

***CONNECTED VEHICLES CORE SERVICES (CVCS)***

***– Support Services Subsystem (SSS)***

**Operations Manual**

Version *<1.2>*

*<09/28/2017>*

|  |  |
| --- | --- |
| Client: | *FHWA Intelligent Transportation Systems Joint Program Office –*  *Ariel Gold* |
| Date: | *9/28/17* |
| Project Number: | *DTFH-61-15-R-00037* |
| Work Location(s): | *Michigan* |
| Team | *Leidos* |

**VERSION HISTORY**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Version #** | **Implemented By** | **Date** | **Approved By** | **Date** |
| 1.0 | *Matt Vahlberg, Erik Hamilton, Diane Newton* | *07/14/2017* | *<name>* | *<mm/dd/yy>* |
| 1.1 | *Matt Vahlberg, Erik Hamilton, Diane Newton* | *09/19/2017* | *<name>* | *<mm/dd/yy>* |
| 1.1 | *Matt Vahlberg, Erik Hamilton* | *09/28/2017* | *<name>* | *<mm/dd/yy>* |
|  |  |  |  |  |

Table of Contents

[1 Introduction 1](#_Toc494895881)

[2 System Overview 1](#_Toc494895882)

[2.1 System Background 1](#_Toc494895883)

[2.2 System Documentation 1](#_Toc494895884)

[CVCS-SSS Operations and Maintenance Manual 1](#_Toc494895885)

[CVCS-DDS Operations and Maintenance Manual 1](#_Toc494895886)

[CVCS-SM Operations and Maintenance Manual 1](#_Toc494895887)

[CVCS Software Upgrade Plan 1](#_Toc494895888)

[CVCS Configuration Management Plan and Database 1](#_Toc494895889)

[RSU Equipment 2](#_Toc494895890)

[CVCS Code and Documentation 2](#_Toc494895891)

[2.3 System Architecture 3](#_Toc494895892)

[Situation Data Warehouse (SDW) 3](#_Toc494895893)

[Situation Data Processing Center (SDPC) 3](#_Toc494895894)

[Situation Data Clearinghouse (SDC) 3](#_Toc494895895)

[Subscription Database 3](#_Toc494895896)

[Object Registration and Discovery Service (ORDS) 4](#_Toc494895897)

[JMS Publication Server 4](#_Toc494895898)

[Authentication Server 4](#_Toc494895899)

[WebSocket Server 4](#_Toc494895900)

[Forwarders 4](#_Toc494895901)

[System Monitor 4](#_Toc494895902)

[SAE J2735 4](#_Toc494895903)

[3 Software Summary 4](#_Toc494895904)

[3.1 Software Description 4](#_Toc494895905)

[3.2 Commercial Services and Software Components 5](#_Toc494895906)

[3.3 UDP Dialog Interfaces 6](#_Toc494895907)

[3.3.1 ADP UDP Dialog Sequence 6](#_Toc494895908)

[3.3.2 UDP Dialog to DEF Plugins Mapping 7](#_Toc494895909)

[3.3 Forwarders 7](#_Toc494895910)

[3.3.1 Rackspace 7](#_Toc494895911)

[3.4 Object Registration and Discovery Service (ORDS) 8](#_Toc494895912)

[3.4.1 ORDS Components 8](#_Toc494895913)

[3.4.2 Developed Sub-Components for Integration into a CV Environment 10](#_Toc494895914)

[3.5 Web Environment 11](#_Toc494895915)

[4 Running the Systems 15](#_Toc494895916)

[4.1 Initiate Forwarders 15](#_Toc494895917)

[4.1.1 Start the Virtual Machines (VMs) 15](#_Toc494895918)

[4.1.2 Install the Software 15](#_Toc494895919)

[4.2 Building the Systems (each system) 16](#_Toc494895920)

[4.2.1 Upload Plugins 16](#_Toc494895921)

[4.2.2 Setup Data Models 16](#_Toc494895922)

[4.2.3 Build the System 17](#_Toc494895923)

[4.2.4 Start the System 17](#_Toc494895924)

[4.2.5 Verify 17](#_Toc494895925)

[4.3 Complete Forwarders 18](#_Toc494895926)

[4.3.1 Configure Addresses 18](#_Toc494895927)

[5 Monitoring 18](#_Toc494895928)

[5.1 Management Console and System Monitor 18](#_Toc494895929)

[5.2 Warehouse Tools 18](#_Toc494895930)

[5.3 Test Routine: Data Type ISD, VSD, etc. 18](#_Toc494895931)

[5.4 Notification Scheduling 19](#_Toc494895932)

[6 Updating OSS Licensing 19](#_Toc494895933)

[7 Troubleshooting 19](#_Toc494895934)

[7.1 Ticketing System 19](#_Toc494895935)

[7.2 Uptime Dashboard 19](#_Toc494895936)

[7.3 DigitalEdge Console 20](#_Toc494895937)

[7.4 AWS Console 20](#_Toc494895938)

[7.5 Rackspace Console 20](#_Toc494895939)

[7.6 Maintenance 20](#_Toc494895940)

[Appendix A: List of Acronyms 21](#_Toc494895941)

[Appendix B: Maintenance Plan 22](#_Toc494895942)

[B.1 Software Updates 22](#_Toc494895943)

[B.1.1 Updating Plugins 22](#_Toc494895944)

[B.1.2 Updating Software Libraries Directly 22](#_Toc494895945)

[B.1.3 Rebuild and Cycle Affected System 22](#_Toc494895946)

[B.1.4 Update Forwarders 23](#_Toc494895947)

[B.2 Continued Operations & Infrastructure Maintenance 23](#_Toc494895948)

[B.2.1 AWS VM Retirement 23](#_Toc494895949)

[B.3 Periodic Diagnostics 24](#_Toc494895950)

[B.3.1 DigitalEdge Management Console 24](#_Toc494895951)

[B.3.2 DigitalEdge System Monitor 24](#_Toc494895952)

[B.3.3 AWS Console 24](#_Toc494895953)

[B.4 Backup Procedure for the Situation Data Warehouse 24](#_Toc494895954)

[B.4.1 Volume Snapshots 24](#_Toc494895955)

[B.5 Maintenance Issues 24](#_Toc494895956)

[B.6 Security Monitoring and Patching 25](#_Toc494895957)

[B.6.1 Software 25](#_Toc494895958)

[B.6.2 Instance Access 25](#_Toc494895959)

[Appendix C: DigitalEdge™ 15](#_Toc494895960)

[C.1 Team 15](#_Toc494895961)

[C.2 Licensing 15](#_Toc494895962)

[C.3 Documentation 15](#_Toc494895963)

List of Tables and Figures

[Table 1. Dynamic Plugin Parameter Values 17](#_Toc494895964)

[**Figure 1- System Architecture** 3](#_Toc494895965)

[**Figure 2- ADP UDP Dialog Sequence** 7](#_Toc494895966)

[**Figure 3- UDP Dialog to DE Plugin Mapping** 7](#_Toc494895967)

[**Figure 4- Relationship between Rackspace and AWS** 8](#_Toc494895968)

[**Figure 5- ORDS Components** 9](#_Toc494895969)

[**Figure 6- ISD Message Creator Application** 12](#_Toc494895970)

[**Figure 7- TIM Message Creator Application** 12](#_Toc494895971)

[**Figure 8- Warehouse Tools Set** 12](#_Toc494895972)

[**Figure 9- Warehouse Tools Map Application** 14](#_Toc494895973)

[**Figure 10- Message Validator Application** 15](#_Toc494895974)

# 1 Introduction

This document provides information concerning maintenance of the Support Services Subsystem for the United States Department of Transportation (USDOT) Southeast Michigan Connected Vehicles Core Services (CVCS) project. Operating these systems requires periodic interaction with the underlying server-side systems, which run in Amazon Web Services (AWS) and Rackspace. This document reflects the Version 2.3 CVCS systems developed by Leidos.

# 2 System Overview

## 2.1 System Background

In 2007, the United States Department of Transportation (USDOT) established a Connected Vehicle (CV) Test Bed in Oakland County, Michigan (known as both the Unified Implementation of the Reference Architecture and the Southeast Michigan Test bed – referred herein as the SEMI Test Bed) as a test facility for Proof of Concept (POC) testing of Dedicated Short-Range Communications (DSRC) using allocated bandwidth within 5.9 GHz. The SEMI Test Bed comprises all of the required elements for a functional V2V and V2I/I2V, 2016 standards compliant, real-world deployment.

## 2.2 System Documentation

**The CVCS system includes numerous components within its software ecosystem. The following documentation**

### **CVCS-SSS Operations and Maintenance Manual**

This document discusses the system as a whole, as well as operations and maintenance of the specific components of the Forwarders, ORDS, and web environment of the Web App server, CAS authentication server and WebSocket server.

### **CVCS-DDS Operations and Maintenance Manual**

Documentation addressing operations and maintenance of the Situation Data Warehouse (SDW), Situation Data Processing Center (SDPC), and Situation Data Clearinghouse (SDC) components with the associated JMS publication server and subscription database.

### **CVCS-SM Operations and Maintenance Manual**

Documentation for operations and maintenance of the System Monitor.

### **CVCS Software Upgrade Plan**

Documentation regarding the precautions and procedures of upgrading the various major software components of the CVCS system.

