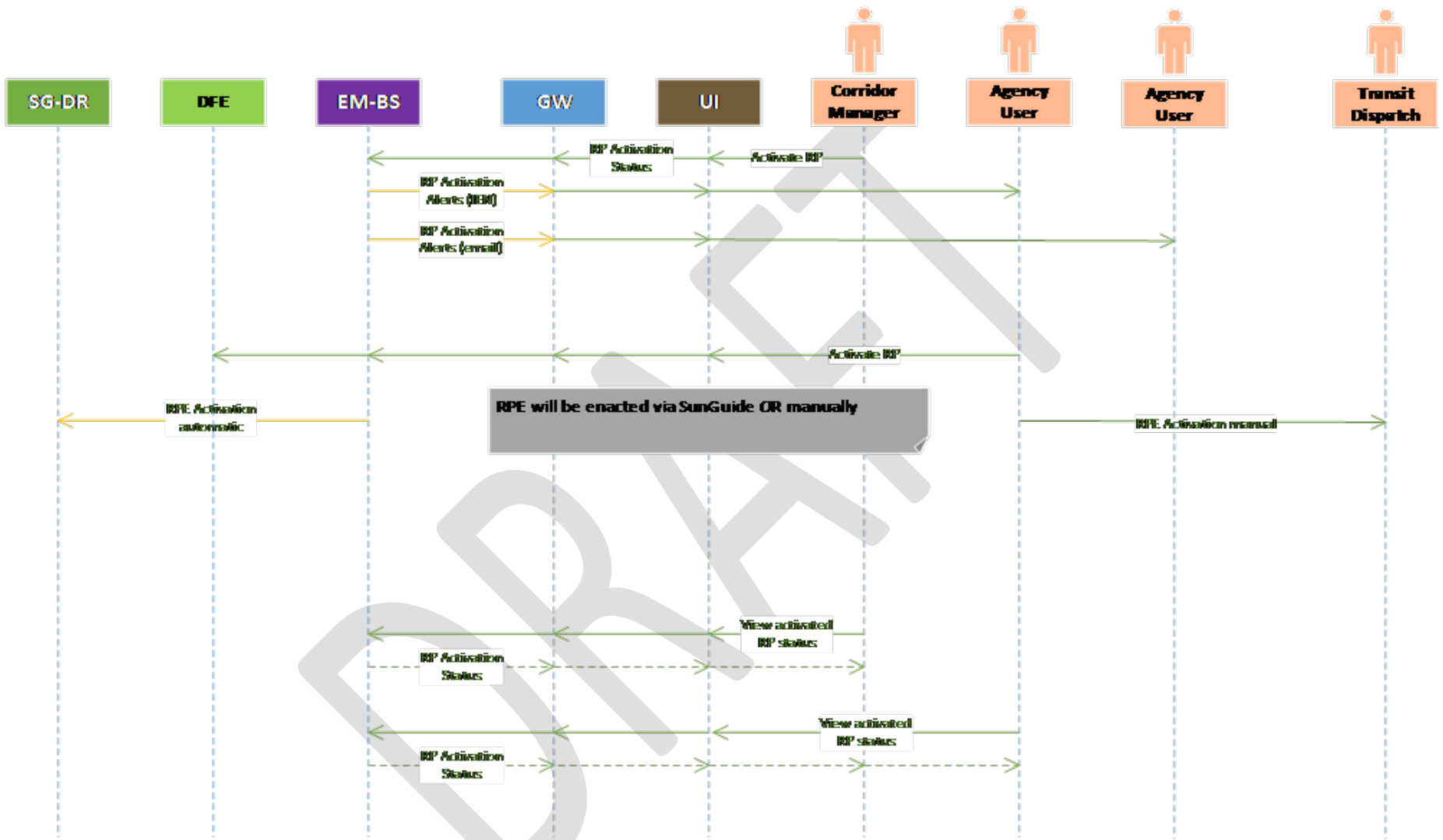


Figure 128 – Response Plan Activation Sequence Diagram

3.6.3.5 Response Plan Monitor/Reevaluation

Figure 129 – Response Plan Monitor/Reevaluation Sequence Diagram shows the continuous re-evaluation of activated response plans to show the Corridor Manager how effective the activated plan is and whether normal operations should resume.

DRAFT

System Design Document for Regional Integrated Corridor Management System (R-ICMS)

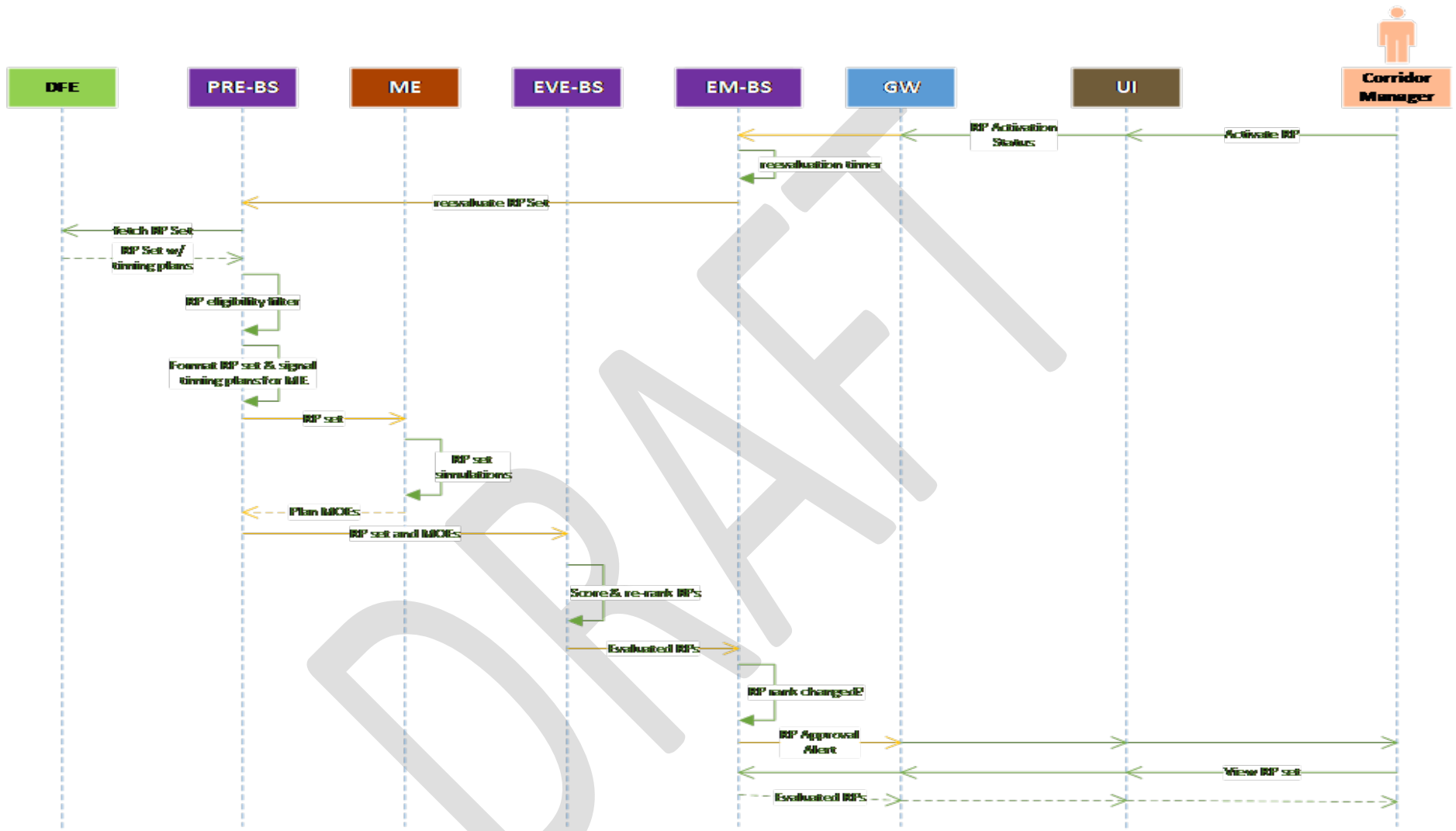


Figure 129 – Response Plan Monitor/Reevaluation Sequence Diagram

3.6.4 Signal Optimization Timing Diagrams

The R-ICMS will provide an offline Signal Optimization Timing Process Service which will allow users to optimize timing patterns on traffic signals within pre-defined corridors. The diagrams in the following subsections depict the ability for users to initiate user defined optimizations, perform periodic optimizations, and to modify the optimized signal timing plans.

3.6.4.1 Periodic Optimization

Figure 130 – SOT - Periodic Optimization Sequence Diagram shows the automatic, periodic background signal optimization process. The process is performed for all corridors on a periodic basis, and each corridor may be configured individually. Once configured, the SOT business service (SOT-BS) periodically clusters traffic demand for corridors into time ranges based on intersection movement counts, fetches the timing plans for the resultant cluster, and interfaces with an HCS7 Streets system to perform timing plan optimizations. The SOT-BS uses the Predictive Engine business service (PRE-BS) to interface with the modeling engine to simulate the implementation of optimized timing plans along with the normal time-of-day timing plans and passes them to the Evaluation Engine business service (EVE-BS) for scoring and ranking to predict the effectiveness of optimized plans.

System Design Document for Regional Integrated Corridor Management System (R-ICMS)

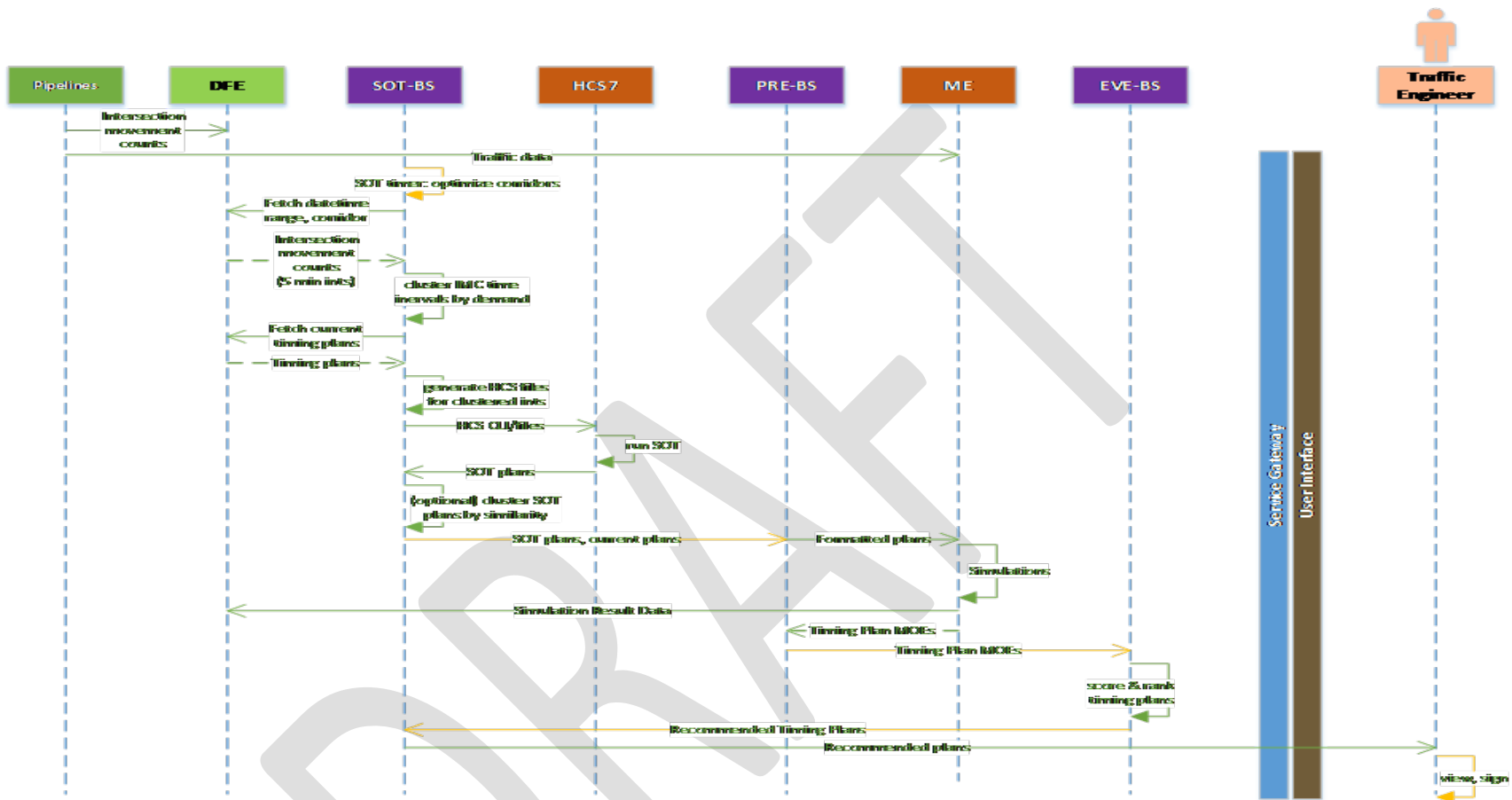


Figure 130 – SOT - Periodic Optimization Sequence Diagram

3.6.4.2 On Demand Optimization

Figure 131 – SOT On Demand Optimization Sequence Diagram shows a user-initiated signal optimization process. Via the UI, a user may select a corridor, a date range of historical intersection count data, and other possible items used by the SOT tool (HCS7 Streets).

