U.S. Department
of Transportation

Federal Highway
Administration

Southeast Michigan Test Bed

Advanced Data Capture Field Testing

Operational Data Environment

Technical Report

Version 1.0

July 21, 2016

NOTE: This is a highly technical document describing in detail the set-up and deployment of an early-release of prototype software. The audience for this document is users with extensive knowledge of UNIX/LINUX operating systems, manual configuration of operating and storage environments, Java, Eclipse IDE, Maven, Spark Hadoop and Python. Also note that command line execution is used throughout these set-ups.

Booz | Allen | Hamilton

8232 Greensboro Drive
McLean, VA 22102
Tel: (703) 902-5000

This page left intentionally blank

ACCEPTANCE / APPROVAL PAGE

// // Name Author’s Title	Prepared by	Date
// // Name Quality Assurance	Reviewed by	Date
// // Name Project Manager	Approved by	Date

DOCUMENT CHANGE HISTORY

Version Number	Date	Description
0.1	01/26/2016	Initial Draft
1.0	6/10/2016	Converted to Task 5 Technical Report

Table of Contents

List of Tables

List of Figures

1 Introduction

The Southeast Michigan (SEMI) Operational Data Environment (ODE) is an early release of software that is being developed to acquire connected vehicle data, conduct data valuation, aggregation and integration functions, and provide it to clients that have connected to it. While a prior white paper described an end-state for the ODE, the current state of the software remains in research & development, and is meant to support parallel research with CV Pilots and Connected Cities. This document details how this research ODE software can be deployed in a hosting environment, and how client applications can interface with it to acquire Connected Vehicle (CV) data, in particular, Vehicle Situation Data and infrastructure data such as Intersection Situation Data.

It is noted that this is a highly technical document describing in detail the set-up and deployment of ODE early-release prototype software. The audience for this document is users with extensive knowledge of UNIX/LINUX operating systems, manual configuration of operating and storage environments, Java, Eclipse IDE, Maven, Spark Hadoop, Python, and other technologies described herein. Additionally, much of the activities described here involved command-line interactions. When and where questions arise in performing the activities described here, those questions should be directed to Ariel Gold at FHWA [ariel.gold@dot.gov].

1.1 Reference documents

Documents used as the basis for creation of this document:

- PWS for Southeast Michigan Testbed Advanced Data Capture Field Testing
- Southeast Michigan Testbed Advanced Data Capture Field Testing Task 3 Whitepaper v1.1
- Southeast Michigan Testbed Advanced Data Capture Field Testing 4 Concept of Operations v1.0
- Southeast Michigan Testbed Advanced Data Capture Field Testing Task 4 Field Test Plan v1.0
- Southeast Michigan Testbed Advanced Data Capture Field Testing Task 4 System Design Document v3.1
- Southeast Michigan Testbed Advanced Data Capture Field Testing Lessons Learned Repository v3.0

1.2 Audience

This document in intended for use by software development teams and specialists with the professional knowledge and experience in the following technologies:

- Amazon Web Services
- Apache Hadoop, Hortonworks ® distribution
- Apache Kafka
- Apache Spark

- Apache Tomcat Application Server
- Apache Web Server
- ASN.1 Standards
- Build automation tools, particularly Maven
- Cyber Security
- Integrated Development Environments (IDE), particularly Eclipse
- Java Enterprise Edition (JEE)
- Java programming language
- Liferay Portal
- Linux Administration
- Python programming language
- Software Configuration Management tools, particularly Git
- Web Application development
- Web-based API

1.3 Glossary

Term	Description
apt-get	A package handling utility https://wiki.debian.org/apt-get
API	Application Program Interface
ASN.1	Abstract Syntax Notation One (ASN.1) is a standard and notation that describes rules and structures for representing, encoding, transmitting, and decoding data in telecommunications and computer networking
AWS	Amazon Web Services https://aws.amazon.com/
CAS	Central Authentication Service
CloudFormation	CloudFormation is a service provided by AWS that gives developers and systems administrators an easy way to create and manage a collection of related AWS resources, provisioning and updating them in an orderly and predictable fashion.
CV	Connected Vehicles
DDS	Data Distribution System
DSRC	Dedicated Short Range Communications
EC2	Elastic Computing service provided by AWS
EVSD	Enhanced Vehicle Situation Data
Git	Git is a free and open source distributed version control system designed to handle everything from small to very large projects with speed and efficiency. https://git-scm.com/
Hadoop	The Apache™ Hadoop® project develops open-source software for reliable, scalable, distributed computing.
ITS	Intelligent Transportation Systems