### **CVCS Configuration Management Plan and Database**

Documentation specifying processes for configuration management of CVCS system, as well as accounting of the various components.

### **RSU Equipment**

For documentation regarding the roadside equipment, please consult the appropriate manufacturer.

### **CVCS Code and Documentation**

For documentation regarding the software, please see the relevant open source code repository and associated documentation. Major components are as follows:

SDC, SDPC, SDW, SM, ORDS and related:

https://github.com/connectedvehiclessitdat

Map Tools:

https://github.com/connectedvcs

MongoDB

https://docs.mongodb.com/

DigitalEdge Framework

https://github.com/deleidos/digitaledge-platform

Java

https://docs.oracle.com/javase/6/docs/

JMS

http://docs.oracle.com/javaee/6/tutorial/doc/bncdq.html

Jetty

http://www.eclipse.org/jetty/documentation/

ActiveMQ

http://activemq.apache.org/getting-started.html

WebSockets

https://docs.oracle.com/javaee/7/tutorial/websocket.htm

CAS

https://apereo.github.io/cas/index.html

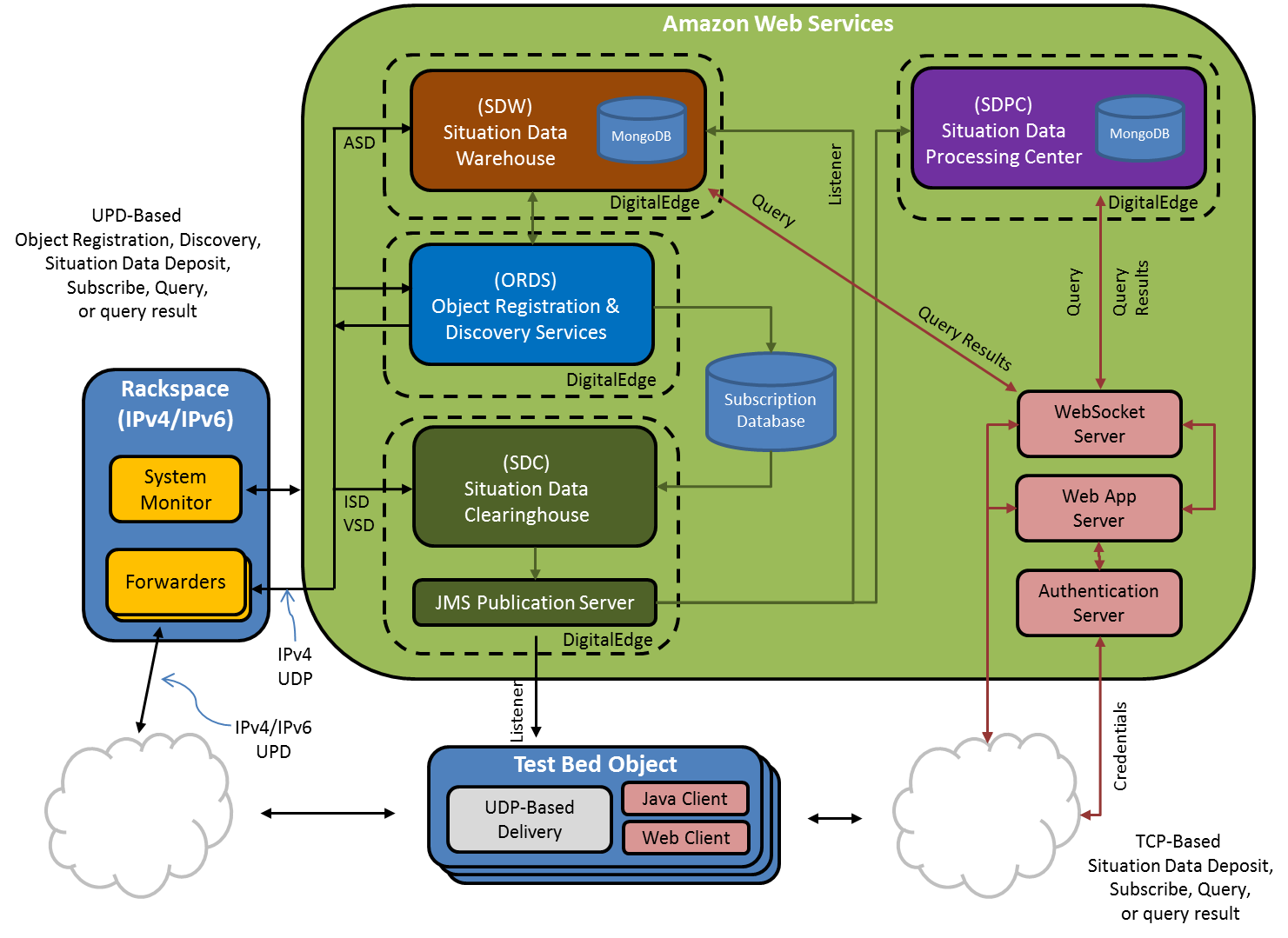
Rackspace

https://developer.rackspace.com/docs/

AWS

https://aws.amazon.com/documentation/

## 2.3 System Architecture

****

**Figure 1- System Architecture**

### **Situation Data Warehouse (SDW)**

Stores VSD and ISD from SDC and Advisory Situation Data (ASD – TIM) directly for up to 30 minutes and makes those available via query service. Based on the TIM content, message retention could be longer.

### **Situation Data Processing Center (SDPC)**

Query service identical to SDW, with data retention for nine months of VSD and ISD messages. It does not retain TIM messages.

### **Situation Data Clearinghouse (SDC)**

Consumes Vehicle Situation Data (VSD - BSM bundles) and Intersection Situation Data (ISD – MAP/SPaT) and makes those available via subscription in near real time. Component does not store data. Purpose is to serve messages available via JMS subscription.

### **Subscription Database**

MongoDB database containing the subscription information.

### **Object Registration and Discovery Service (ORDS)**

The ORDS facilitates object registration and discovery, and supports UDP query for SDW and subscription requests for SDC. It also is responsible for sending out results of query to SDW.

### **JMS Publication Server**

JMS service for exposing a data-receiving endpoint for subscribers wishing to connect to the SDC. A consumer of the data must first subscribe to the ORDS.

### **Authentication Server**

The Authentication Server uses CAS to provide the authentication layer for the web applications, as well as the websockets data layer.

### **WebSocket Server**

The WebSocket Server provides a data service layer for the web applications to access to the SDPC, SDC and SDW.

### **Forwarders**

The forwarders receive data streams on IPv6 or IPv4, via UDP, and forward as UDP over IPv4 to SDC, SDW and ORDS. Unlike most of the other components, this system runs in Rackspace.

### **System Monitor**

The System Monitor is a health and wellness utility whose purpose is to monitor system uptime. Unlike most of the other components, this system runs in Rackspace.

### **SAE J2735**

Uses custom UDP interface extensions in the style of J2735 that provide data exchange information in addition to data description provided by SAE J2735. The system supports non secure and secure (based on IEEE 1609.2 message exchange over IPv4 and IPv6, and supports UPD and WebSockets interfaces.

# 3 Software Summary

## 3.1 Software Description

The CVCS software system operates in Amazon Web Services (AWS) and Rackspace Cloud environments. Rackspace is used to accept, translate, forward, and distribute User Datagram Protocol (UDP) packets over Internet Protocol Version 6 (IPv6). This is the interface to the systems from roadside units (RSUs), which receive vehicle situation data and intersection data from devices within range of the RSU. The forwarders running in Rackspace translate the packets from IPv6 to the more commonly used Internet Protocol Version 4 (IPv4) for data ingestion into the AWS cloud. Within AWS, the systems are run on the Leidos DigitalEdge framework. DigitalEdge provides general-purpose ingest, processing, and data storage services for large, complex systems. The ORDS system is housed in the DigitalEdge framework.

## 3.2 Commercial Services and Software Components

The CVCS cloud infrastructure contains libraries and certificates for commercial services and software components that the system depends upon to handle encoding/decoding per the ASN.1 BER format and encrypting/decrypting the 1609.2 messages. This section will describe an overview and introduction for each of the various commercial services and software components, for more details refer to the Software Architecture section of this document.

***Amazon Web Services (AWS)*** – Enables the operation of the ORDS via cloud infrastructure. The cloud resources utilized through AWS are Elastic Computing (EC2), Elastic Block Storage (EBS), Simple Storage Service (S3), and Route53.

* EC2 – Handles processing dialogs and situation data messages.
* EBS – Stores the registrations in a Database (DB).
* S3 – Stores configuration files, software components, and other miscellaneous artifacts such as the Java keystore and truststore files.
* Route53 – Provides DNS access to the services.

***Rackspace*** – AWS does not support UDP traffic over the IPv6 network, which is a requirement for receiving intersection and vehicle situation data. As such, Rackspace is used to forward situation data messages from the IPv6 network into the CVCS systems running in AWS on an IPv4 network. The cloud resource used here is the virtual cloud server, and it provides a dual stack network interface that supports both IPv6 and IPv4.

***DigitalEdge –*** DigitalEdge is a big data platform developed by Leidos that runs in AWS. DigitalEdge runs the four systems that comprise the CVCS data distribution and support services subsystems. Included within DigitalEdge are management tools that support data modeling, system configuration and construction, system operation and status, and system performance monitoring. To meet the needs of a CV environment, the Leidos team created multiple running DigitalEdge systems.