DRAFT

System Design Document for Regional Integrated Corridor Management System (R-ICMS)

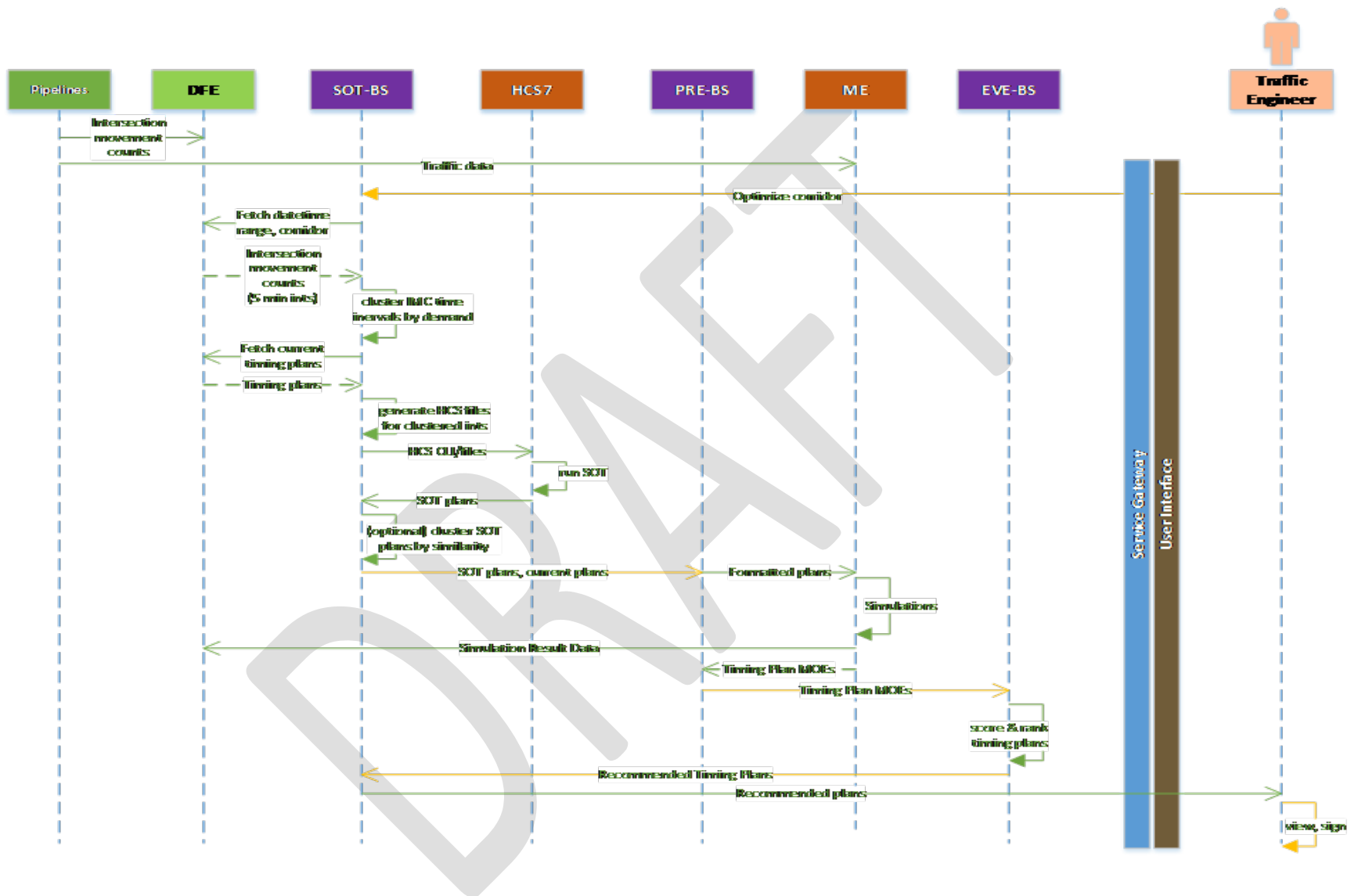


Figure 131 – SOT On Demand Optimization Sequence Diagram

3.6.4.3 Modified Timing Plan

Figure 132 – SOT Modified Timing Plan Sequence Diagram shows the process of manually modifying the offsets of a suggested signal timing plan and re-evaluating the modification.

DRAFT

System Design Document for Regional Integrated Corridor Management System (R-ICMS)

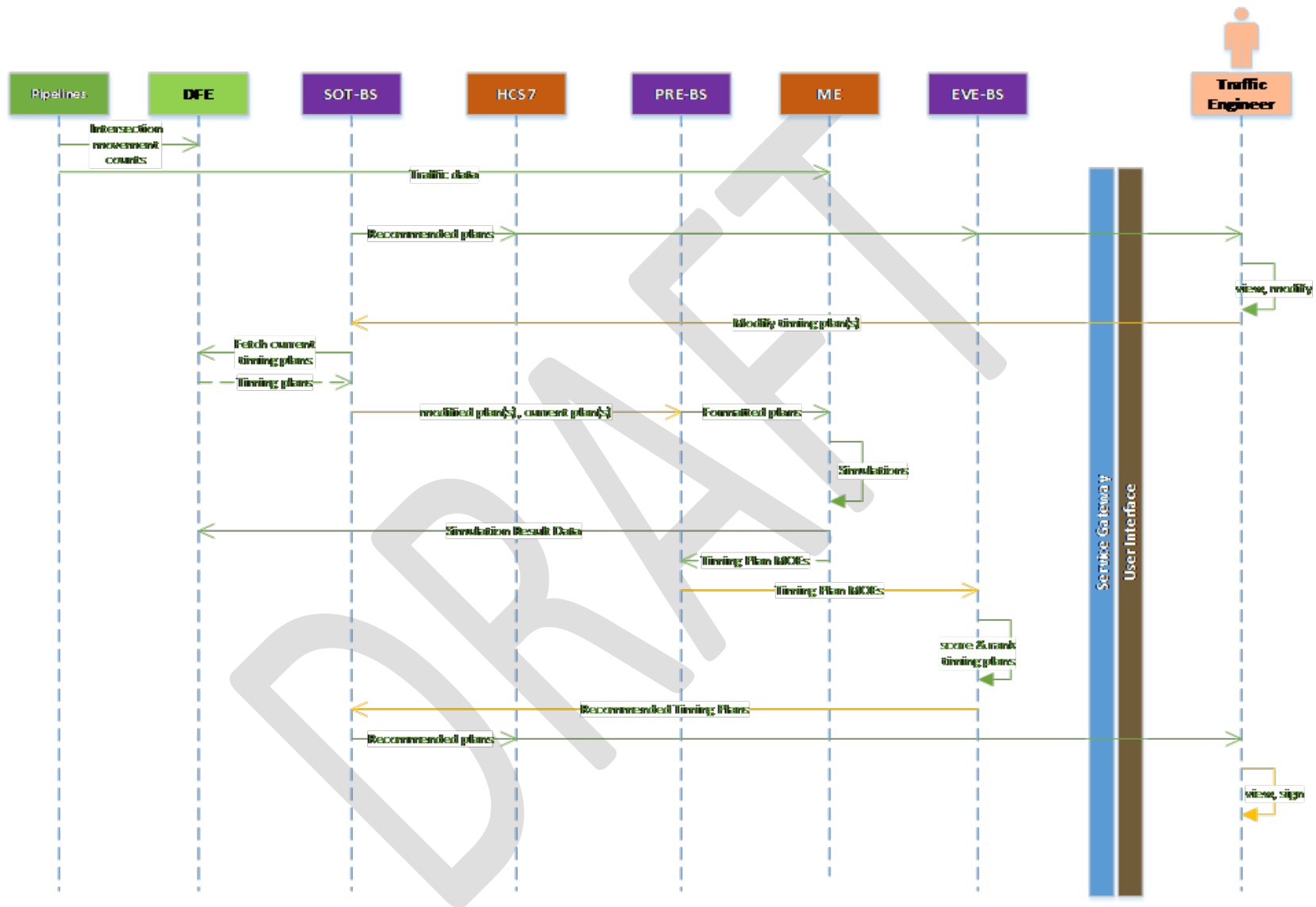


Figure 132 – SOT Modified Timing Plan Sequence Diagram

3.6.4.4 Detailed: Create Corridor

Figure 133 shows the process of creating a corridor based on user configured set of intersections and activation times. Initial intersection features, restrictions, phasing and timing are sourced via data services and pipelines, and user input modifications are captured in the SOT operational database.

DRAFT

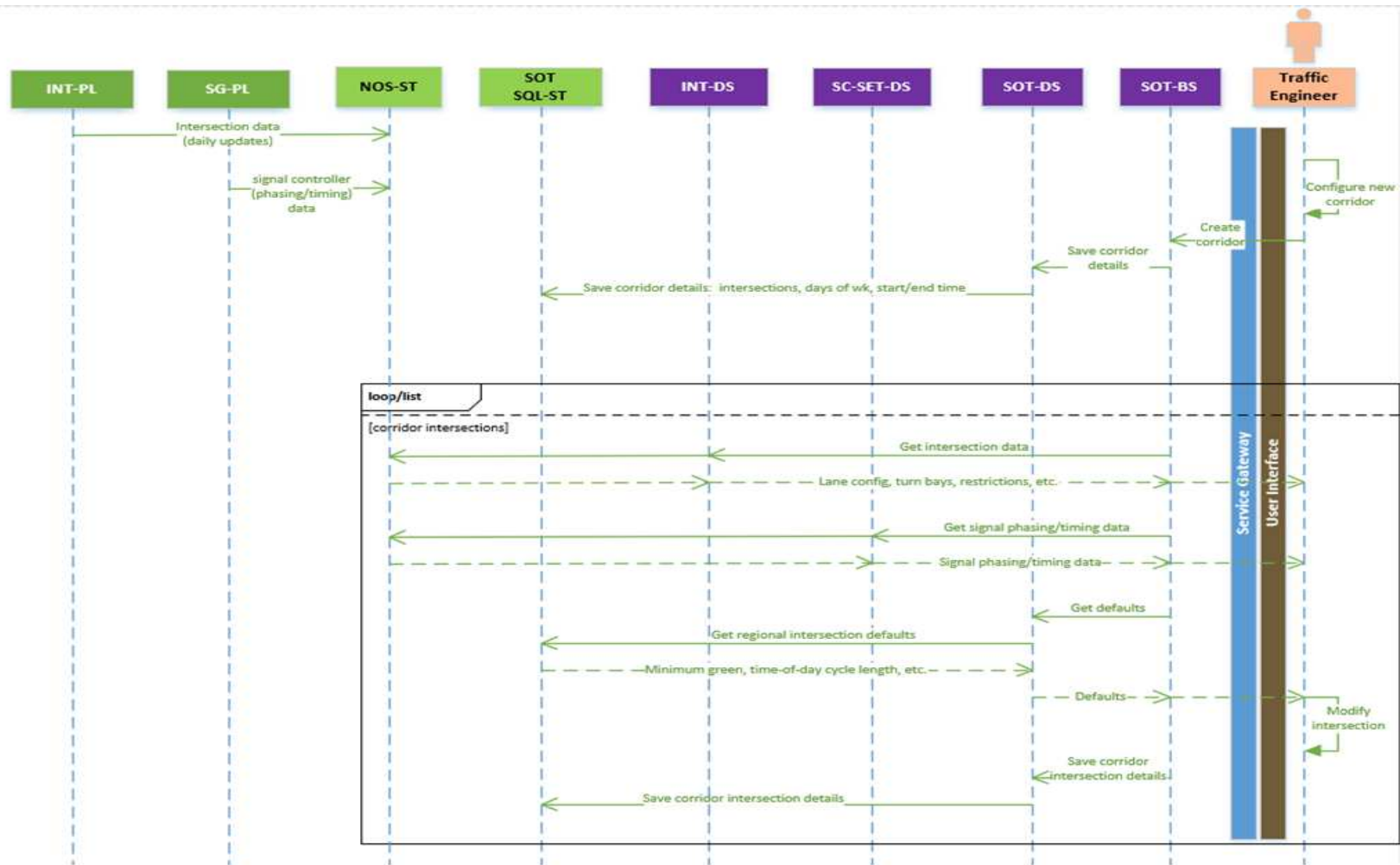


Figure 133 – SOT Detailed Create Corridor Sequence Diagram

3.6.4.5 Detailed: Intersection Features

Figure 134 shows interaction between backing data sources, pipelines, and stores used to provide intersection features to the SOT business service.