Yarn	Hadoop Yarn is a framework for job scheduling and cluster resource management.
HDFS	Hadoop Distributed File System is a distributed file system that provides high-throughput access to application data.
ISD	Intersection Situation Data
JDK	Java Development Kit
JPO	Joint Program Office
JRE	Java Runtime Environment
JVM	Java Virtual Machine
Kafka	Apache Kafka is publish-subscribe messaging rethought as a distributed commit log.
Liferay Portal	Liferay Portal is an enterprise web platform for building business solutions.
POJO	Plain Old Java Object
PWS	Performance Work Statement
SAE	SAE International is a global association of more than 128,000 engineers and related technical experts in the aerospace, automotive and commercial-vehicle industries.
j2735	This SAE Standard specifies a message set, and its data frames and data elements specifically for use by applications intended to utilize the 5.9 GHz Dedicated Short Range Communications for Wireless Access in Vehicular Environments (DSRC/WAVE, referenced in this document simply as "DSRC"), communications systems.
SCP	Secure Copy
SDC	Situation Data Clearinghouse (US DOT)
SDPC	Situation Data Processing Center
SDW	Situation Data Warehouse
SFTP	Secure File Transfer Protocol
Spark	Apache Spark is a fast and general engine for big data processing, with built-in modules for streaming, SQL, machine learning and graph processing.
SPaT	Signal Phase and Timing
SSD	Solid State Disk
SSH	Secure Shell
SSL	Secure Socket Layer
TSD	Traveler Situation Data
US DOT	Unites States Department of Transportation
Vagrant	Vagrant enables users to create and configure lightweight, reproducible, and portable development environments. https://www.vagrantup.com
VM	Virtual Machine
VPC	Virtual Private Cloud
WebSocket	WebSocket is designed to be implemented in web browsers and web servers, but it can be used by any client or server application. The WebSocket Protocol is an independent TCP-based protocol. Its only relationship to HTTP is that its handshake is interpreted by HTTP servers as an Upgrade request.

yum	Yum is an automatic updater and package installer/remover for rpm systems. http://yum.baseurl.org/
ZooKeeper	Apache ZooKeeper is a centralized service for maintaining configuration information, naming, providing distributed synchronization, and providing group services.

2 Development Environment

2.1 Java Development Tools

The ODE team uses Java as the primary programming language.

Tools:

- Java
- Eclipse IDE
- Maven

2.1.1 Java

Install Java Development Kit (JDK) 1.8

http://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-2133151.html

2.1.2 Eclipse IDE

Download and install Eclipse.

https://eclipse.org

Configure Eclipse to use Java 1.8 JDK. Local installation of Tomcat can integrate with Eclipse and can help with prototyping or debugging the application.

2.1.3 Maven

Download and install Maven: https://maven.apache.org/

Maven is a build and dependency management tool. Newer versions of Eclipse include Maven plugins, which may include embedded versions of Maven. However, if you wish to build the ODE applications outside of Eclipse, or wish to use an IDE other than Eclipse, you will need maven in order to develop and build the ODE application code.

2.2 Python Development Tools

While the ODE application is built in Java, Python has been used to build Simulated Client applications for testing and it is used to construct the ODE CloudFormation script that generates the ODE Virtual Private Cloud (VPC). This section describes some of the basic steps in setting up some Python development tools

2.2.1 Installing Python

Python can be installed on Windows by downloading an MSI installer from the Python website. Python related scripts and programs that have been developed for the ODE use Version 2.7.x. Python can be downloaded from https://www.python.org/downloads/release/python-2710/

Please select the Installer version that is appropriate for your Operating System.

Note:

Python 2 is the preferred version to use as it has many 3rd party modules created for it than Python 3. Also we should use Python 2.x because Python 2.6.x or 2.7.x is installed by default on the Linux servers used in our AWS environment or our local development Vagrant VMs.