***OSS ASN.1******Studio*** – CVCS uses a software library provided by OSS Nokalva to handle encoding and decoding the Connected Vehicle dialog and situation data messages in UPER format. The studio is a desktop application that comes with the OSS Nokalva software package and is used to generate the Java source which the ORDS uses in AWS. In order to use the studio, users must acquire the Connected Vehicle and Dedicated Short Range Communication ASN.1 specification files.

* DSRC ASN.1 Specification - <http://standards.sae.org/j2735_201603/>
* Connected Vehicle ASN.1 Specification - <https://gitlab.com/connectedvehicles/fedgov-cv-asn1/blob/master/src/main/resources/SEMI_v2.3.0_070616.asn>

***1609.2 Certificates*** – All encrypted messages will be encrypted following the IEEE 1609.2-2016 standard. A certificate will need to be obtained from the Security Certificate Management System (SCMS) to decrypt any encrypted ASN.1 UPER messages.

## 3.3 UDP Dialog Interfaces

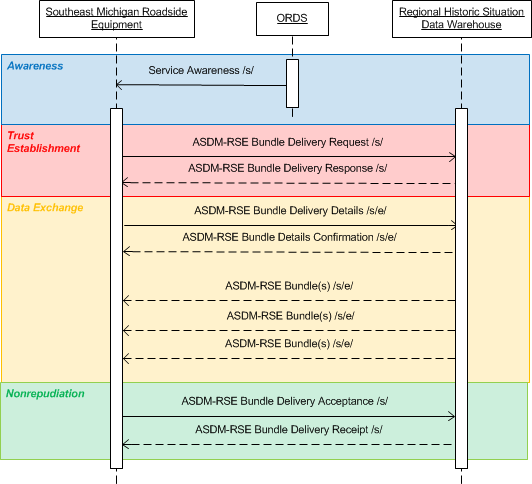
The UDP dialog interfaces provide different modes and message types. The message exchange sequences are:

* Data deposit for VSD, ISD and ASD
* Data distribution for VSD, ISD and ASD
* SDC subscription request and cancellation
* Service/object registration and discovery

The UDP message exchanges are enclosed in a secure 1609.2 envelope.

### 3.3.1 ADP UDP Dialog Sequence

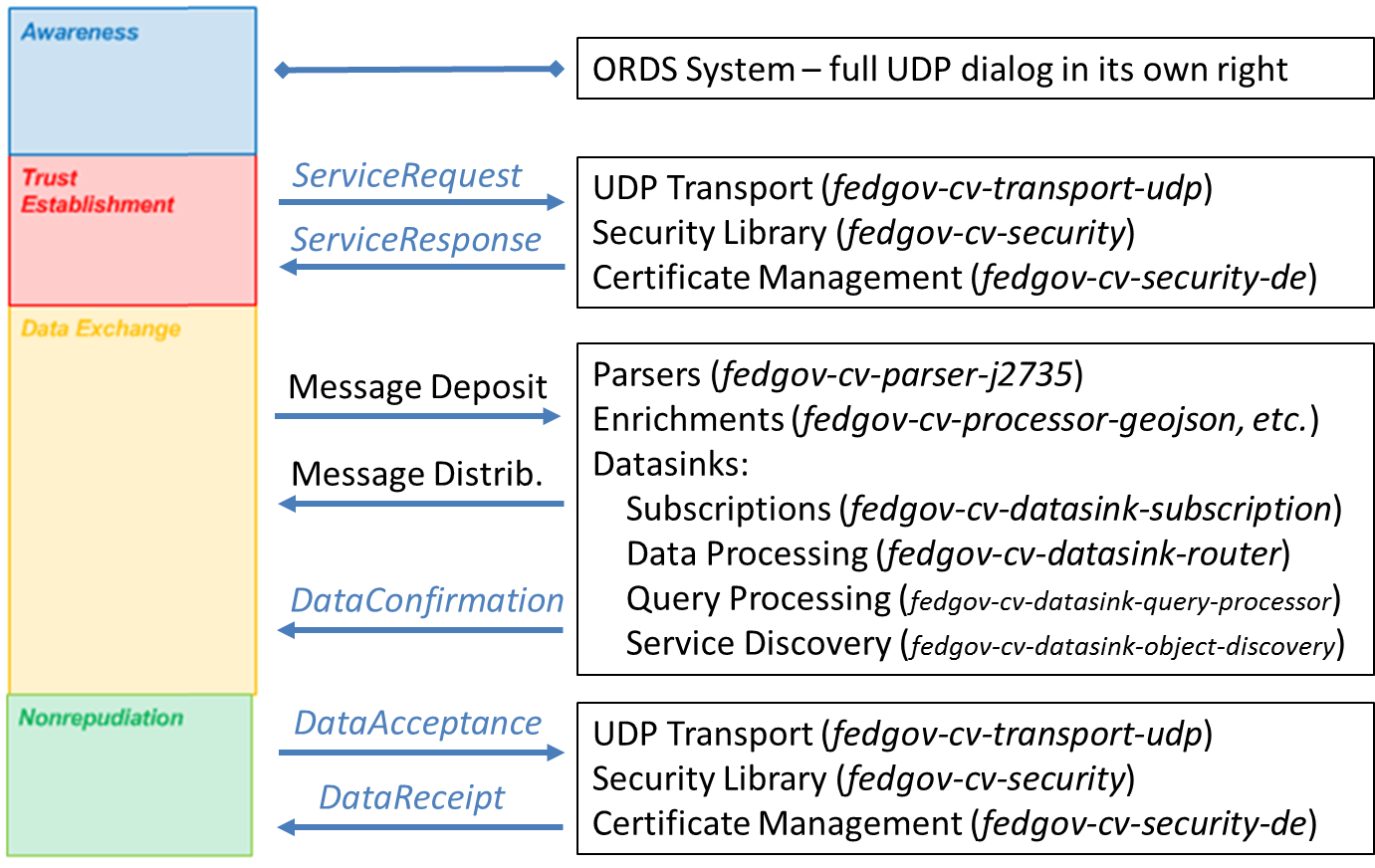
In the figure below, an example of an ADP UDP Dialog Sequence is provided:



**Figure 2- ADP UDP Dialog Sequence**

### 3.3.2 UDP Dialog to DEF Plugins Mapping

In the figure below, the mapping of DigitalEdge Framework (DEF) plugins to the corresponding UDP dialog is shown:



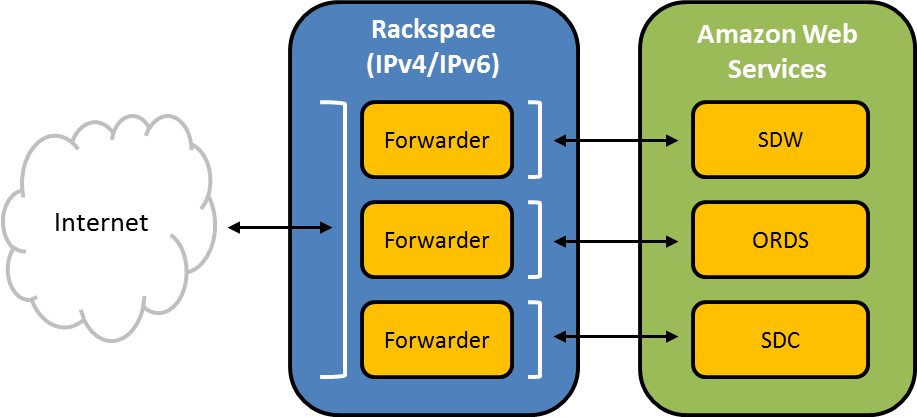
**Figure 3- UDP Dialog to DE Plugin Mapping**

## 3.3 Forwarders

The forwarders exist as AWS does not support UDP over IPv6. Because AWS does not support this capability, Rackspace.com is utilized since its virtual instance provides a dual stack IPv6/IPv4 network interface. To complete the functionality, a message forwarding component was implemented that resides in Rackspace, receives incoming messages in IPv4 or IPv6, forwards them to DigitalEdge Framework (DEF) systems in AWS over IPv4, receives messages from DEF systems in IPv4 and sends them out to the original requestor via IPv4 or IPv6.

### 3.3.1 Rackspace

One of the requirements of for the CV systems is to receive requests via IPv6 networks using UDP. As shown in Figure 4, Rackspace runs custom developed applications that handle data deposit and data request messages from both IPv6 and IPv4 networks and forwards them to the appropriate DigitalEdge system transport. In return, the application forwards responses back out, providing a single IP Address for third party applications to send and receive. All IPv6 input and output traffic must go through the Rackspace forwarder application. Optionally, the DigitalEdge systems in AWS can be configured to send directly to third party applications if the response is going out on the IPv4 network, for example the web application utilize this approach.



**Figure 4- Relationship between Rackspace and AWS**

On the Rackspace servers, the following java command-line applications are present:

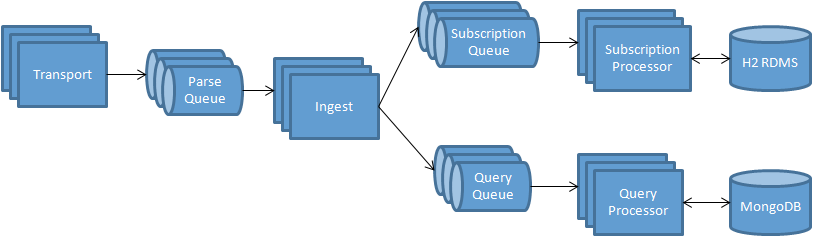
* Forwarder application: installed on a Rackspace instance for forwarding UDP traffic over the IPv6 and IPv4 into and out of the SDC and SDW.
* System Monitor application: installed on a Rackspace instance to monitor the health and uptime of all the system’s components.