DRAFT

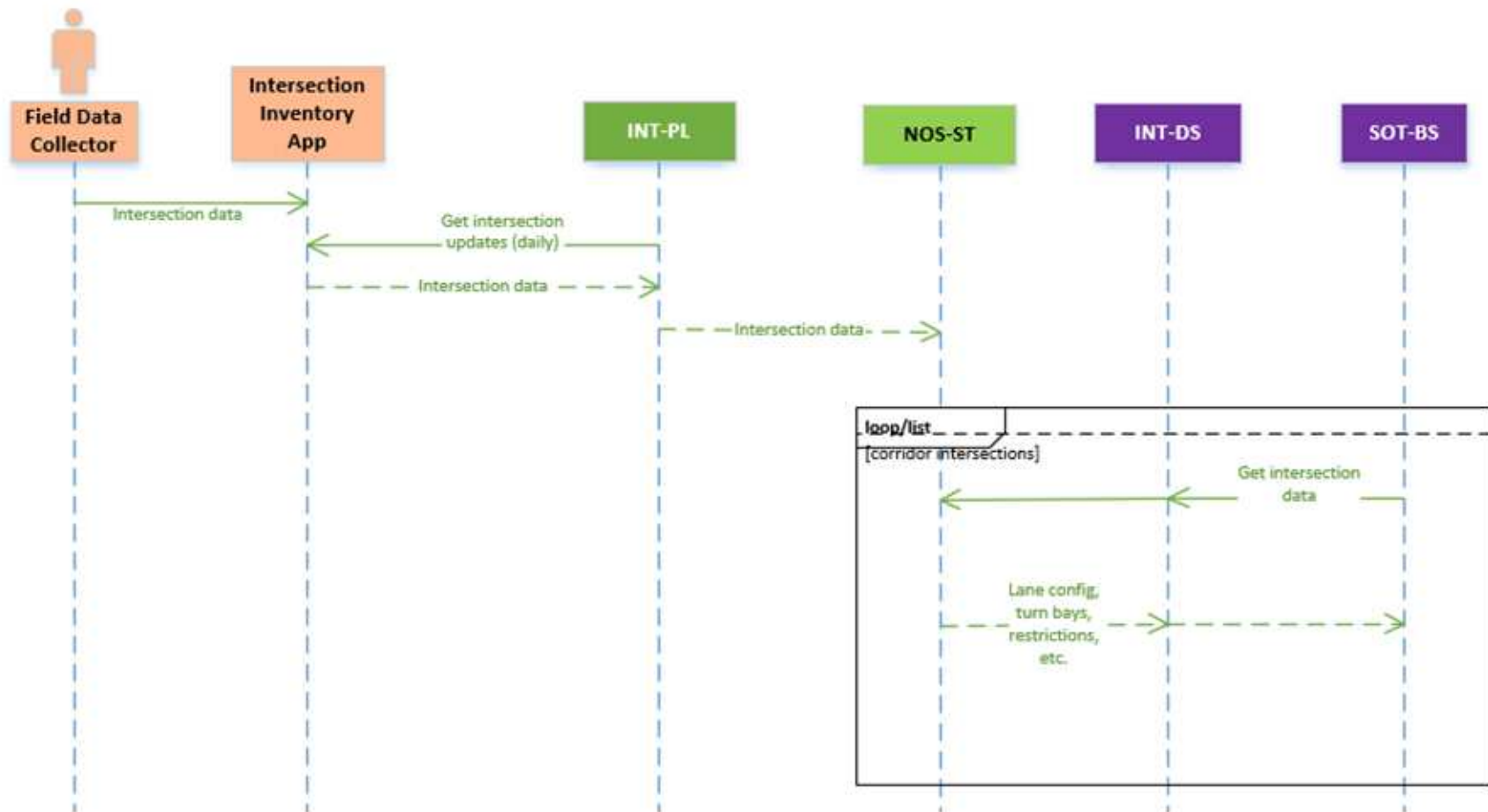


Figure 134 – SOT Detailed Intersection Features Diagram

3.6.4.6 Detailed: Modify Corridor

Figure 135 shows the process of a user modifying a SOT corridor configuration. Note that the corridor and associated intersection details are updated as a unit.

DRAFT

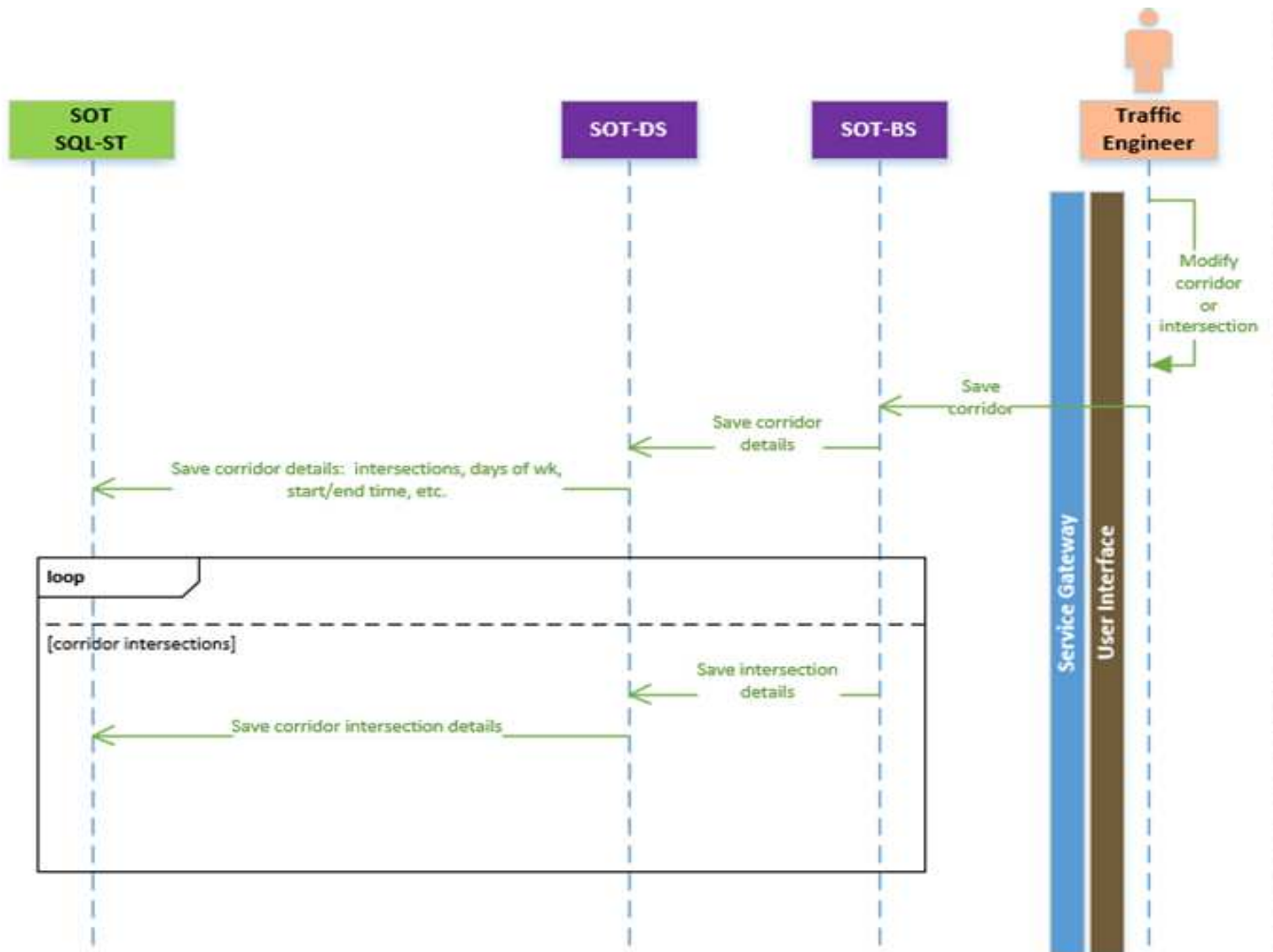


Figure 135 – SOT Detailed Modify Corridor Sequence Diagram

3.6.4.7 Detailed: Corridor Optimization 1

Figure 136 – SOT Detailed Corridor Optimization 1 shows the SOT corridor optimization process.

DRAFT

System Design Document for Regional Integrated Corridor Management System (R-ICMS)

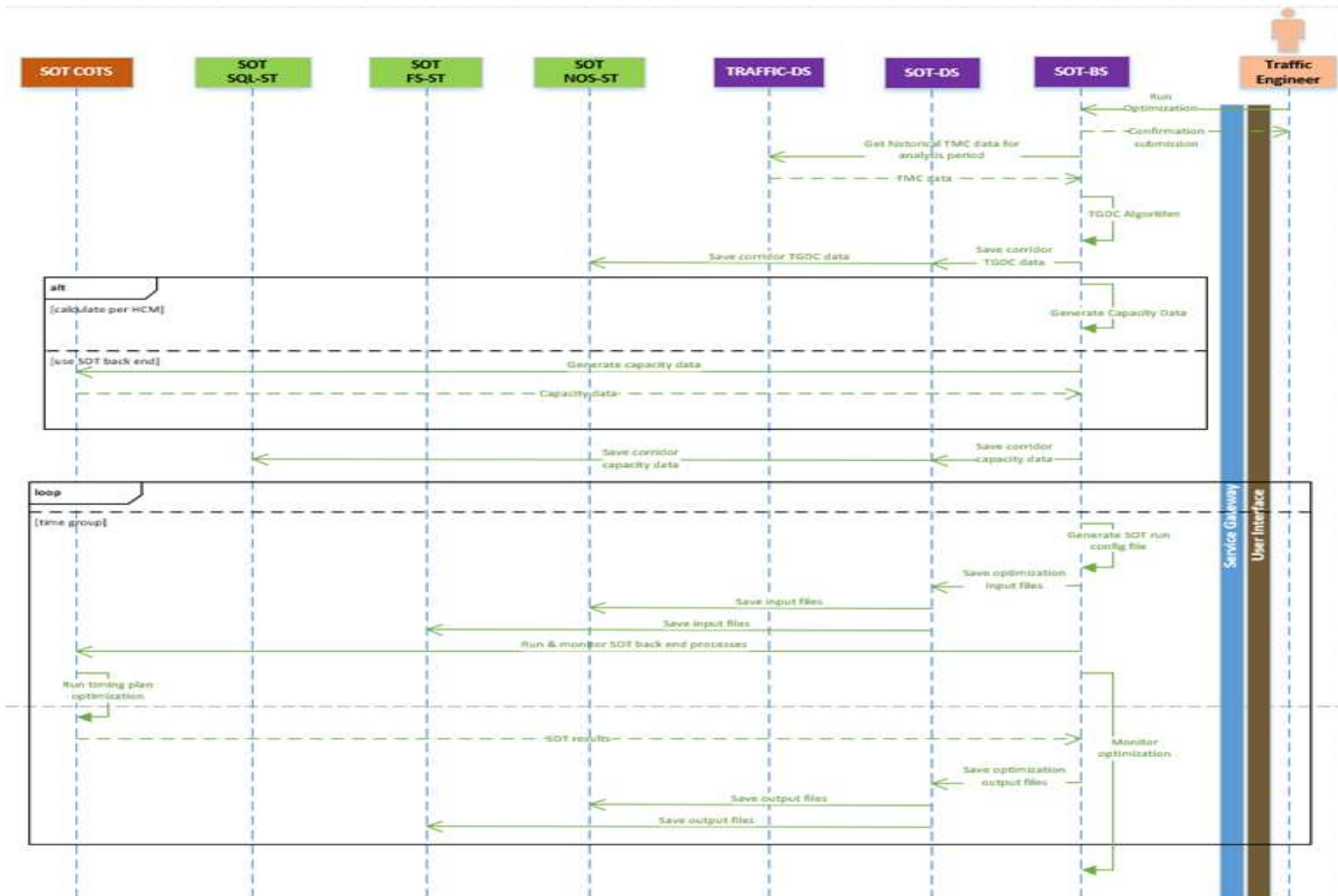


Figure 136 – SOT Detailed Corridor Optimization 1

3.6.4.8 Detailed: Corridor Optimization 2

Figure 137, a continuation of Figure 136, shows the extraction of timing plans from the output of a corridor optimization and user notification of the prepared results via the Alert business service