If you wish to execute Python scripts / programs from your Windows command line, you will need to update your System Path environmental variable to include the Python executable file.

2.2.2 IDE

While Python files can be edited using a standard text editor, development can be made easier by using an IDE. One such IDE is PyCharm which can be downloaded from https://www.jetbrains.com/pycharm/download/

2.2.3 Dependency Management

3rd Party python modules can be installed in a number of ways depending on the Module and the Operating System. Most 3rd party modules can be installed using the Python package install program called pip.

On Windows computers, 3rd party packages can be installed by using the following command:

```
python -m pip install <package-name>
```

On Linux Computers, 3rd party packages can be installed by using the following command:

sudo pip install <package-name>

Pip will install the requested module and any dependencies the module.

Some modules are Python bindings to C/C++, etc libraries and may require other dependencies in order to compile the libraries and install on your system. For Linux systems, this may mean we need to install the missing development libraries through apt-get or yum. On Windows based machines, it may require you to download a pre-built version of the module.

A requirements.txt file, can be used to track multiple 3rd party modules and the versions that has been used to create test script or program. For instance, the ODE's AWS CloudFormation script and the emulated client test applications are generated by a Python script that use specific libraries (see below).

In order to run these programs on a windows machine or a Linux machine you would need to install the dependencies that these programs required. Each script/program has a requirements.txt that enumerates the 3rd party modules and the version.

```
# For Windows Machines, start a command prompt with administrative rights
python -m pip install -r path/to/requirements.txt
```

For Linux based servers, execute the command:

```
pip install -r /path/to/requirements.txt
```

The dependencies mentioned in the requirements.txt can be installed manually, however the use of requirements is purely to make development and deployment easier since you have version locked modules that you know will work with your program. Additionally, IDEs like PyCharm will detect the requirements.txt in your project working directory and will attempt to install the missing dependencies automatically.

2.2.3.1 Sample requirements.txt files

requirements.txt to Create CloudFormation Script

```
troposphere==1.1.0
boto==2.38.0
```

requirements.txt for Emulated Client Test application

```
websocket-client==0.32.0
```

2.2.4 ODE Python SDK

A python SDK under ODE configuration control is available for application developers. The SDK is a useful tool for application developers to get up and running quickly with the interface to the ODE. The SDK also facilitates easier and more reliable dissemination of updates of the ODE API and will help minimize changes to the application connectors or code as the ODE API evolves and matures. For more details about the ODE Python SDK and how to obtain the SDK, please refer to the ODE SDK Developer Guide.

2.3 Setting up a local sandbox

2.3.1 Setting up a local instance using Vagrant

You can use Vagrant to set up a local instance of the ODE operational environment in a Linux VM. Vagrant files for doing so can be found in the ODE Git repository under the path /ode/Vagrant/ode-hadoop2.3

2.3.2 To get a local instance up and running without Vagrant.

** Indicates that these need to be run in sequence for code changes to take effect on local machine.

*** Indicates things that need to be running in order to run the web application, but don't need to be restarted to see code changes.

1. Clone Stash repo via https://username@buildforge.cnsi.com:9443/scm/ode/ode-repo.git
    1. Before cloning either:
        1. download the SmartSuite pem and add it via:

           ```
           git config --global http.sslCAInfo
           ~/Documents/SmartSuiteCA.pem
           ```

        2. disable ssl verification via:

           ```
           git config --global http.sslVerify false
           ```

2. Install JDK 1.8 and set environmental variables accordingly (Noted as $JAVA_HOME for the rest of the document)
    1. For reference look at the batch file setenv.bat located in ode-repo/ode/development/apps
3. Install Apache Tomcat 7 and note the folder location (Noted as $TOMCAT_HOME for the rest of the document)
4. Install 3rd party JARs into the local MAVEN repository by executing the appropriate command line script
    1. For mac/linux/unix:
        1. chmod 755 ode-repo/ode/development/apps/mac-build-local.sh
        2. ./ode-repo/ode/development/apps/mac-build-local.sh
    2. For PC:
        1. Run the batch file: ode-repo/ode/development/apps/build-local-repo.bat
5. Run a maven clean install on the root directory (i.e.: ode-repo/ode/development/apps) – **
6. Find the war file created from the maven install and copy it to $TOMCAT_HOME/webapps **– ****
7. Create a linux VM (CentOS 7 Minimal ISO recommended: https://www.centos.org/download/)
    1. During installation ensure that your networking module is installed
8. In the VM's settings set up two adapters
    1. Adapter 1: NAT, Cable Connected (for internet connection)
    2. Adapter 2: Host-only Adapter, and create a Host-only adapter on your host machine
        1. To verify connection log into the VM and run "ip addr show"