## 3.4 Object Registration and Discovery Service (ORDS)

In order to meet the requirements of the ORDS, the DigitalEdge system runs in the AWS cloud environment. There are three core systems. Two systems handle the real-time push of data to a data warehouse. The third system separates data requests from situation data deposits, so subscription and query requests can be fulfilled in a timely manner and also facilitates the ORDS functionality of registering authorized subscribers to receive real-time data or to develop query requests for data. Components developed specifically for the Southeast Michigan project are denoted in italic font. Native DigitalEdge components are identified in regular font.

Many of the roadside or center-based services offered by providers operating within the Southeast Michigan Test Bed geographic boundary will require registration in order to provide potential users (e.g., Test Bed vehicles and other roadside and center-based services) an advertisement of their existence and cyber location. The Object Registration and Discovery Service (ORDS) will facilitate the registration of services by the respective service providers and the subsequent query-based discovery of these registered services. Please note that center-based services operating outside of Southeast Michigan Test Bed geographic boundary will not be allowed to register.

### 3.4.1 ORDS Components



**Figure 5- ORDS Components**

Within the ORDS, formerly known as Distribution System, shown in **Figure 5Error! Reference source not found.**, each box and cylinder in the diagram above represents a virtual instance running in the AWS environment. The boxes and cylinders that are stacked indicate that one or more instance of the same type can be configured to run depending upon the configuration of the ORDS. The ability to create additional instances of the same type provides a way to horizontally scale the system to meet the performance requirements.

* Transport - This instance type runs the *fedgov-cv-transport-udp* custom component, and it receives and responds to subscription and data requests from ASN.1 UPER dialog messages. When a DataSubscriptionRequest, RsuAdvisorySituationDataRequest, or IntersectionSituationDataRequest message is received, the Transport will decrypt and validate it to ensure that it is properly formed and within the specified service region of the system before placing it on the parse queue for processing. One or multiple Transport instances can be configured to run at once in the Distribution system.
* Parse Queue - This instance type runs an ActiveMQ JMS Server and is responsible for storing the subscription and data request messages until the Ingest instance picks them up for processing. This ensures that no messages are lost when the Ingest instance is busy. One or three Parse Queue instances can be configured to run in the Distribution system.
* Ingest - This instance type runs the *fedgov-cv-parser-j2735* custom component. It parses the subscription and data request messages and maps the extracted information into JSON data records as per the associated DigitalEdge data model. One or multiple Ingest instances can be configured to run at once, or the Distribution system can be configured to scale the number of Ingest instances dynamically depending on the load of the system.
* Subscription Queue - This instance runs an ActiveMQ JMS Server and is responsible for storing subscription request JSON data records until a Subscription Processor instance can accept those records for processing. This ensures that no messages are lost when the Processor instance is busy. One or three Subscription Queue instances can be configured to run in the Distribution system.
* Subscription Processor – This instance type runs the *fedgov-cv-processor-subscription* custom component. It adds new subscriptions or actively and passively removes subscriptions from the H2 RDBMS. Active removal pertains to an action initiated by a subscriber via the DataSubscriptionCancel ASN.1 UPER message. Passive removal pertains to expired subscriptions either by the subscription end time reached or by expiration of the subscriber certificate. When a new subscription is added, the Subscription Processor assigns the subscriber an ID between 10,000,000 and 99,999,999; this is also the name of the topic the subscriber will connect to on the Pub/Sub Server. Only one Subscription Processor is recommended, but more than one can be configured.
* H2 RDBMS – This is referring to the same instance described for the Clearinghouse system. The Subscription Processor places a new subscription and actively (initiated by the client) or passively (expired) removes the subscription from the H2 RDBMS instance.
* Query Queue - This instance type runs an ActiveMQ JMS Server and stores data request situation JSON records until a Query Processor instance accepts them for processing. This ensures that no messages are lost when a Query Processor instance is busy. One or three Query Queue instances can be configured to run in the Distribution system.
* Query Processor - This instance type runs the *fedgov-cv-datasink-query-processor* custom component. It queries the MongoDB database for vehicle and intersection situation data. After receiving the query response, the Query Processor packages the result into one or more RsuAdvisorySituationDataBundle or IntersectionSituationDataBundle ASN.1 UPER messages depending on the query type. It encrypts the messages using the requestor’s 1609.2 certificate, and sends it back to the requestor over UDP. One or multiple Query Processor instances can be configured to run at once, or the Distribution system can be configured to scale the number of Query Processor instances dynamically depending on the load of the system.
* MongoDB – This is referring to the same MongoDB database instance specified in the Warehouse system. The Query Processor connects to the MongoDB instance to perform queries for advisory and intersection situation data.

### 3.4.2 Developed Sub-Components for Integration into a CV Environment

The native DigitalEdge platform alone does not meet the system needs for the ORDS. To complete the required functionality several plug-in components were developed using the DigitalEdge software development kit (SDK). DigitalEdge supports integration of custom components into the platform using a simple upload feature. Below is a description of the various software components developed to ensure integration of the ORDS into a real-world CV Test Bed environment.

***CVCS Sub-Components*** *– These are components specifically developed for the USDOT Southeast Michigan CVCS project. While the following components in this subsection may be useful for others implementing similar services, it is important to note that they were developed for integration with a specific CV Test Bed. The components run in the DigitalEdge to support specific functionality:*

* fedgov-cv-asn1: This plugin is generated from using the OSS Studio desktop tool using the Connected Vehicle and DSRC ASN.1 specification files. It also contains common utility classes used to assign the encoding and decoding of the Connected Vehicle messages.
* fedgov-cv-common: This plugin contains common Java classes used by the other projects.
* fedgov-cv-transport-udp: This plugin contains the data receiver component of the ORDS. The transport handles all Connected Vehicle messages.
* fedgov-cv-parser-j2735: This plugin is for the ORDS to parse subscription messages, query messages, object registration data, and object discovery data requests and translate them according to mappings specified in the DigitalEdge data models.
* fedgov-cv-processor-geojson: This plugin is for the ORDS to enrich object registration data with GeoJSON information.
* fedgov-cv-datasink-subscription: This plugin is to handle subscriptions. The subscription information is stored on the DigitalEdge internal H2 database.
* fedgov-cv-datasink-object-discovery: This plugin is for the ORDS to handle object discovery data requests.
* fedgov-cv-datasink-object-registrar: This plugin is for the ORDS to hand object registration data.

fedgov-cv-datasink-query-processor: This plugin is for the SDW to service query requests for intersection and traveler information messages.

## 3.5 Web Environment

Within CVCS, there are three web components to facilitate the web applications. They are the web server, the CAS server and the websocket server. The environment supports the set of web applications developed to visualize the situation data routed and stored in the CVCS system. Within this environment, the following items are present:

#### Single Sign-on

This is the Central Authentication Service (CAS) web application. The application runs in the background and has no operational interaction. It provides a single sign-on mechanism for the other web applications.

#### Intersection Data (ISD) Message Creator

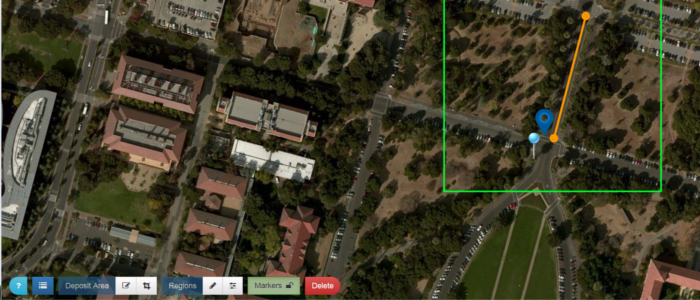
This web application allows for the creation, editing, and encoding of Intersection MAP and SPaT messages. The ISD Message Creator allows a user to define the lanes and approaches of an intersection using a graphical interface. Once designed, the user can convert the message to ASN.1 or Hex.



**Figure 6- ISD Message Creator Application**

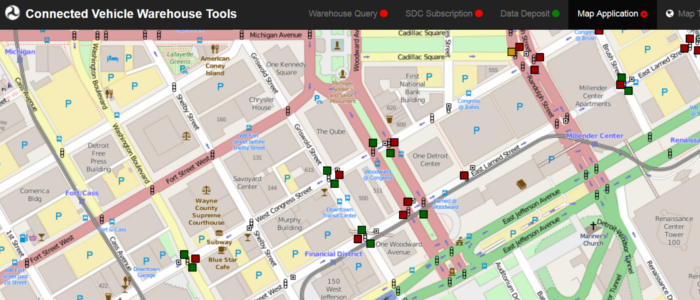
#### Traveler Information Message (TIM) Message Creator

This web application allows for the creation, editing, encoding, and deposit of Traveler Information messages. The TIM Message Creator tool allows users to build traveler information messages regarding sign and work zone details using a graphical interface. Once designed, the user can convert the message to ASN.1 or Hex and deposit it to the SDW.