DRAFT

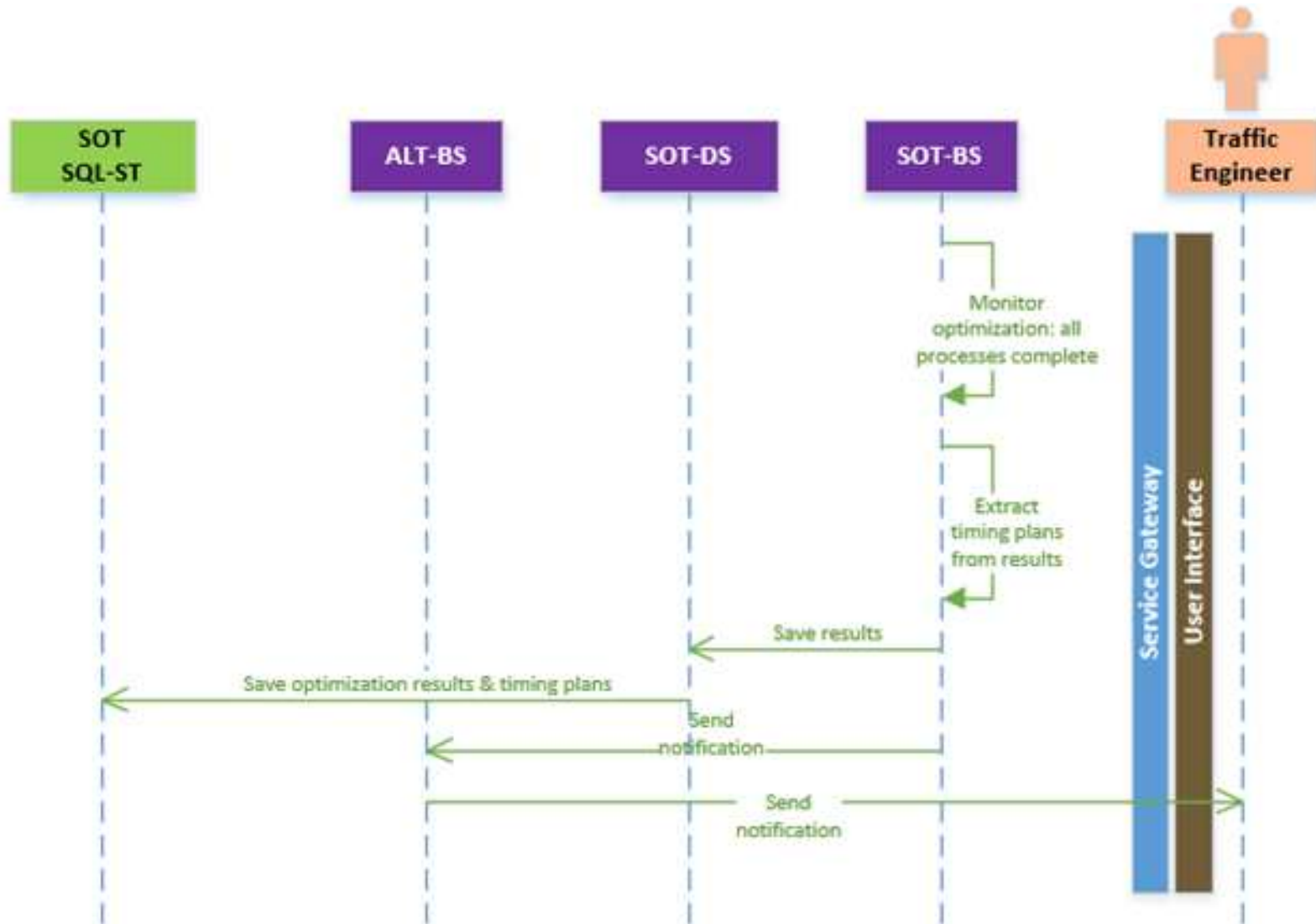


Figure 137– SOT Detailed Corridor Optimization 2

3.6.4.9 Detailed: Deploy Corridor

Figure 138 illustrates the use cases associated with a user marking a set of corridor timing plans as deployed. Corridor timing plans cannot be deployed without resolving conflicts, such as an intersection in multiple corridors at the same time. Conflicts may be resolved by modifying or deactivating one of the conflicting corridors

DRAFT

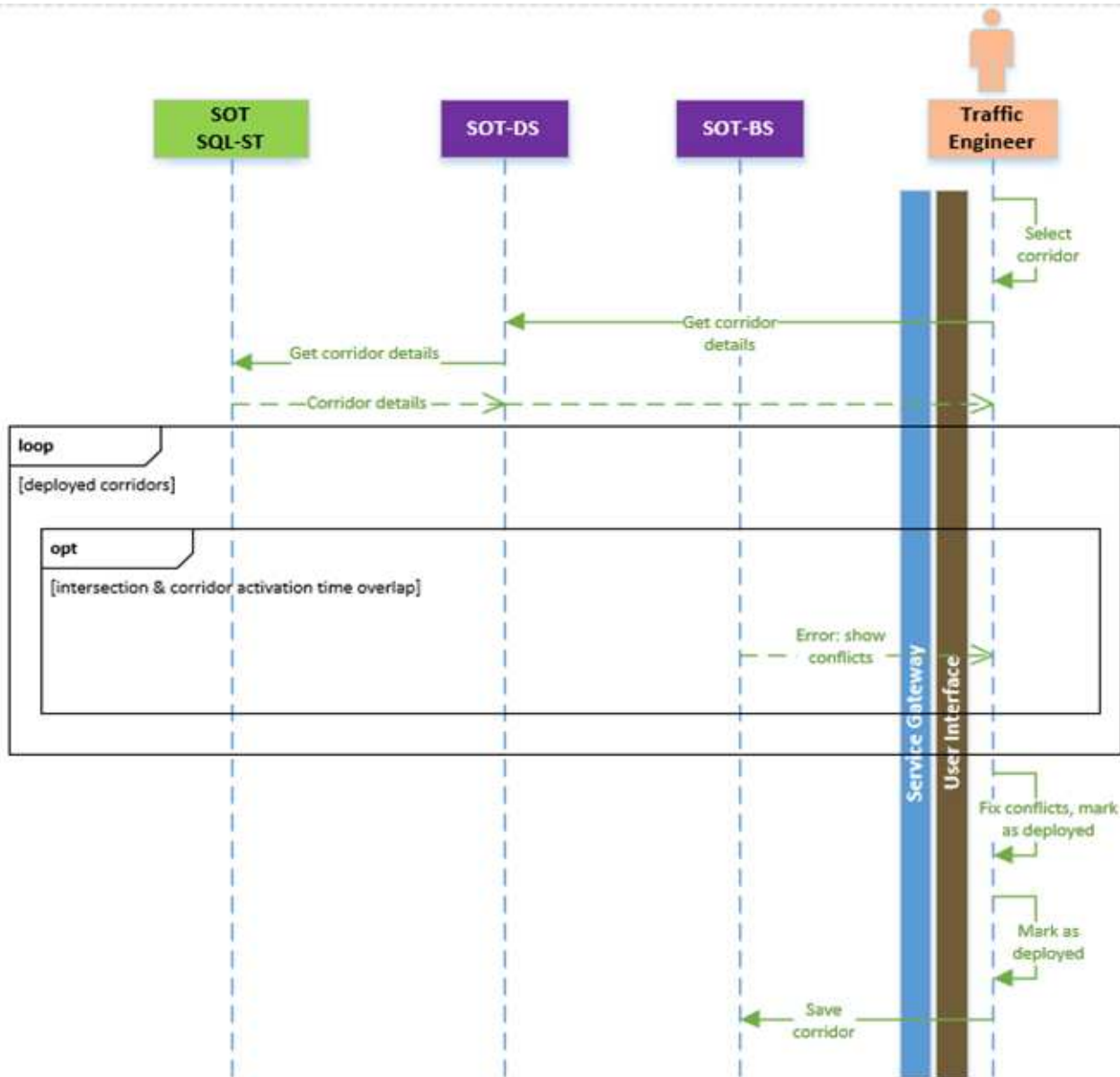


Figure 138 – SOT Detailed Deploy Corridor

3.6.4.10 Detailed: SOT Results Simulation Interface

Figure 102 shows the interaction between the SOT system and the third-party simulation engine. The Simulation Engine Interface is a REST API implemented by the third party to allow RICMS services to initiate simulation runs. The Simulation Callback Interface is a RES API implemented by the RICMS team to allow the third-party to send results to the RICMS. The Simulation Data Service provide stored simulation data to other RICMS services. These interfaces are used by the SOT business service after optimization a corridor timing plan set to run two simulations, one for the optimized timing plans and one for the exiting timing plans

DRAFT

System Design Document for Regional Integrated Corridor Management System (R-ICMS)

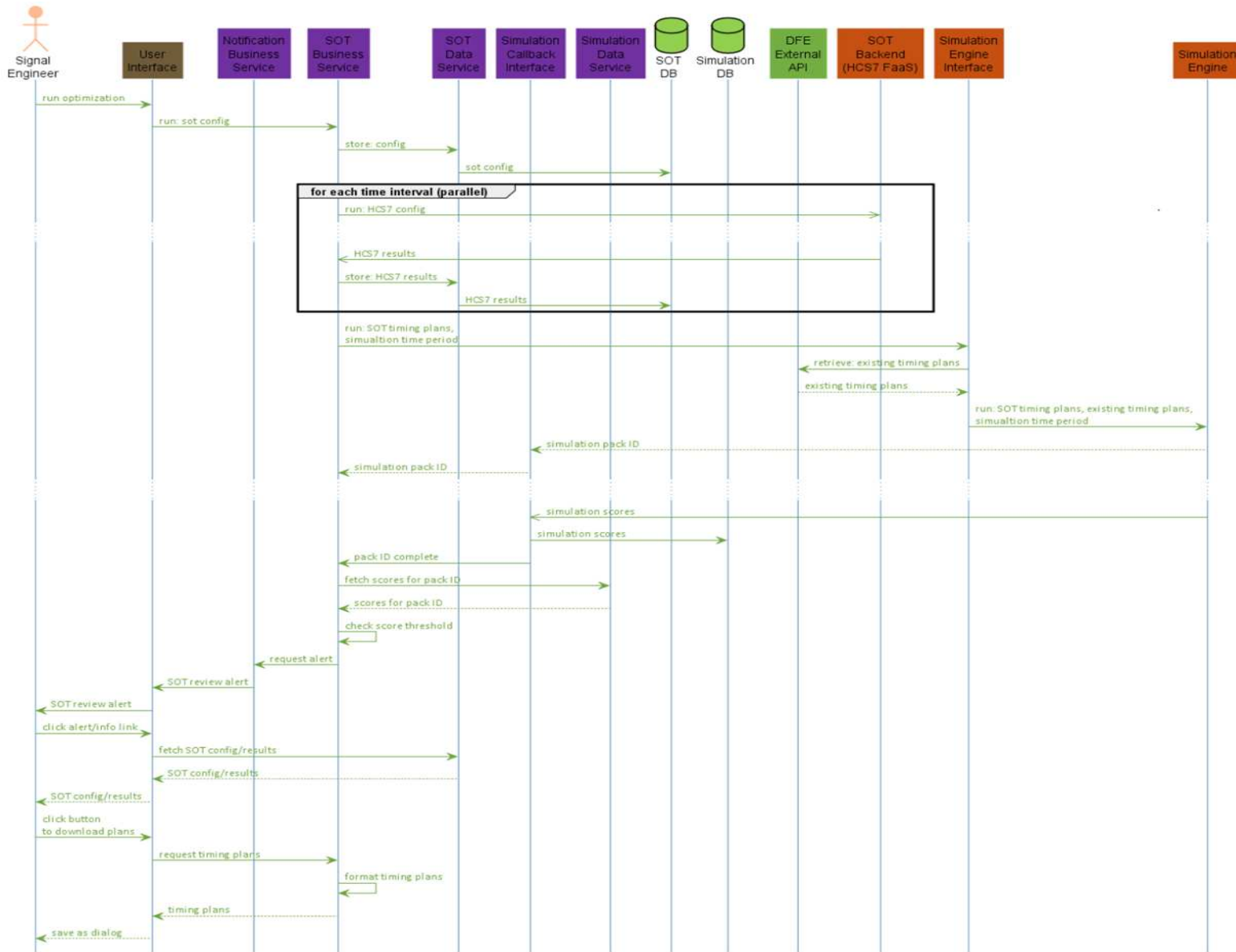


Figure 139 – SOT Detailed Results Simulation Interface

3.6.5 Other Use Case Diagrams

Various other functionalities will be needed and included as part of the R-ICMS. The following diagrams depict these functionalities including external access to the DFE, alerting functionality, bulk loading, system monitoring and logging.

3.6.5.1 Other: External Access – Authentication

Figure 140 shows the sequence for logging in external users. External users log in the same way as a regular user, but do so without the aid of the UI. They ultimately authenticate against Active Directory/LDAP and must have permissions assigned to them within the R-ICMS application. They receive a token to be used with all requests for data.