**Figure 7- TIM Message Creator Application**

#### The Warehouse Tool Set



**Figure 8- Warehouse Tools Set**

The Warehouse Tools set is a full suite that allows a user to subscribe, query, and deposit messages to the warehouses. There is also a mapping application that displays vehicle and signal information in real-time. The tool set includes:

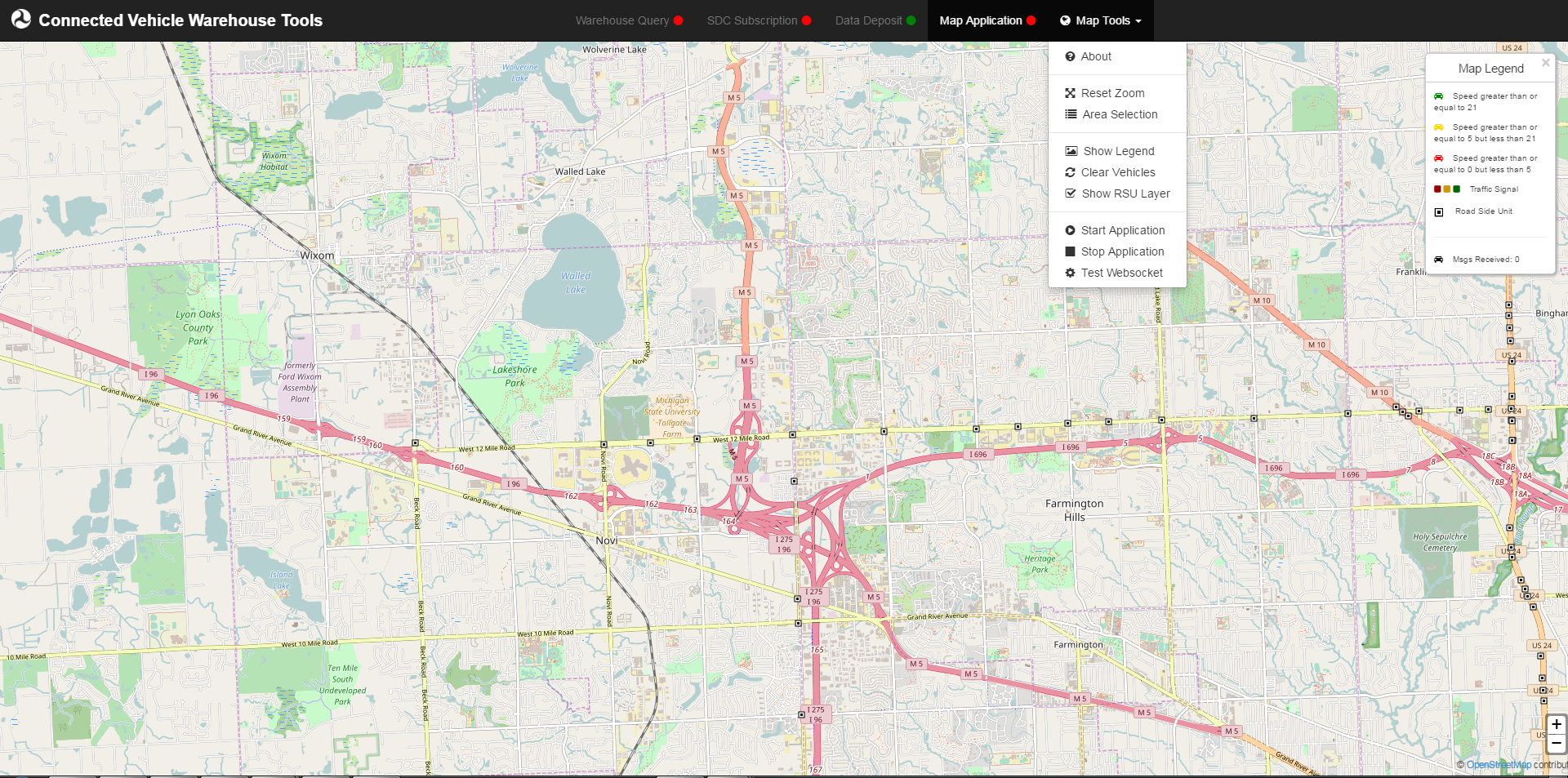
* Warehouse Query: This web application provides a way to query the SDW for traveler information and situation data.
* SDC Subscription: This web application provides a way to subscribe to the SDC and receive near-real time situation data.
* Data Deposit: This web application allows direct deposit of messages into the SDC or SDW.
* Map Application: This web application subscribes to the SDC for vehicle situation data and plots them on a map.

##### Warehouse Tools Map Application

The Map Application visualizes vehicle situation and intersection data from the SDC. The user interface (UI) connects to a WebSocket server running in AWS. The WebSocket server is a subscriber to the SDC that implements a special listen-all subscription that receives every deposited data input into the SDC. The UI can select information to display according to user-specified geospatial regions.

To use the UI, connect a web browser to the URL of the WebSocket server. The URL is: *https://webapp2.connectedvcs.com/whtools23/*. Upon connecting, the UI requires a valid username and password. Once valid credentials are entered, the application is listed as *Map Application* along the top navigation bar.

A screen shot of the application follows in Figure 9. Vehicle situation data is plotted on the map once a second for all vehicles transmitting situation data bundles falling within the assigned geographic region. Each one-second bundle is plotted on the map as a color-coded dot. Each dot disappears off the map 30 seconds later to avoid the display from becoming cluttered with old information. The 30-second period is the default setting but is customizable on the WebSocket server. The dot colors are red, yellow, or green according to the speed of the vehicle, which is also customizable on the WebSocket server.



**Figure 9- Warehouse Tools Map Application**

The default dot values, and those depicted on the above figure are as follow:

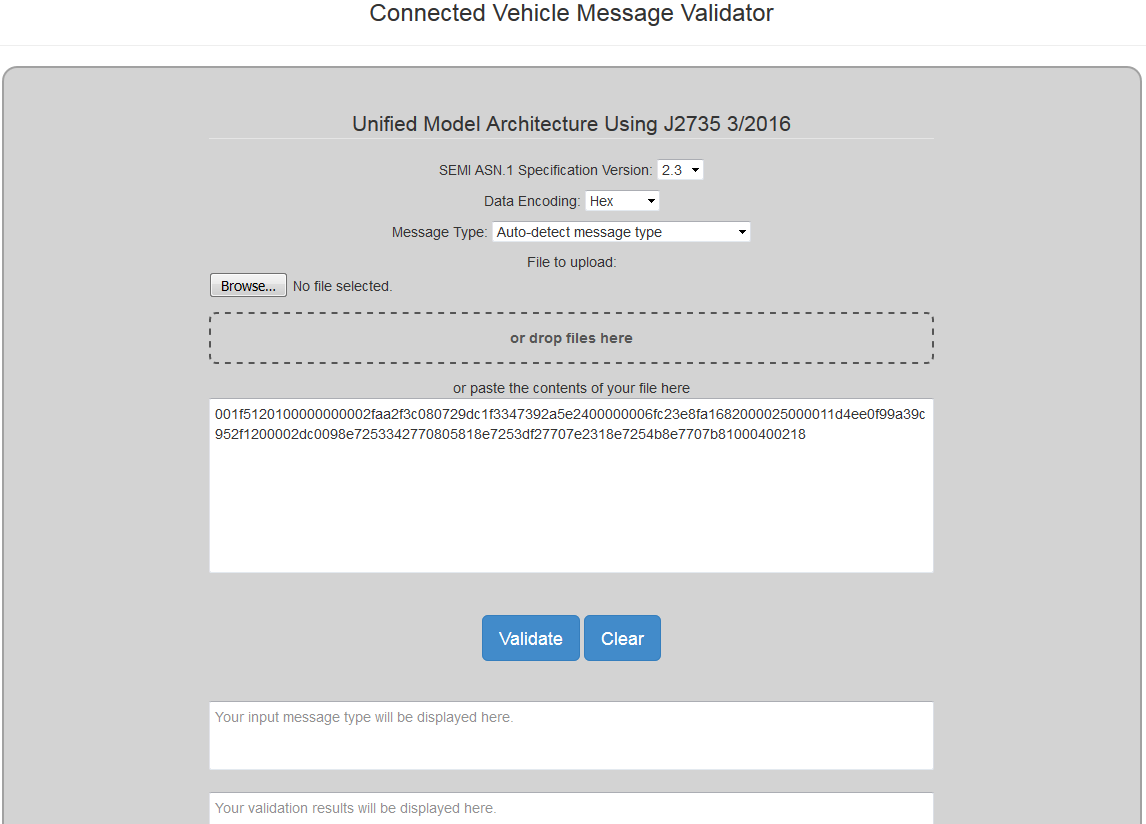
* Red for speeds less than 5 mph
* Yellow for speeds greater than or equal to 5 mph and less than 21 mph
* Green for speeds greater than or equal to 21 mph

Intersection data from traffic signals appear as color-coded squares by lane plotted at the geospatial coordinated provided in the metadata. The color of the square corresponds to the state of the traffic signal for each lane.

A single pull-down menu, the Map Tools menu, supports operation of the application. This menu allows users to start and stop the application, clear all data from the map, and to enable or disable the map layer contain the fixed-location RSUs. The Show/Hide Legend function toggles between whether or not the legend in the upper right corner of the screen shot is visible or not. The Area Selection menu option will open a dialog which allows a user to choose a custom geographic area in which to display SDC data. By default, the UI shows the greater Detroit area. The user has the option to switch to other pre-defined geographic areas or to specify a custom rectangular area defined by latitude and longitude of the rectangle corners.

#### Message Validator

This web application offers a way for third party integrators to check if their Connected Vehicle message is encoded and formatted correctly before sending to the SDC and SDW. The Message Validator tool is used to check UPER/ASN.1/HEX versions of messages for accuracy against the specifications and standards before depositing into a warehouse.



**Figure 10- Message Validator Application**

# 4 Running the Systems

## 4.1 Initiate Forwarders

### 4.1.1 Start the Virtual Machines (VMs)

To initiate the forwarder, bring up a VM for SDC/ORDS Forwarder in Rackspace, if it does not exist already. **Note the IPv4 address of the virtual machine (VM) as it will be required to configure the DE Transport nodes later in the Building the System, System Builder section of this manual.** To log into the Rackspace account go to <https://mycloud.rackspace.com/>. A Rackspace account is required.

### 4.1.2 Install the Software

To install the SDC forwarder software and scripts on the SDC/ORDS forwarder, create the following directories: /usr/local/cvData23 and /usr/local/cvData23\_secure. Upload fedgov-cv-apps-1.0.0-SNAPSHOT-jar-with-dependencies.jar, runcvDataNOSCREEN.sh, runcvData.sh, and the settings.properties file associated with the SDCForwarder-Nonsecure to the /usr/local/cvData23 directory. Upload fedgov-cv-apps-1.0.0-SNAPSHOT-jar-with-dependencies.jar, runcvDataNOSCREEN.sh, runcvData.sh, and the settings.properties file associated with the SDCForwarder-Secure to the /usr/local/cvData23\_secure directory.

To start the forwarder processes and setup the forwarder auto restart process on the SDC/ORDS forwarder, upload cvDataV23.conf and cvDataV23\_secure.conf to the /etc/init folder. Then run the following commands: “sudo start cvDataV23” and “sudo start cvDataV23\_secure”.

To install the SDW forwarder software and scripts on the SDW forwarder, create the following directories: /usr/local/SDWv23 and /usr/local/SDWv23\_secure. Upload fedgov-cv-apps-1.0.0-SNAPSHOT-jar-with-dependencies.jar, runSDW23NOSCREEN.sh, runSDW23.sh, and the settings.properties file associated with the SDWForwarder-Nonsecure to the /usr/local/SDWv223directory. Upload fedgov-cv-apps-1.0.0-SNAPSHOT-jar-with-dependencies.jar, runSDW23\_secureNOSCREEN.sh, runSDW23.sh, and the settings.properties file associated with the SDWForwarder-Secure to the /usr/local/SDWv23\_secure directory.

To start the forwarder processes and setup the forwarder auto restart process on the SDW forwarder, upload SDWv23.conf and SDWv23\_secure.conf to the /etc/init folder. Then run the following commands: “sudo start SDWv23” and “sudo start SDWv23\_secure”.

To install the ORDS forwarder software and scripts on the SDC/ORDS forwarder, create the following directories: /usr/local/cvDistribution23 and /usr/local/cvDistribution23\_secure. Upload fedgov-cv-apps-1.0.0-SNAPSHOT-jar-with-dependencies.jar, runcvDistributionNOSCREEN.sh, runcvDistribution.sh, and the settings.properties file associated with the ORDSForwarder-Nonsecure to the /usr/local/cvDistribution23 directory. Upload fedgov-cv-apps-1.0.0-SNAPSHOT-jar-with-dependencies.jar, runcvDistributionNOSCREEN.sh, runcvDistribution.sh, and the settings.properties file associated with the ORDSForwarder-Secure to the /usr/local/cvDistribution23\_secure directory.

To start the forwarder processes and setup the forwarder auto restart process on the SDC/ORDS forwarder, upload cvDistV23.conf and cvDistV23\_secure.conf to the /etc/init folder. Then run the following commands: “sudo start cvDistV23” and “sudo start cvDistV23\_secure”.

## 4.2 Building the Systems (each system)

### 4.2.1 Upload Plugins

Please refer to the plugins tab of *CVCS Configuration Report 20170912.xlsx*[[1]](#footnote-1) document for the full list of plugins required for the CVCS DigitalEdge Systems. Ensure that all the plugins have been uploaded into DigitalEdge using the DigitalEdge Management Console. Refer to the DigitalEdge Operations Guide[[2]](#footnote-2), Chapter 5, section *Managing the component repository* for instructions on uploading plugins to DigitalEdge.

### 4.2.2 Setup Data Models

For each CVCS system to be built (SDC, ORDS, etc.), please refer to the corresponding named System tab of the *CVCS Configuration Report 20170912.xlsx* document for the list of Data Models required for that system. Ensure that all required Data Models are uploaded to DigitalEdge. For instructions on uploading Data Models, refer to the DigitalEdge Configuration Guide[[3]](#footnote-3), Chapter 4, section *Upload and Download Data Models Between System.* After all required Data Models have been uploaded; open the Data Model using the DigitalEdge Data Model Editor to ensure that all Field Types and Translations match what is listed in the *CVCS Configuration Report 20170912.xlsx* document for each Data Model. Please refer to the DigitalEdge Configuration Guide3, Chapter 4, *Defining the Data Model* for detailed information about working with Data Models in DigitalEdge.

### 4.2.3 Build the System

For each CVCS system to be built, please refer to the corresponding named System tab of the *CVCS Configuration Report 20170912.xlsx* document for the list of required plugins and system build parameters. For instructions on configuring and building a system, please refer to the DigitalEdge Configuration Guide3, Chapter 5, *Building the System*. Please note that the systems should be built in the following order: SDC, ORDS, SDW, and finally SDPC.

Most plugin parameter values are static and can be entered into DigitalEdge System Builder exactly as they are shown in *CVCS Configuration Report 20170912.xlsx*, but the following values are dynamic and must be updated in DigitalEdge prior to building each system.

Table 1. Dynamic Plugin Parameter Values

|  |  |  |  |
| --- | --- | --- | --- |
| **System** | **Plugin** | **Field** | **Value** |
| ORDS | CvUDPTransportService | forwader-host-name | IPv4 DNS address of SDC/ORDS Forwarder |
| ORDS | QueryProcessor | bundle-forwarder-host | IPv4 DNS address of SDC/ORDS Forwarder |
| ORDS | QueryProcessor | mongo-server-host | Private IP of SDW mongodb.standalone |
| ORDS | ObjectDiscovery | bundle-forwarder-host | IPv4 DNS address of SDC/ORDS Forwarder |
| ORDS | Object Discovery | mongo-server-host | Private IP of SDW mongodb.standalone |
| ORDS | ObjectRegistrar | bundle-forwarder-host | IPv4 DNS address of SDC/ORDS Forwarder |
| ORDS | ObjectRegistrar | wh-sdc-host-ipv4 | IPv4 DNS address of SDC/ORDS Forwarder |
| ORDS | ObjectRegistrar | wh-sdc-host-ipv6 | IPv6 DNS address of SDC/ORDS Forwarder |
| ORDS | ObjectRegistrar | wh-sdw-host-ipv4 | IPv4 DNS address of SDW Forwarder |
| ORDS | ObjectRegistrar | wh-sdw-host-ipv6 | IPv6 DNS address of SDW Forwarder |

### 4.2.4 Start the System

After building a system, go to the Management Console, find the newly built system and press the start button. The systems cannot be started simultaneously, start one system at a time in the following order: SDC, then ORDS, then SDW, and finally SDPC. For instructions on using Management Console and starting a system, please refer to the DigitalEdge Operations Guide2, Chapter 5, *Managing DigitalEdge from the Management Console*.

### 4.2.5 Verify

When starting subsequent systems verify that the status of the system that was just started is OK before starting the next system one. For details on system status, please see the DigitalEdge Operation Guide2, Chapter 6, *System Monitoring*.

## 4.3 Complete Forwarders

### 4.3.1 Configure Addresses

The settings.properties files on the Rackspace forwarders now have to be updated with the IP addresses of the Transport nodes of the new CVCS DE systems.

On the SDC/ORDS forwarder update the /usr/local/cvDistribution\*/settings.properties files. Set the “TransportIP” and “TransportIPPool” to the new public IP Address of the ORDS Transport node.

On the SDC/ORDS forwarder update the /usr/local/cvDistribution\*/settings.properties files. Set the “TransportIP” and “TransportIPPool” to the new public IP Address of the ORDS Transport node.

On the SDW forwarder update the /usr/local/ SDW\*/settings.properties files. Set the “TransportIP” and “TransportIPPool” to the new public IP Address of the SDW Transport node.

After all settings.properties have been updated, restart the Forwarder VMs for the changes to take effect. The forwarders are configured to startup on boot.

# 5 Monitoring

## 5.1 Management Console and System Monitor

To manage and monitor the DigitalEdge systems, please refer to the DigitalEdge Operations Guide2 Chapter 5, *Managing DigitalEdge from the Management Console* and Chapter 6, *System Monitoring.*

## 5.2 Warehouse Tools

The *Warehouse Tools Set* allows a user to subscribe, query, and deposit messages into the warehouses. This also includes a mapping application that displays vehicle and signal information in real-time. The tools provide a way to send a unique test message through the systems to a specified warehouse. The tools can then be used to query for the test message to determine if a part of the data path is down, or if data is flowing completely through to the end. The tools themselves rely on a Websocket connection to the system. No connection is indicated by a red dot next to the tool’s name in the navigation panel at the top of the page, and a successful connection is indicated by a green dot. If the tools are unable to connect prior to deposit or query, that may indicate an issue with the Websocket server components of the systems.