DRAFT

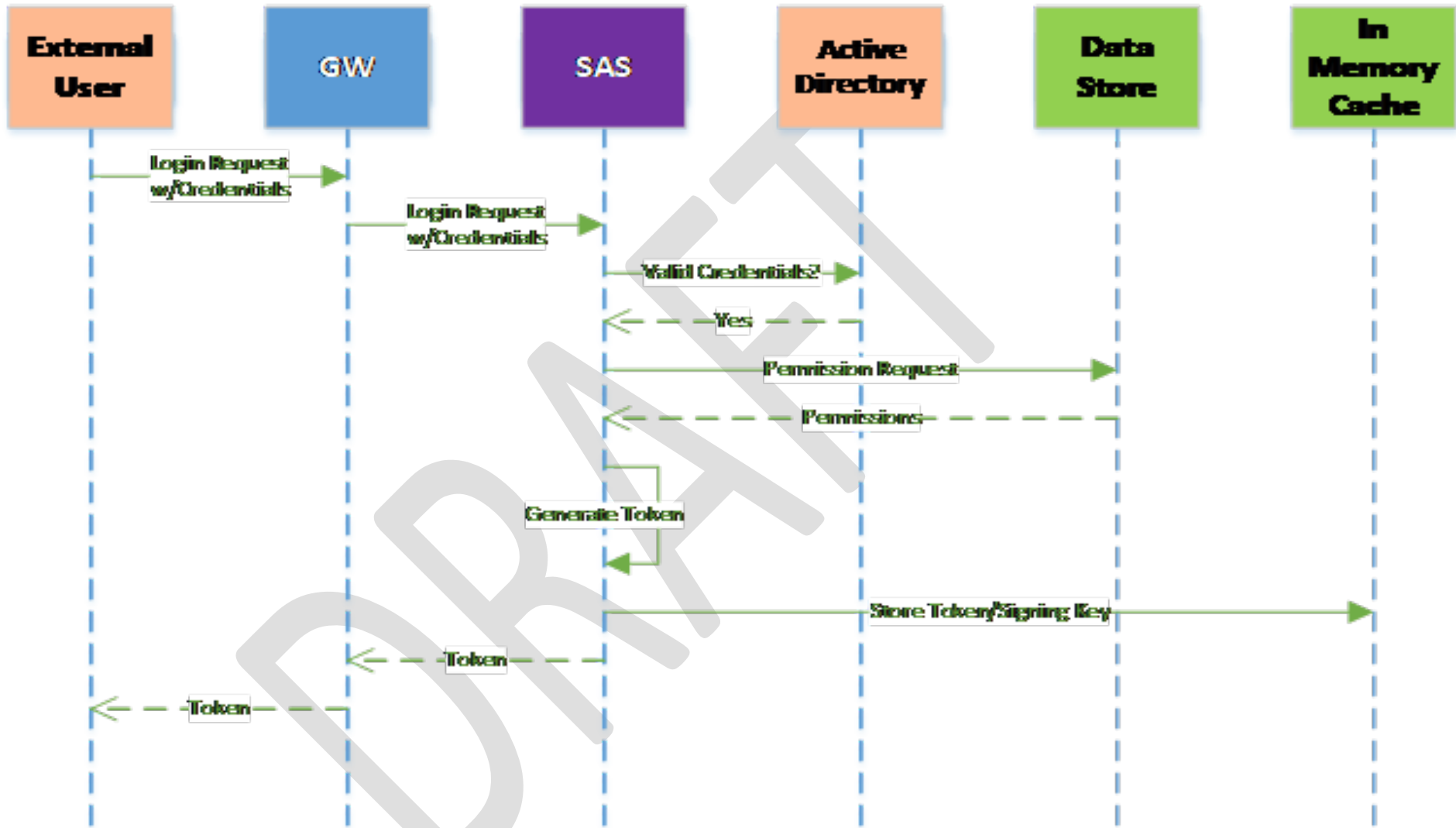


Figure 140– External Access - Authentication Sequence Diagram

3.6.5.2 Other: External Access – Data Request

Figure 141 shows the sequence used to satisfy external data requests. External users make data requests directly to the gateway without the aid of the UI. As with regular users, the token is passed in and authorization performed (not shown here).

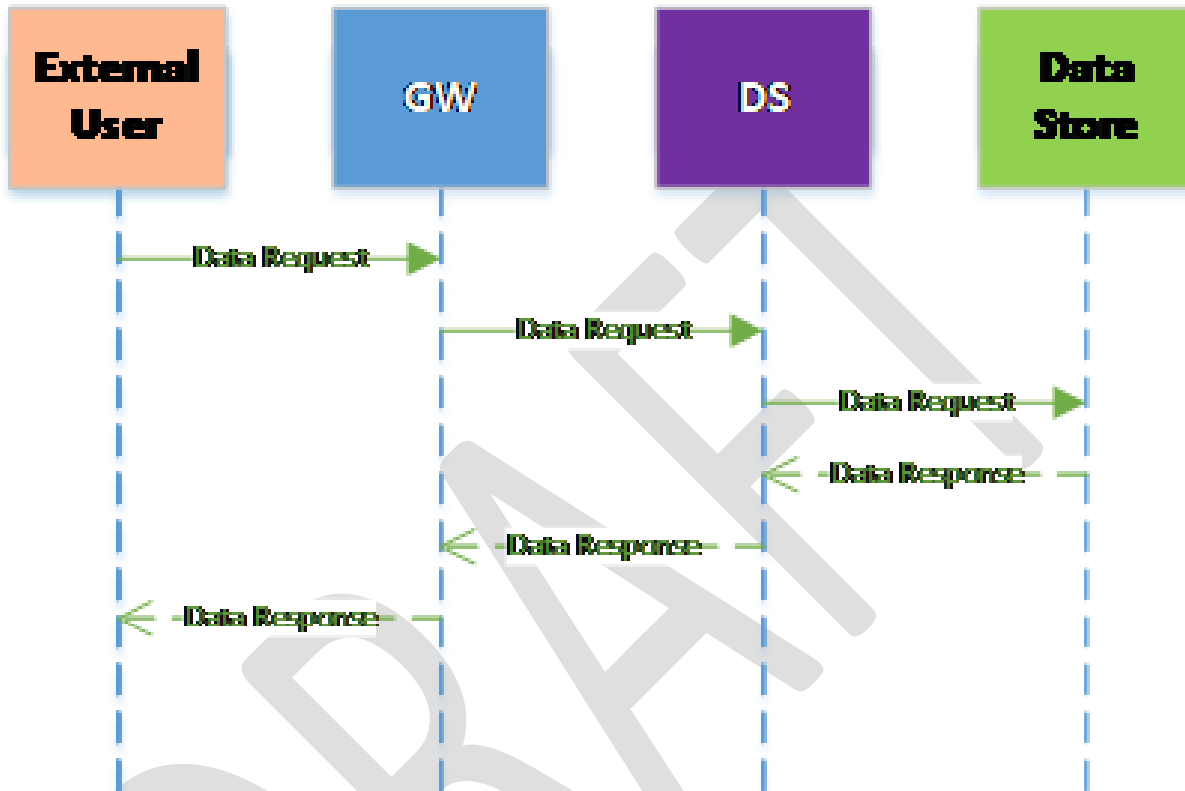


Figure 141 – External Access – Data Request Sequence Diagram

3.6.5.3 Other: External Access – Streaming

Figure 140 shows the activities involved in accessing streaming data by an external user. The process of streaming data to an external user is similar to streaming to a regular user except that the request is made directly to the gateway. A subscription is requested and established, then updates are flowed down to the user as they come in.

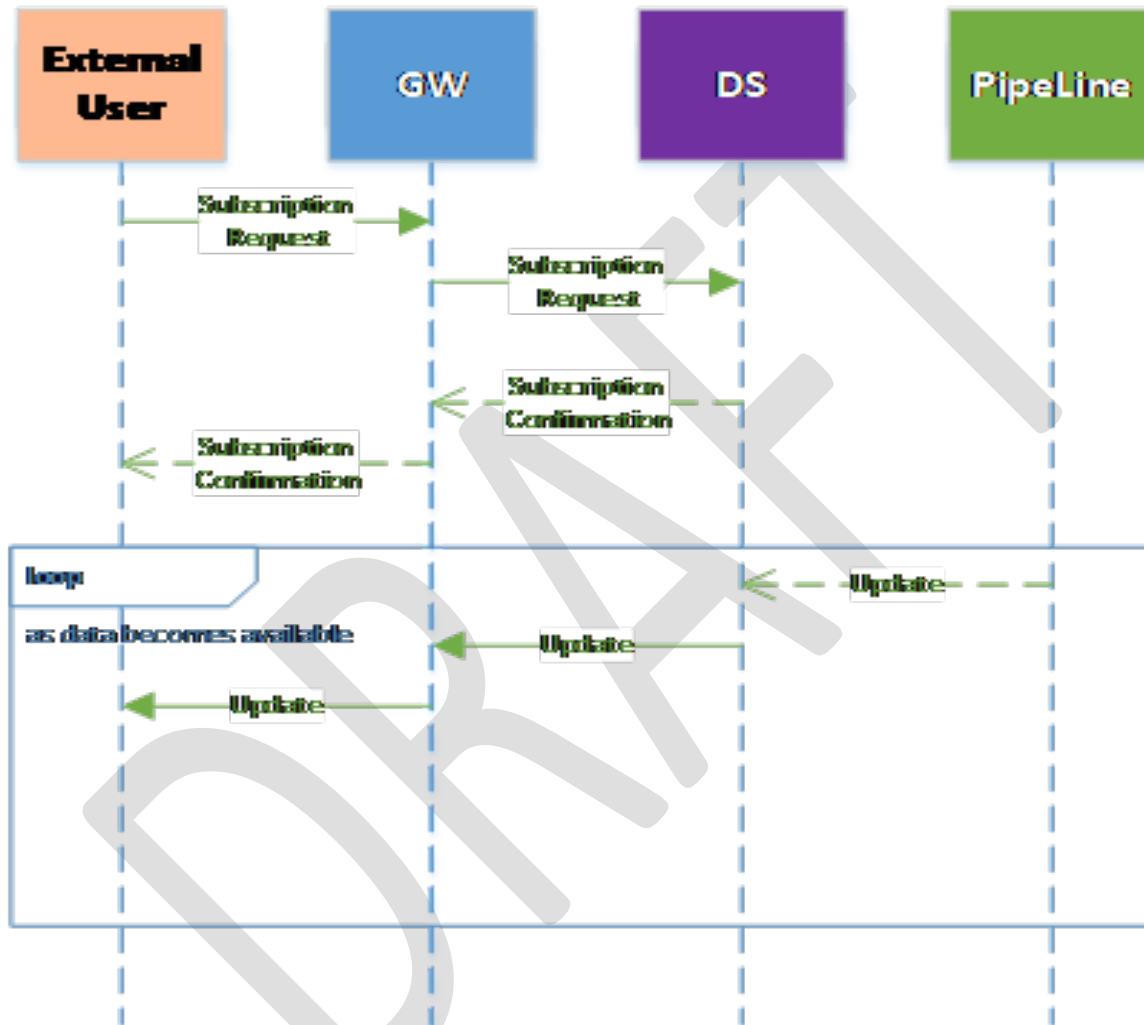


Figure 142 – External Access – Streaming Sequence Diagram

3.6.5.4 Other: External Access – Metadata

Figure 143 shows the activities that occur when an external user requests metadata. A request for metadata is a plain data request that is made to the Metadata Data Service.

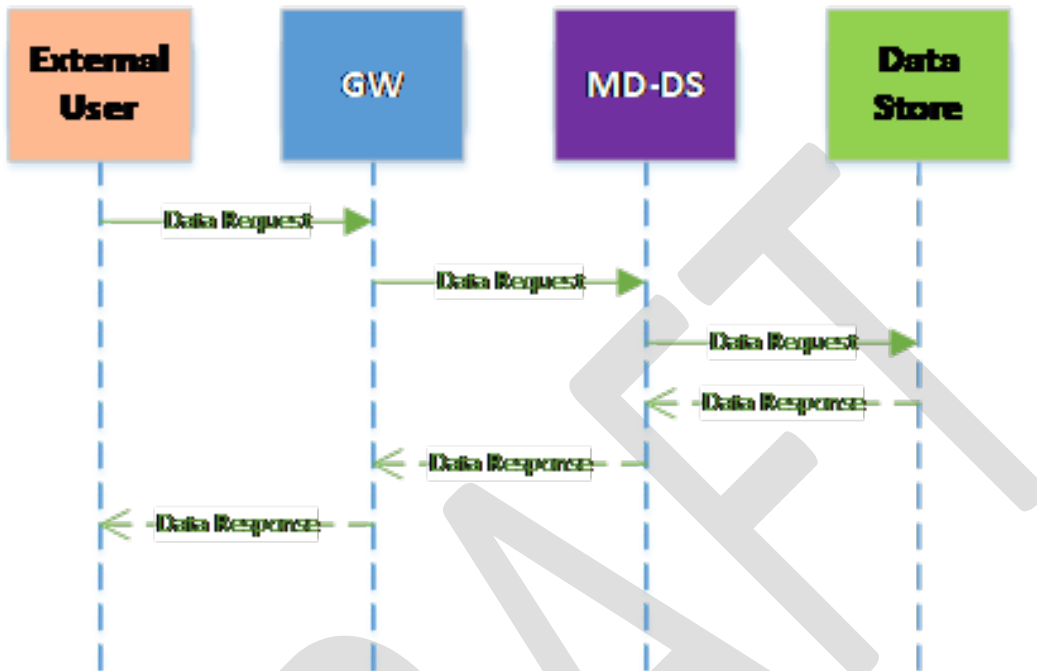


Figure 143 – External Access – Metadata Sequence Diagram

3.6.5.5 Other: Parking

Figure 144 shows the sequence of activities related to updating parking information. Pushing parking updates is similar to other update mechanisms in that an external system provides the data and it finds its way through the driver and pipelines to the relevant services, which then surface the changes via a push to an established subscription.