## 5.3 Test Routine: Data Type ISD, VSD, etc.

To test ISD and VSD messages go to the Warehouse Tools Set and deposit a new unique message in the Data Deposit tab. For verification, check if the message has gotten through either by doing a Warehouse Query, an SDC Subscription, or just check on the Map Application.

To test TIM messages go to the Warehouse Tools Set and deposit a new unique message (Data Deposit). Check if the message has gotten through by doing a Warehouse Query.

## 5.4 Notification Scheduling

The System Monitor application will monitor message flow via test messages and detect downtime/system failures. When a failure is detected it will automatically create a Samanage incident report and distribute alert emails to a configurable list of users defined in the System Monitor’s configuration. For a detailed description of the System Monitor and it’s configuration options, please refer to the System Monitor Operations Manual. Upon recovery, it will send recovery emails to a different configurable list of emails. On the first day of every month, a report of components uptimes will be generated for the previous month and emailed to a third configurable list of emails.

# 6 Updating OSS Licensing

1. Install new OSS liscense (“ossinfo” file) into the base folder where the OSS executable is located
2. Restart OSS Studio
3. Compile the ASN to Java using OSS Studio
4. Delete the old compiled code from the fedgov-cv-asn1-generated-j2735 Git repository: src/main/java/gov/usdot/asn1/generated/j2735
5. Copy the new compiled code to the same location in the Git repository
6. Build the fedgov-cv-asn1-generated-j2735 project
7. Build the rest of the projects that have dependencies on the generated code in the following order:
   1. fedgov-cv-asn1
   2. fedgov-cv-common
   3. fedgov-cv-parser-j2735
   4. fedgov-cv-transport-udp

The build process will generate the DigitalEdge plug-ins for each of these projects.

1. Upload the plug-ins to the Plug-Ins section of the DE Management Console. This will overwrite the previous version of the plug-ins.
2. In the System Builder, build a new system for the ORDS system.
3. Start the newly built system using the Management Console.
4. When the new system is started, transfer all publically facing ORDS domain names to point to the new systems using the AWS Route53 service.

# 7 Troubleshooting

## 7.1 Ticketing System

If a problem is detected, visit Samanage portal and view the current system status by opening the incident report.

<https://cvcs.samanage.com/login>

## 7.2 Uptime Dashboard

Go to CVCS monitoring dashboard to view the system status. When this is available, this should be the first step in troubleshooting. In the meantime, DigitalEdge console, and AWS console and Rackspace console should be used for reference.

## 7.3 DigitalEdge Console

One way to monitor the state of each system is to go to the Tenant Console <https://default.tms.aws-dev.deleidos.com/tenantconsole/>. Verify that the status of a system is OK; if in the Warning state please refer to the DigitalEdge Operations Guide2 under the sections *System Monitoring->Viewing system status* and *Troubleshooting*.

Another way to monitor each system is in DigitalEdge System Monitor <https://default.tms.aws-dev.deleidos.com/systemmonitor/>. Generally if there is zero for the Average Throughput or a very steep incline, the system is in a bad state. For more information please refer to the DigitalEdge Operations Guide2 under the section *System Monitoring->Checking system metrics*

## 7.4 AWS Console

In AWS Console <https://console.aws.amazon.com/console/home>, instances can be monitored by checking the Instance State of each Instance to verify that it is running. If an instance is in the warning state or if there is a scheduled downtime please refer to either the support center <https://console.aws.amazon.com/support/home?region=us-east-1#/> or the documentation for ec2 instances <https://aws.amazon.com/documentation/ec2/>.

## 7.5 Rackspace Console

Login to Rackspace console <https://mycloud.rackspace.com/> to verify that all of the servers are green. If there are any warning signs (not showing up green) or if there is any scheduled downtime, please click on the *Support* link on the console.

## 7.6 Maintenance

For maintenance of the systems, please refer to Appendix B: Maintenance Plan.

# 

# Appendix A: List of Acronyms

|  |  |
| --- | --- |
| ARC-IT | Architecture Reference for Cooperative and Intelligent Transportation |
| AWS | Amazon Web Services |
| CVCS | Connected Vehicles Core Services |
| DB | Database |
| DSRC | Dedicated Short-Range Communications |
| EBS | Elastic Block Store |
| EC2 | Elastic Compute Cloud |
| IEEE | Institute of Electrical and Electronics Engineers |
| IP | Internet Protocol |
| IPv4 | Internet Protocol version 4 |
| IPv6 | Internet Protocol version 6 |
| ISD | Intersection Data |
| JMS | Java Messaging Service |
| RSU | Roadside Unit |
| SAE | Society of Automotive Engineers |
| SCP | Secure Copy Protocol |
| SDC | Situation Data Clearinghouse |
| SDK | Software Development Kit |
| SDW | Situation Data Warehouse |
| SSH | Secure Socket Shell |
| USDOT | United States Department of Transportation |
| UPD | User Datagram Protocol |
| VM | Virtual Machine |

# Appendix B: Maintenance Plan

## B.1 Software Updates

The software systems running within CVCS are operational systems. Therefore, determining reasonable software updates is and should be biased toward ensuring maximum uptime.

### B.1.1 Updating Plugins

Before updating plugins, please note that plugins are java based software components and familiarity with Java will be required.

If an existing DigitalEdge plugin needs to be updated, or a new plugin needs to be created, refer to the DigitalEdge Software Development Kit (SDK) Guide for details on developing and updating a plugins. Once the updated or new plugin is ready it must be uploaded to the DigitalEdge repository before it can be used in a DigitalEdge system. Refer to the DigitalEdge Operations Guide[[4]](#footnote-4), Chapter 5, section *Managing the component* repository for instructions on uploading plugins.

### B.1.2 Updating Software Libraries Directly

Sometimes a software update is required for a library that isn’t a DigitalEdge plugin. In these situations, it is necessary to access the DigitalEdge nodes directly. The nodes are accessible via Secure Socket Shells (SSH). In order to do so, the security group for the instance needs to be opened via the AWS Console for the Internet Protocol (IP) of the machine accessing the instance[[5]](#footnote-5) and a copy of the public key from the key pair used to start to instance must be obtained. To upload new software onto the instance, any Secure Copy Protocol (SCP) client can be used. Once the instance is accessed and the new software library files are uploaded, the software update can be performed.

The location of the software depends on which software is being updated. Any DigitalEdge software libraries can be found in the sub-directories under /usr/local/rtws/\*. The libraries will be located in “lib” directories and there will be scripts to stop and start the various services located in corresponding “bin” directories. On transport instances, DigitalEdge transport specific software is located in /usr/local/rtws/transport/\*. DigitalEdge datasink and enrichment specific software is located on their corresponding ingest instances under the directory /usr/local/rtws/ingest/\*. Non-DigitalEdge datasink software locations depend on how the instance is configured to install the software. For example, a MongoDB Datasink instance is configured to store the Database (DB) on a mounted volume under /mnt/rdafs/mongodb/. JMS instances key software is Apache ActiveMQ located at /usr/local/apache-activemq/\* with scripts to start and stop the service located under /usr/local/rtws/commons-core/bin/boot/.

In order to perform the update, after locating the correct software library, the service needs to be stopped, the correct library files need to be replaced with new copies, and the service needs to be started again. It is recommended to create backup copies of the original library files before replacing them with the new version. This will ensure old versions can be restored if there is an issue with the new software.

### B.1.3 Rebuild and Cycle Affected System

After new or updated plugins have been uploaded or a virtual machine (VM) has been retired, the effected DigitalEdge system(s) must be rebuilt and restarted for the software updates to take effect. To do this, first open the target system in System Builder. Add and/or update the required plugins (including plugin parameters) for the system and give the system a new unique name. This system, with the software updates and new name, will run in parallel with the old system until it can be confirmed the changes were successful. Build the new system and start it in the Management Console. For detailed instructions on building and starting a DigitalEdge system, please refer to the DigitalEdge Configuration Guide[[6]](#footnote-6), chapter 5, *Building the System*.

### B.1.4 Update Forwarders

After starting the new system, the respective forwarder’s settings.properties files need to be updated. Refer to the *CVCS Configuration Report 20170912.xlsx*1 file for the names and locations of these files. Change the TransportIP in each of the files from the public IP of the Transport of the old system to the public IP of the Transport of the new system. After doing this, restart the forwarder. To verify that data is getting through, use the webtools (such as Warehouse Tools Set). It is recommended to track Intersection Data (ISD) messages through the system to verify the software update has worked. If successful the old system can be stopped.

## B.2 Continued Operations & Infrastructure Maintenance

### B.2.1 AWS VM Retirement

Periodically, AWS needs to retire (shutdown, restart, etc.) 1 or more of the virtual machines (nodes) that make up a DigitalEdge system. If there is ever an issue with an AWS node, an email will be sent to the AWS account administrator about the node in question. In addition to the email, EC2 instances which require maintenance will have a scheduled event listed for them on the AWS Console. To view which EC2 instances have scheduled events, please refer to the AWS documentation[[7]](#footnote-7). When this happens, use the DigitalEdge or AWS Console to determine the DigitalEdge system(s) that will be affected. Rebuild and restart the affected as discussed in sections B.1.3 Rebuild and Cycle Affected System and B.1.4 Update Forwarders.

There is also a way to restart the EC2 instance without restarting the whole system. To do so, first go to the AWS Console. Right click on the EC2 instance that requires restarting. Hover over the “Instance State” context menu option and select the “Terminate” option from the menu. While the instance is terminating, open an SSH terminal connected to the system’s Master node. When connected to the master node, run “ps aux | grep ClusterManager” to determine the Process Identifier number. Return to the AWS Console and wait for the EC2 instance that is terminating to reach a state of “Terminated”. When it is listed as terminated, return to the Master node SSH terminal and run “sudo kill -9 <PID>”, where <PID> is the process identifier number from the “ps aux …” command that was ran. At this point, the Master process should be down. Next change directories to the Master’s scripts: “cd /usr/local/rtws/master/bin”. Restart the Master process: “sudo –u rtws ./start\_master.sh”. The Master process will start and attempt to connect to any existing nodes in the system. This will re-establish a connection to the rest of the system. When the Master process looks for the terminated instance, it will detect that it is missing and start a new EC2 instance part to replace the old. You will be able to see the new EC2 instance starting on the AWS Console; it may take a bit for the EC2 instance to start. Once the EC2 instance starts, it will also take some time for it to download and start the DE software.

## B.3 Periodic Diagnostics

There are a few places where diagnostics can be done for a running system. To view the status the DigitalEdge system as a whole, there are two tools available: the DigitalEdge Management Console and the DigitalEdge System Monitor. To view the health of individual instances that compose the systems, the AWS Console can be used.

### B.3.1 DigitalEdge Management Console

The DigitalEdge Management Console provides a two ways to perform diagnostics for a system. The first is the system view list which provides a quick snapshot of a running system, including what process groups are running, the number of instances and types, and the number of volumes and sizes. The second tool is the log viewer which allows access to the logs of each node in a DigitalEdge system. The use of both of these is described in the DigitalEdge Operations Guide4, Chapter 6, in sections *Viewing system status* and *Checking log files*.

### B.3.2 DigitalEdge System Monitor

The DigitalEdge System Monitor is a dynamic console tool for monitoring system health and potential problems in real time by visually depicting system activity and resource scaling. It provides the capability to see the throughput of ingest instances, the amount of data on the internal system queues, and the current storage capacity of the datasink instances. The use of the DigitalEdge System Monitor is outlined in the DigitalEdge Operations Guide4, Chapter 6, section *Checking system metrics*.

### B.3.3 AWS Console

The AWS Console can be used to check the health and status of the individual instances that compose the DigitalEdge systems. Using the AWS Elastic Compute Cloud (EC2) service, a web service that provides secure, resizable compute capacity in the cloud, each instance can be selected in order to show: its current state, a detailed description of the instance and its attributes, status checks, monitoring, and much more. Detailed documentation on how to use AWS EC2 is available through AWS EC2 Documentation[[8]](#footnote-8). For ease of system use, the AWS Elastic Block Store (EBS) is also utilized to provide persistent block storage volumes for use with AWS EC2 instances.

## B.4 Backup Procedure for the Situation Data Warehouse

In the SDW, the technology used to store the data is the NoSQL document database, MongoDB. The database is stored on RAIDed EBS volumes attached to the DigitalEdge datasink instance. Since the database is completely stored on volumes separate from the instance, this makes using an AWS EBS snapshot the ideal way to create a backup of the data.

### B.4.1 Volume Snapshots

Creating a volume snapshot is an easy process using the AWS console and is documented by AWS EC2 Documentation[[9]](#footnote-9). Since multiple volumes are used and RAIDed by the DigitalEdge system, it is necessary to complete the snapshot process for each volume on the instance. Once the process is completed, the snapshots will remain stored by AWS indefinitely until manually deleted from the AWS EC2 Console.

## B.5 Maintenance Issues

Maintenance issues are currently reported by customers using the ticketing system from the Samanage Service Desk[[10]](#footnote-10). The service desk is accessible from the “Support” link in the top right corner of the Connected Vehicles Tool Library page. Once a ticket is submitted, it will be addressed as quickly as possible and closed upon resolution.

## B.6 Security Monitoring and Patching

### B.6.1 Software

Due to the nature of Connected Vehicle environments, security must be built into every facet of a Connected Vehicle network. The SDC and SDW were developed using the security libraries, recommended by the USDOT and industry experts, in the following industry standards: IEEE 1609.2-2016 and SAE J2735. Because security is built directly into the software, any patches or updates to the security of the system can be applied using the, above, instructions for software updates in Section B.1.

### B.6.2 Instance Access

Security for the access of the DigitalEdge instances is provided by multiple means. When a new instance is launched, it is done so using the public key from an RSA key pair. This public key is controlled by AWS and not accessible to anyone else. In order to access an EC2 instance launched using the public key, a copy of the private key that was generated with the public key must be used for authentication. Without the private key, it is not possible to access the instance. In addition to the use of key pairs, AWS implements Security Groups. A Security Group is a virtual firewall that can be used to control traffic to instances. The Security Group maps IPs, IP ranges, or other Security Groups to specific port or port ranges. Each instance must be part of a Security Group, and only specified IPs or other instances in specified Security Groups will be allowed to access the instance on the specified ports. Therefore, the Security Group needs to be maintained to only ensure specific access to the instance. Description on how to configure the Security group can be found in AWS Documentation5.

# Appendix C: DigitalEdge™

DigitalEdge is a cloud-based platform for creating big data solutions. It offers:

* *Scalable, pluggable, flexible, pre-integrated, data management platform* to enable quick implementation and simplified management of near real time (or batch) big data applications
* *Software* that leverages the power and dynamism of cloud computing to support rapid out-of-the-box creation and integration of IT systems
* *Data enrichment engine* that standardizes, integrates, and enhances diverse data sources to clean up imperfect data and transform it into information, rich with meaning.
* *Administration tools* that support rapid creation and spawning of systems that perform pre- ingest and enrichment

## C.1 Team

The DigitalEdge platform was developed and is maintained by Leidos. The DigitalEdge Team is comprised mostly of software engineers, based in the National Security Sector of Leidos, and contains a depth of experience ranging from college interns to 30+ years creating large scale enterprise systems.

The DigitalEdge Team prefers a Linux based environment, leveraging Docker in Amazon Web Services, Microsoft Azure, Eucalyptus and Rackspace. Favorite programming languages include Java and AngularJS.

## C.2 Licensing

DigitalEdge is provided for use under the Apache 2.0 Open Source License (https://raw.githubusercontent.com/deleidos/digitaledge-platform/master/LICENSE). The source information for DigitalEdge is available on GitHub at [https://github.com/deleidos/digitaledge-platform](https://webmail.leidos.com/owa/redir.aspx?SURL=hBHXcWQeF-nMmF0Jz8jDe4XkE0szysv0nJzt3mHNg9_KDPaNC-rSCGgAdAB0AHAAcwA6AC8ALwBnAGkAdABoAHUAYgAuAGMAbwBtAC8AZABlAGwAZQBpAGQAbwBzAC8AZABpAGcAaQB0AGEAbABlAGQAZwBlAC0AcABsAGEAdABmAG8AcgBtAA..&URL=https%3a%2f%2fgithub.com%2fdeleidos%2fdigitaledge-platform).

## C.3 Documentation

The below documentation may have been referenced in this manual.

* DigitalEdge Overview Guide
* https://github.com/deleidos/digitaledge-platform/raw/1.2.1/docs/Overview\_Guide.pdf
* DigitalEdge Operations Guide
  + https://github.com/deleidos/digitaledge-platform/raw/1.2.1/docs/Operations\_Guide.pdf
* DigitalEdge Configuration Guide
  + https://github.com/deleidos/digitaledge-platform/raw/1.2.1/docs/Configuration\_Guide.pdf

Additional documentation can be found at:

https://github.com/deleidos/digitaledge-platform/tree/1.2.1/docs

1. <https://gitlab.com/connectedvehicles/fedgov-cv-documents/blob/master/CVCS%20Configuration%20Report%2020170912.xlsx> [↑](#footnote-ref-1)
2. <https://github.com/deleidos/digitaledge-platform/raw/1.2.1/docs/Operations_Guide.pdf> [↑](#footnote-ref-2)
3. <https://github.com/deleidos/digitaledge-platform/raw/master/webconsole-help/src/main/documentation/Configuration_Guide.pdf> [↑](#footnote-ref-3)
4. <https://github.com/deleidos/digitaledge-platform/raw/1.2.1/docs/Operations_Guide.pdf> [↑](#footnote-ref-4)
5. <http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/authorizing-access-to-an-instance.html> [↑](#footnote-ref-5)
6. <https://github.com/deleidos/digitaledge-platform/raw/1.2.1/docs/Configuration_Guide.pdf> [↑](#footnote-ref-6)
7. <http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/monitoring-instances-status-check_sched.html#viewing_scheduled_events> [↑](#footnote-ref-7)
8. <https://aws.amazon.com/documentation/ec2/> [↑](#footnote-ref-8)
9. <http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ebs-creating-snapshot.html> [↑](#footnote-ref-9)
10. <https://cvcs.samanage.com/> [↑](#footnote-ref-10)