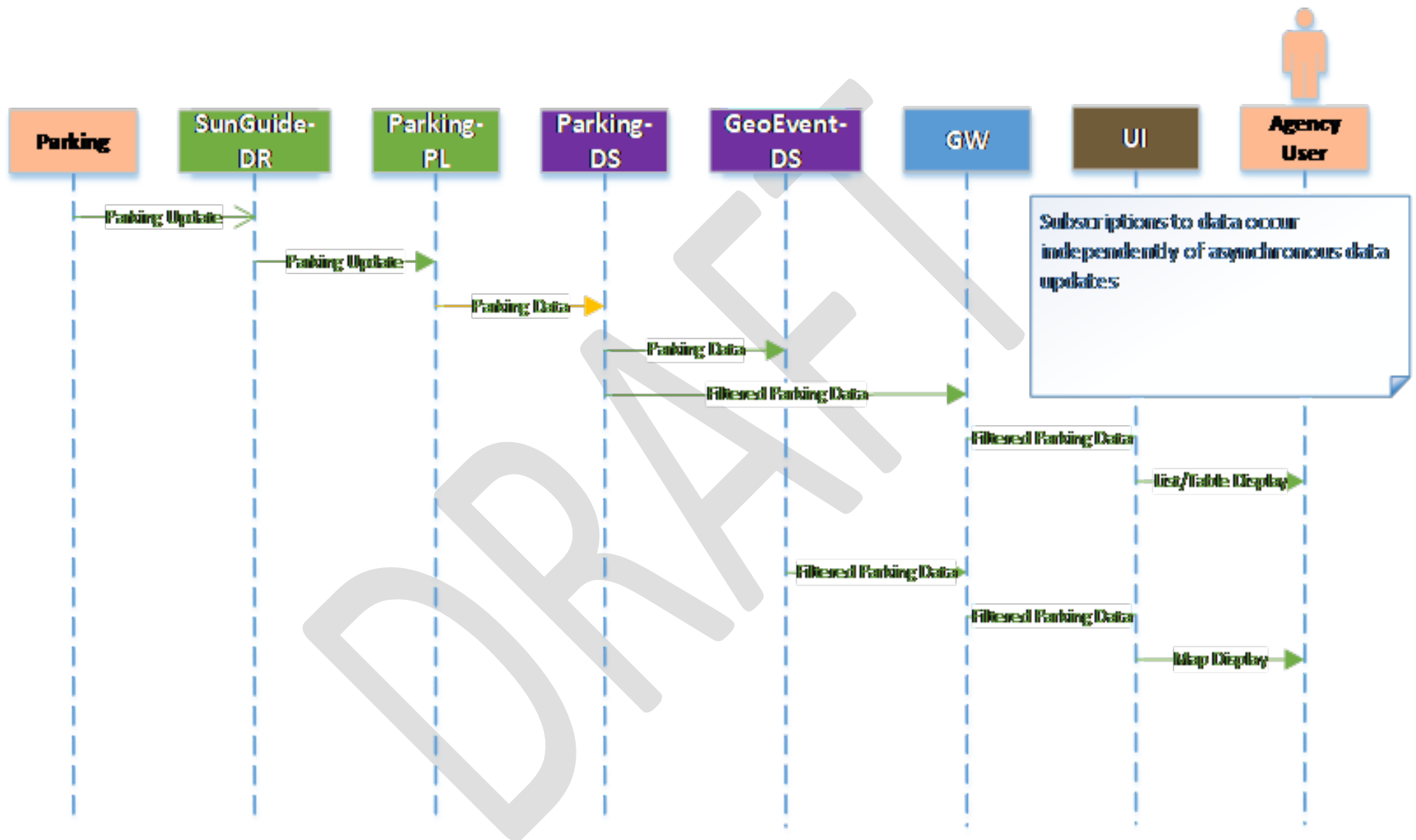


Figure 144– Parking Sequence Diagram

3.6.5.6 Other: Alerts, Information, and Notification

The diagram in Figure 145 is a flow-chart type diagram that shows how the R-ICMS determines who approves RPEs and generates alerts for approval of an RPE using device and agency profiles.

DRAFT

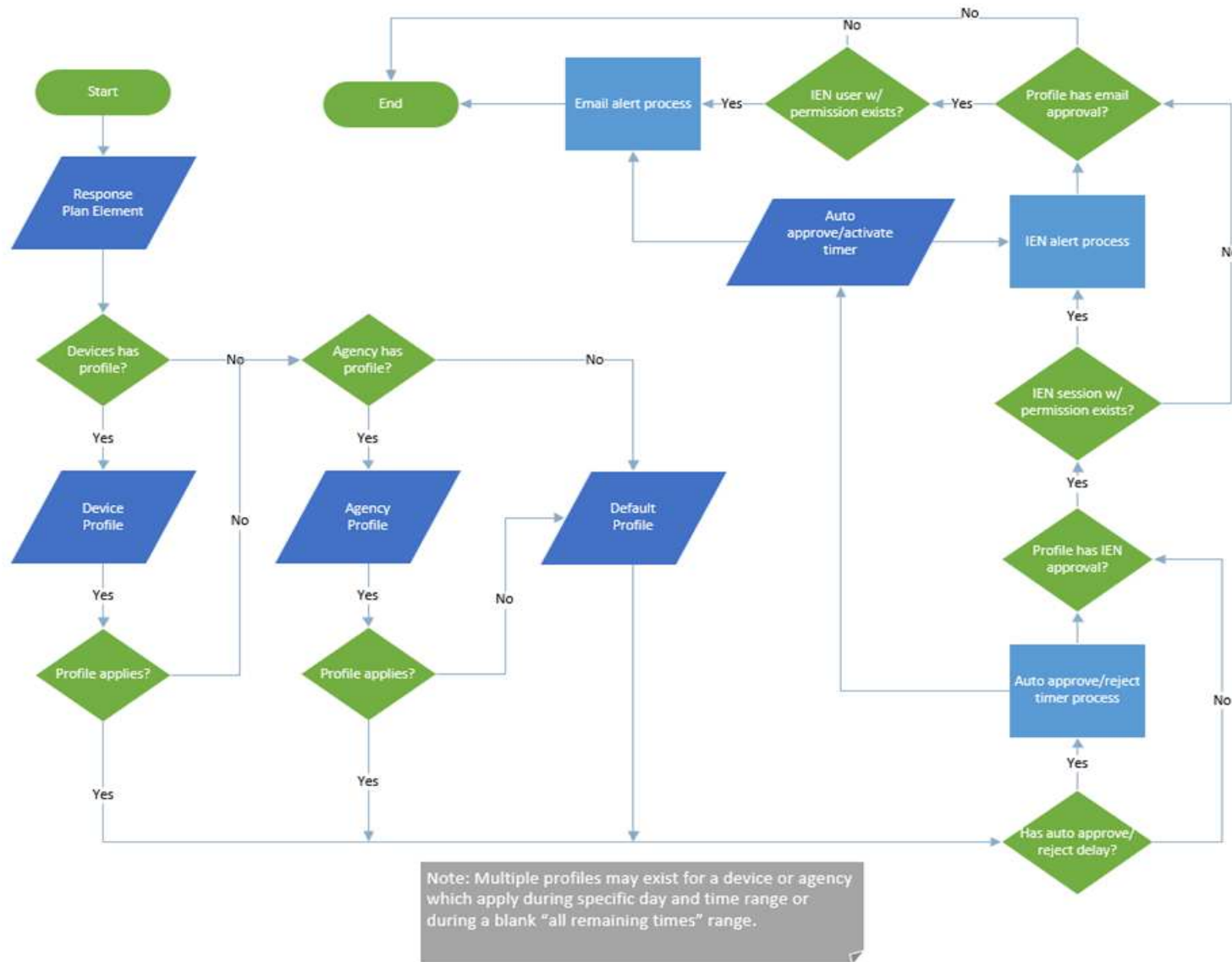


Figure 145– Alerts, Information, and Notification Flow Chart Diagram

3.6.5.7 Other: Alerts, Information, and Notification

The diagram in Figure 146 is a flow-chart type diagram that shows how the R-ICMS determines who approves RPEs and generates alerts for approval of an RPE using device and agency profiles

DRAFT

System Design Document for Regional Integrated Corridor Management System (R-ICMS)

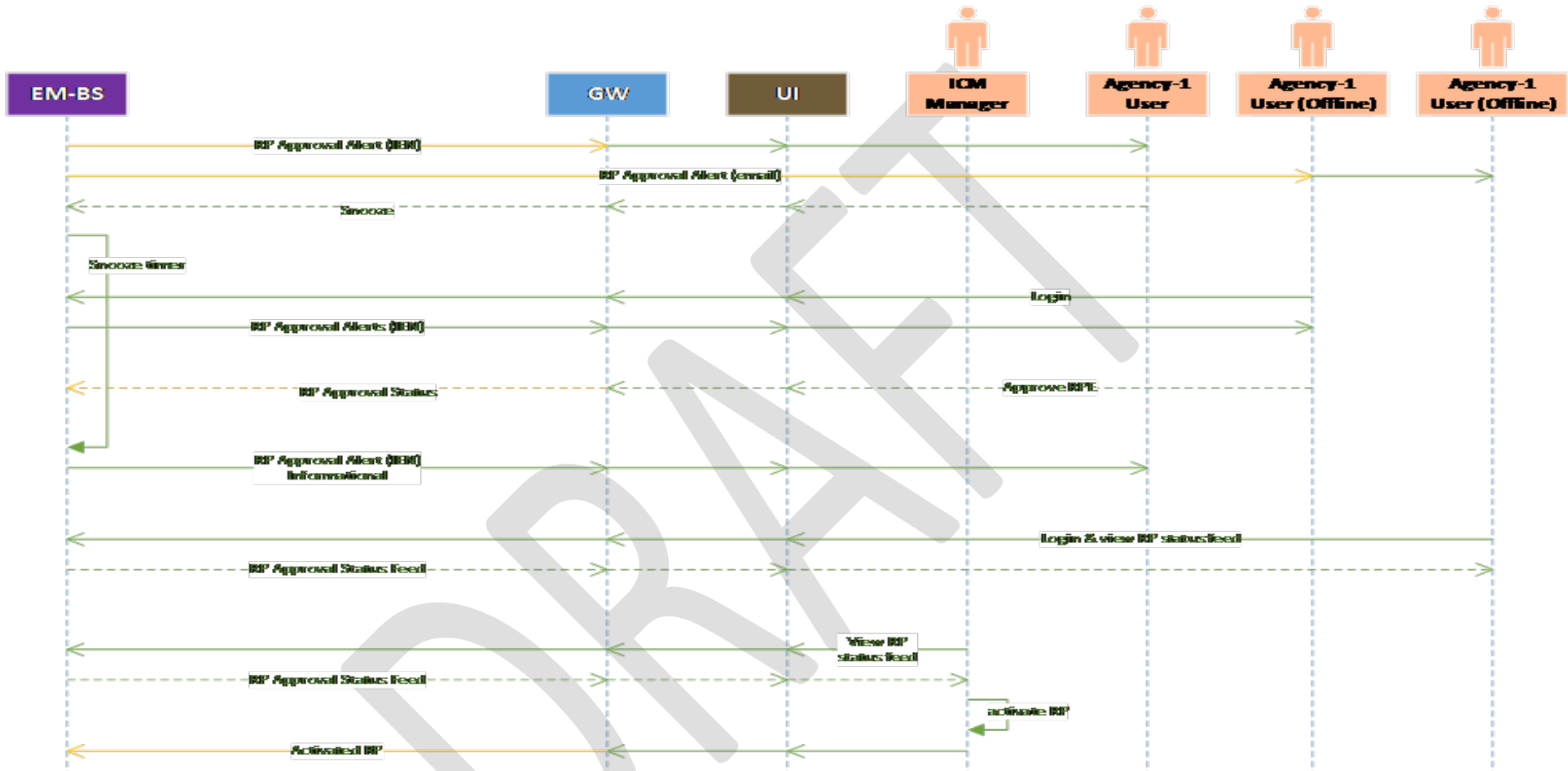


Figure 146– Response Plan Notification Sequence Diagram

3.6.5.8 Other: Data Source Outage Alert

Figure 147 shows a Data Source Outage alert notification process and depicts the following:

- Multiple users may subscribe to alerts for the Data Source Outage
- Users may also subscribe to offline, alerts
- Online alerts are received by IEN, offline alerts are received by email
- Snoozing alerts only affects a single user; a snoozed alert should always reappear

DRAFT

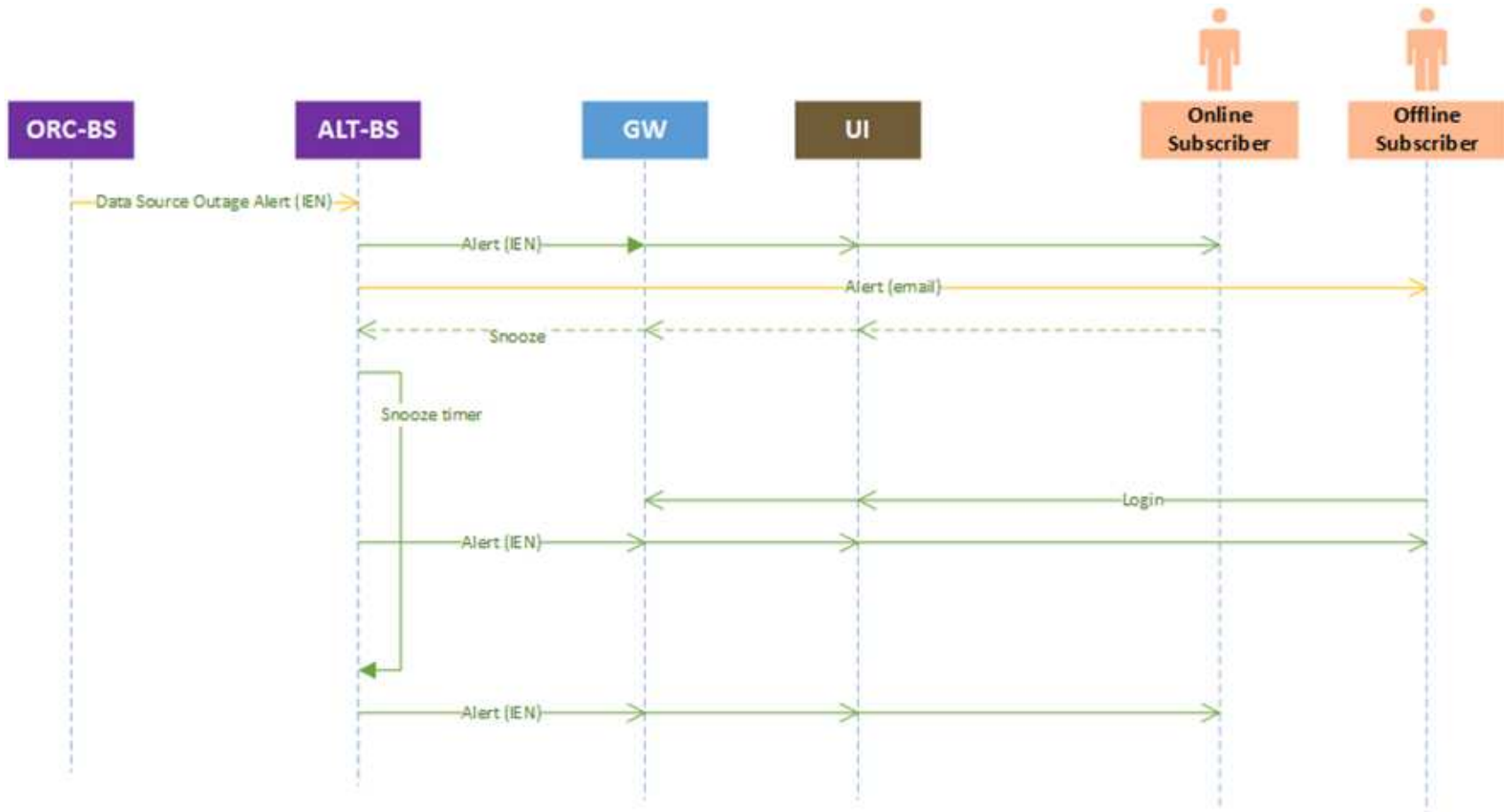


Figure 147 – Data Source Outage Alert Sequence Diagram

3.6.5.9 Other: Logging

Figure 148 shows an authenticated user making a request with examples of when logging occurs and the type of data it should contain. All logs will be saved for analysis and monitoring. Similar logging will be placed in all services.

DRAFT

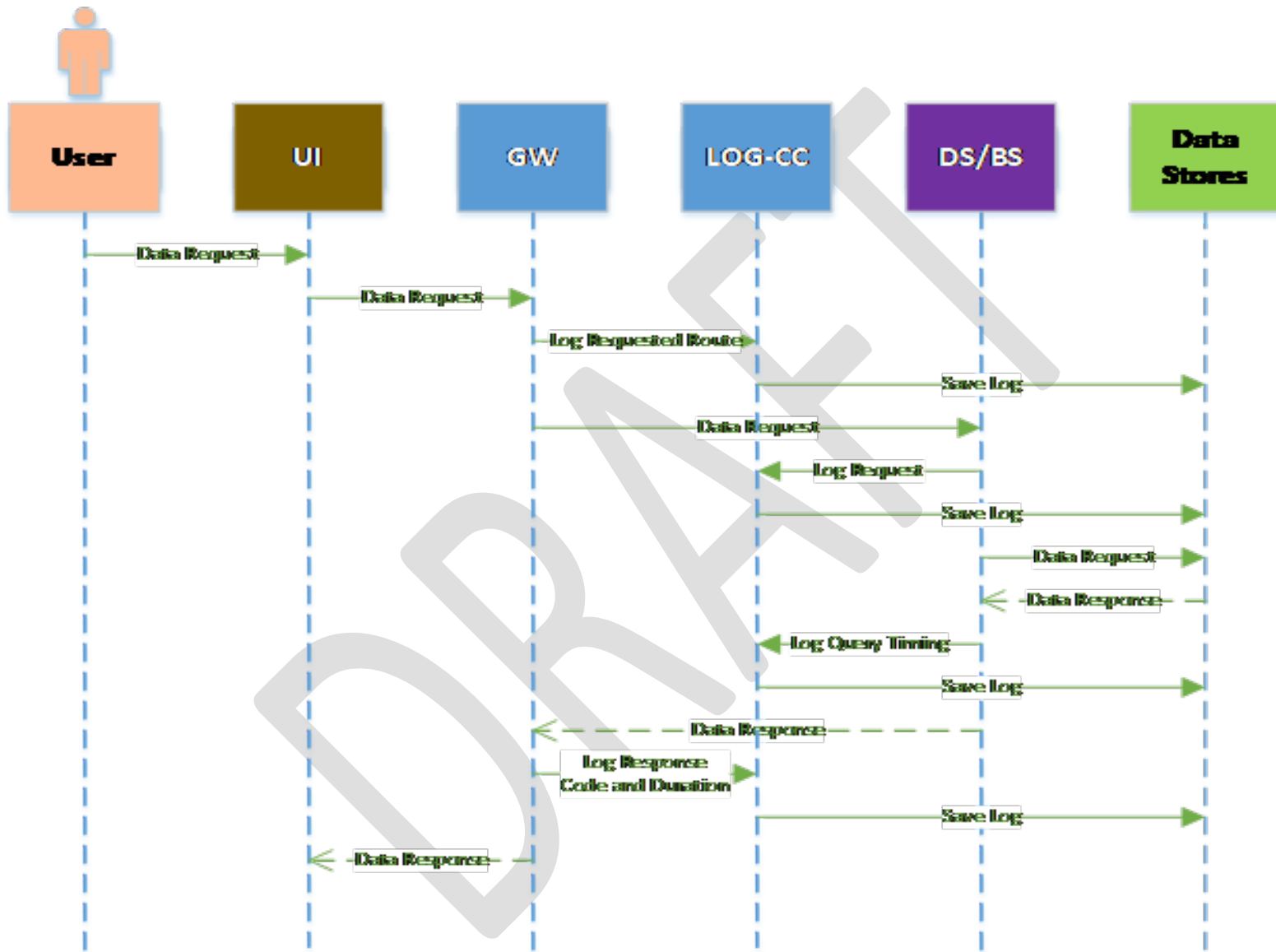


Figure 148– Logging Sequence Diagram

3.6.5.10 Other: Monitoring

Figure 149 shows the activities that occur during system monitoring. The system monitor will periodically check the metrics gathered from logging and determine whether there's an issue with the system. When one is detected, the issue is logged, which saves the issue in a data store, and an alert is triggered.

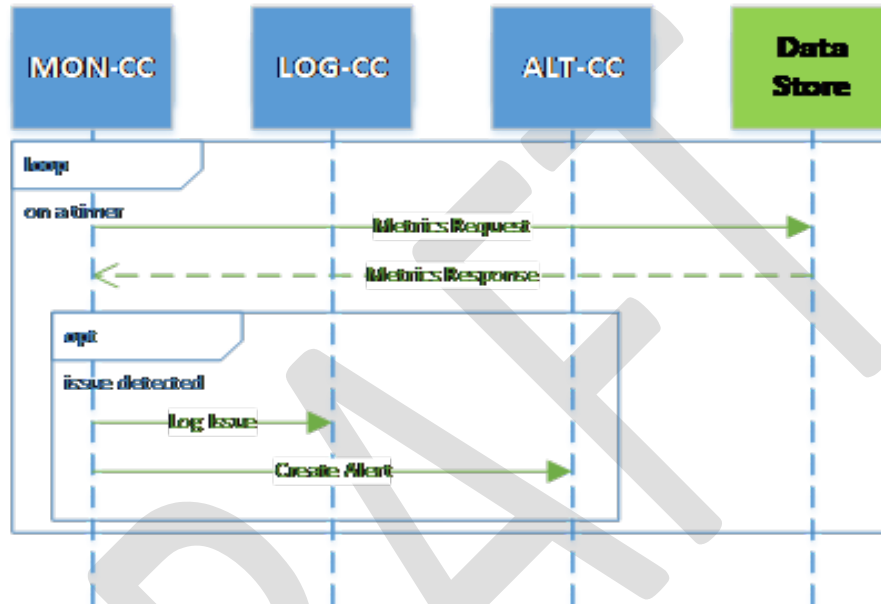
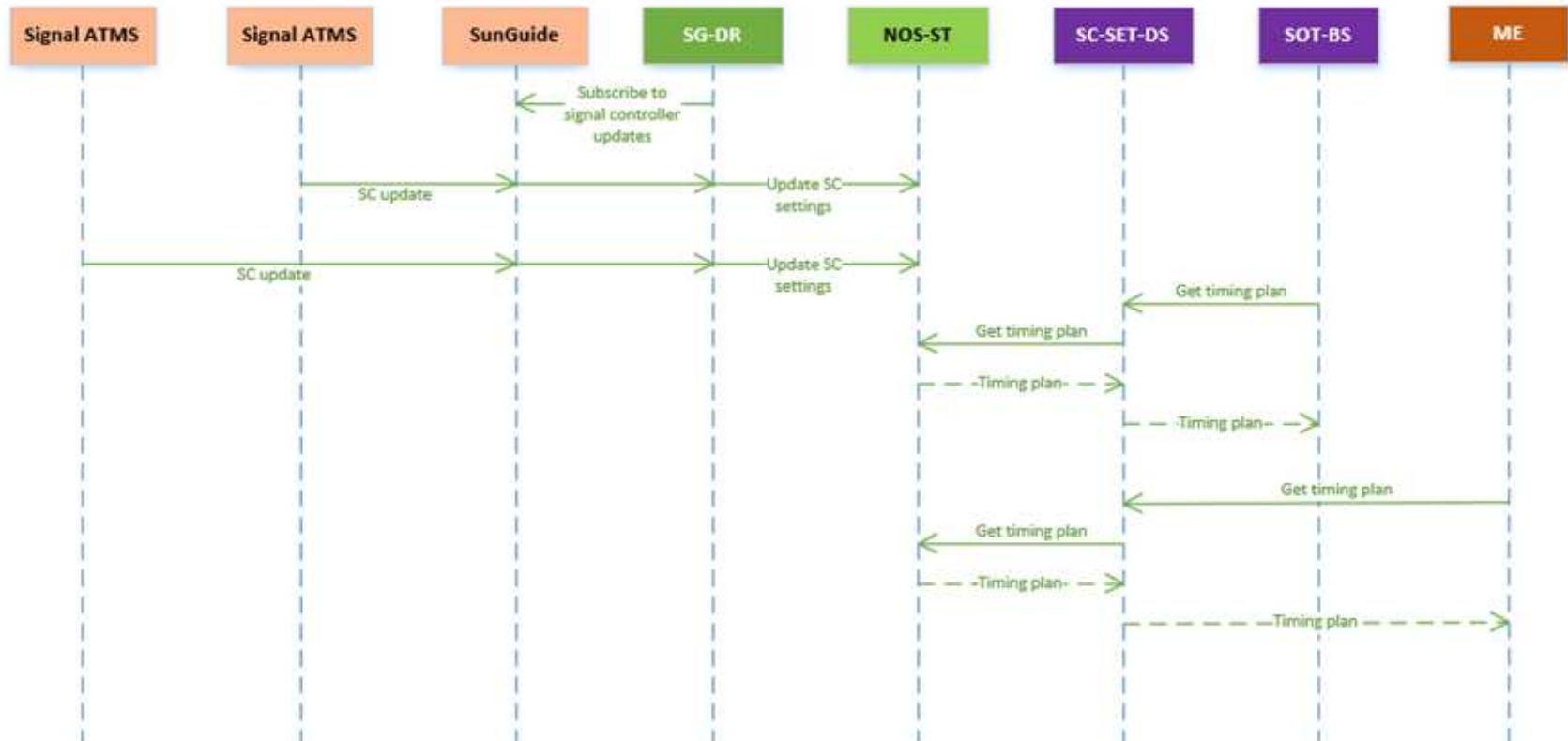


Figure 149– Monitoring Sequence Diagram

3.6.5.11 Signal Controller Settings Data Service

shows that an R-ICMS data service may combine information from multiple backing data stores or pipelines and provide it via a single interface for all internal services and end users. In this example, the SC-SET-DS exposes current signal timing plans, which originate from SunGuide and various signal vendor ATMS interfaces, to the SOT business service and the modeling engine.

System Design Document for Regional Integrated Corridor Management System (R-ICMS)



3.7 System Deployment

The section contains the diagrams and descriptions showing the deployment of physical and virtual components that comprise the R-ICMS system.

3.7.1 Service Host Deployment

Figure 150 shows the interaction between various service components as deployed to physical servers hosting the R-ICMS. Lines between the boxes in Figure 150 indicate that the components interact.

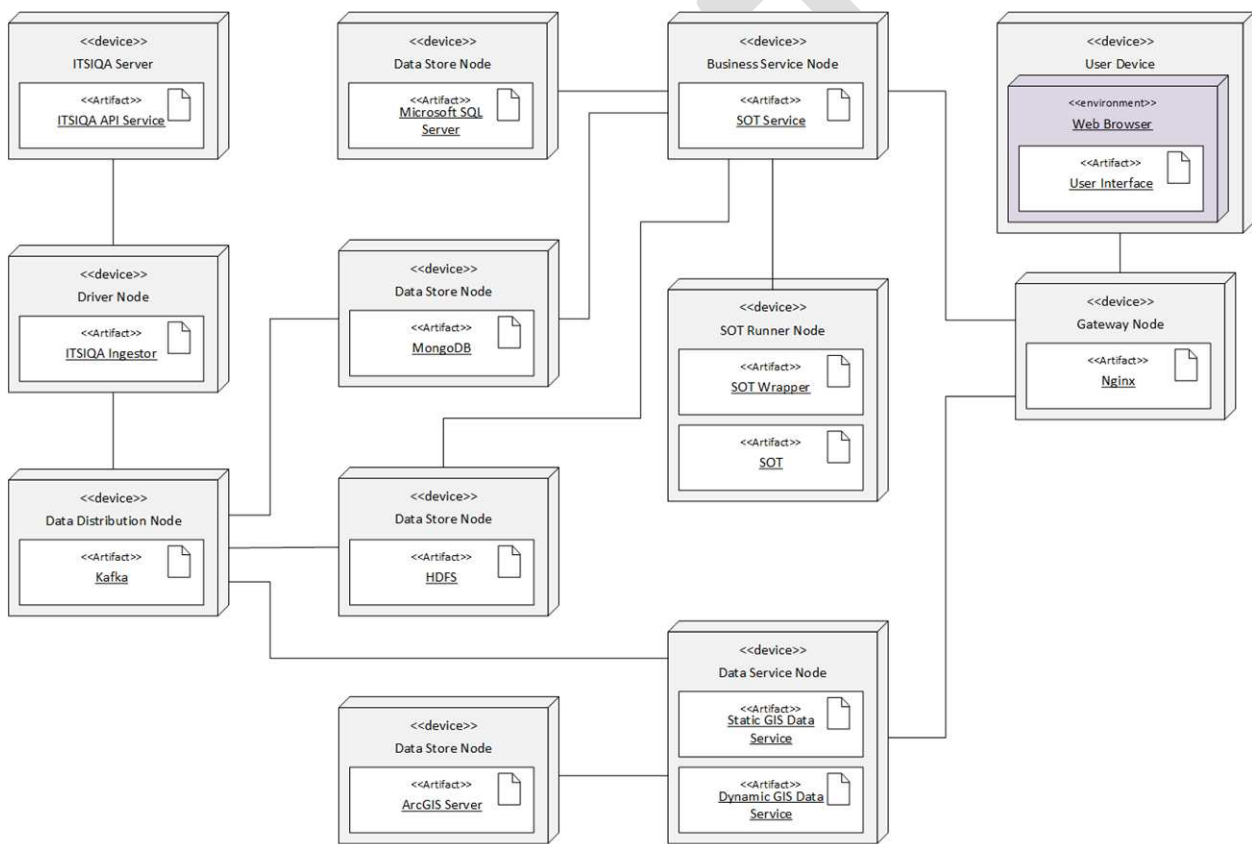


Figure 150– Service Host Deployment Diagram

3.7.2 Containerized Service Orchestration

Figure 151 shows the scalable deployment and interaction between an internal R-ICMS service (Service X), the authentication and authorization service, and logging and monitoring services. Services are deployment onto host systems within Docker containers grouped into scalable pods by the Kubernetes service orchestration tool. This diagram assumes use Kubernetes as the orchestration tool and Elastic Stack for logging and monitoring with a Kafka buffer.

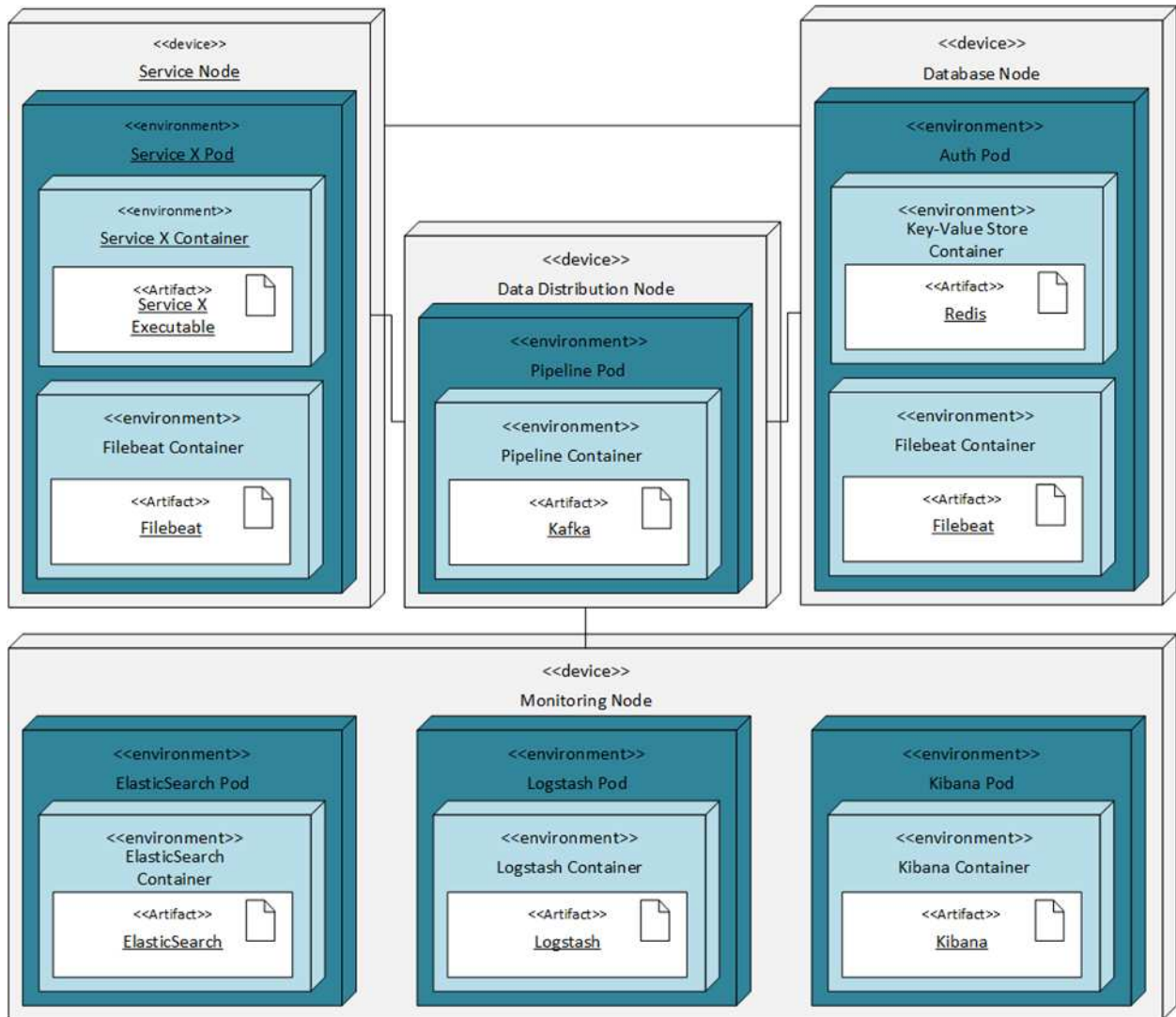


Figure 151 – Containerized Service Orchestration Diagram

4 Notes

4.1 Definitions

1. Component – Generally, a part of some larger system or product
 1. Traffic system component – Part of the transportation system, such as a DMS or a traffic sensor
 2. Architectural components – Part of the designed implementation architecture, such as a service, a data store, a pipeline, or a user interface
2. Corridor – Except where used in the title of this system, an acyclic set of coordinated neighboring signalized intersections
3. Layer –
 1. A set of services/capabilities within the design architecture that provides similar capabilities and that interact in similar ways with other layers
 2. A set of viewable artifacts on a map interface that can be turned on or off (i.e., made visible or invisible to the user)
4. Response Plan – A set of response plan elements which can be considered for implementation in response to traffic conditions
5. Response Plan Element – A specific change that can be made to controllable elements of the traffic network; e.g., activation of a signal timing plan or putting a set of message elements on a DMS
6. Special Event – A future planned event; e.g., parade, football game
7. Subsystem – generally a system that is part of a larger system
 1. One of the three parts of the R-ICMS identified in the original Invitation to Negotiate (the IEN, DSS, and DFE)
 2. A part of SunGuide dealing with specific types of activities or data

DRAFT

Appendix A

Table 57 – Data Sources Table

R-ICMS Data Source	Data Source Detail (if decomposition is needed)	Data Source availability mechanism	Update Interval
ITSIQA	Traffic Conditions Data	ITSIQA	1 min
District 5 SunGuide System	Traffic Incident Data	SunGuide / Databus / EM	real-time
District 5 SunGuide System	CCTV Status	SunGuide / Databus / CCTV	real-time
District 5 SunGuide System	Ramp Meter Status	SunGuide / Databus / RMS	real-time
District 5 SunGuide System	Dynamic Message Signs Status	SunGuide / Databus / DMS	real-time
District 5 SunGuide System	Connected Vehicle Roadside unit status	SunGuide / Databus / CV	real-time
Signal Performance Measures	Volume for each movement	ATSPM D5 Deployment	1 min
Signal Performance Measures	Arrival timing per movement	ATSPM D5 Deployment	1 min
Transit GTFS-RT - SunRail			real-time
Transit GTFS-RT - Lynx Clever			real-time
Transit GTFS-RT - Lynx Trapeze			real-time
Transit GTFS-RT - Votran			real-time
Transit GTFS-RT - Lake Express			real-time
Transit GTFS-RT -SCAT (Space Coast Area Transit)			real-time
Transit GTFS-RT -SunTran (Marion County)			real-time
Transit GTFS - SunRail			
Transit GTFS - Lynx			
Transit GTFS - Votran			
Transit GTFS - Lake Express			
Transit GTFS -SCAT (Space Coast Area Transit)			
Transit GTFS -SunTran (Marion County)			
Weather	National Weather Service Watches and Warnings	NOAA; Is there capability to do weather radar for map overlay for IEN	real-time
Parking Data	Garages, surface lots, weigh stations, rest areas, beaches, on-street parking	SunGuide / Databus / TPS or Parking Subsystem	real-time
basemap "backdrop"		HERE Navstreets or ESRI ArcGIS system	quarterly
basemap links		FDOT (manually corrected version of HERE.com basemap)	3 months
Roadway Characteristics Inventory (RCI)	# lanes at intersection	FDOT RCI	
Predictive Engine Data		PRE	
Expert Rules Engine – Response Plans		ERE	

System Design Document for Regional Integrated Corridor Management System (R-ICMS)

R-ICMS Data Source	Data Source Detail (if decomposition is needed)	Data Source availability mechanism	Update Interval
Evaluation Engine		EVE	
Intersection Geometry Data		TBD	
Intersection Plans and Schedules		TBD	
Intersection Movement Counts Data	turning movement counts	IMC	
Intersection Movement Counts Data	approach/direction	IMC	
Intersection Movement Counts Data	# lanes	IMC	
Intersection Movement Counts Data	saturation flow rates	IMC	
Intersection Movement Counts Data	vehicle detection	IMC	
Intersection Movement Counts Data	bike detection	IMC	
Intersection Movement Counts Data	pedestrian detection	IMC	
Origin Destination - SunGuide BlueTOAD	BlueTOAD	SunGuide D4 Enhancement	
Origin Destination - Tag Reads	BlueMAC	SunGuide	
Controller or ATMS	Signal Phasing	Signal ATMS Software or SunGuide	
Controller or ATMS	Detector Status Data (Vehicular and Pedestrian)	Signal ATMS Software or SunGuide	
Controller or ATMS	Controller Data	Signal ATMS Software or SunGuide	
Controller or ATMS	Available Controller Timing Patterns with timing plan details	Signal ATMS Software or SunGuide	
Controller or ATMS	Controller Timing pattern status	Signal ATMS Software or SunGuide	
Controller or ATMS	Corridor Plan	Signal ATMS Software or SunGuide	
Special Event Information		AAM Dashboard	irregular
School locations		FDOT provided GIS layer	
School zones		FDOT provided GIS layer	
School schedules		FDOT provided data	
Express lanes status	status, current price, current pricing model, if the system has any dynamic features such as changeable toll rates, changeable lane configurations	SunGuide - Future	
Emergency Responder Locations	Hospitals, Police stations, Fire stations	TBD (GIS layer possibly)	
Railroad Signaling System	PTC Insaldo ATMS (SunRail)	TBD	
Video - iVEDDS			