2. And on your host machine ping the local ip of the VM referred to as $VMIP from here on out

9. In your VM run the commands to open port 80:
    1. firewall-cmd --zone=public --add-port=80/tcp --permanent
    2. firewall-cmd --reload
10. In your VM download the Bitnami Liferay stack:
    1. wget --no-check-certificate https://bitnami.com/redirect/to/48549/bitnami-liferay-6.2-6-linux-x64-installer.run
    2. chmod 755 bitnami-liferay-6.2-6-linux-x64-installer.run
    3. sudo ./bitnami-liferay-6.2-6-linux-x64-installer.run --mode unattended --base_password test --prefix /opt/bitnami
    4. To run the liferay server: sudo /opt/bitnami/ctlscript.sh start – ***
    5. To stop the liferay server: sudo /opt/bitnami/ctlscript.sh stop
11. Installing ambari management of HDP stack
    1. sudo wget -nv -O http://public-repo-1.hortonworks.com/ambari/centos7/2.x/updates/2.1.0/ambari.repo /etc/yum.repos.d/ ambari.repo
    2. sudo yum install ambari-server
    3. sudo yum install ambari-agent
    4. sudo ambari-agent start – ***
    5. sudo ambari-server setup --silent
        1. will install defaults (postgresql, java 8, etc)
    6. sudo ambari-server start – ***
    7. Note your VMs localhost name. To get this run: "curl -u admin:admin localhost:8080/api/v1/hosts"
    8. Log into the ambari server from your host computer. Go to <$VMIP>:8080. Default username:password is admin:admin
        1. If you cannot access it ensure that you have opened port 8080
        2. Also you can verify your VM IP address by running "ip addr show" in terminal
    9. Create a new cluster with whatever name you see fit.
        1. Install the minimum of the following services on the cluster
            1. HDFS
            2. MapReduce2
            3. Yarn - in settings set min container size between 256 and 512MB
            4. ZooKeeper
            5. Kafka
            6. Spark
        2. During the checks phase, if the ntpd service is required, or any service for that matter ensure that it is installed on the VM
            1. For ntpd: "sudo yum install ntp" and "service ntpd start"
12. Update your ode-web/src/main/webapp/WEB-INF/web.xml file with the following parameter values:
    1. liferay.ws.serverhost = http://$VMIP/liferay
    2. liferay.ws.companyId = 20155
        1. This is the default. To ensure this value is correct go to $VMIP/liferay and sign in with username:password as user@liferay.com:test
        2. Go to Control Panel > Configuration > Portal Instances

        3. Under Instance ID that number will go under liferay.ws.companyId
        4. This Instance ID may change based on location changes
13. Start up the tomcat server and ensure TOMCAT_JRE is pointing to $JAVA_HOME – **
    1. Log into localhost:8080/ode to view the web application (default tomcat port is 8080)

2.3.3 Checklist for starting up your local instance every day

1. Log into your VM
    1. /opt/bitnami/ctlscript.sh start
    2. ambari-agent start
    3. ambari-server start
2. Go to $VMIP/ambari
    1. Start all services
3. If a code change (if not go to step 4):
    1. mvn clean install -f ode-repo/ode/Development/apps/webapp
    2. On windows: move ode.war from ode-repo/ode/Development/apps/webapp/ode-web/target/ode.war TO $TOMCAT_HOME/webapps/
    3. On mac: cp ode-repo/ode/Development/apps/webapp/ode-web/target/ode.war $TOMCAT_HOME/webapps/
4. Regardless of code change:
    1. Start tomcat: $TOMCAT_HOME/bin/startup.sh OR windows equivalent
    2. Optional - View Tomcat logs for debugging: View $TOMCAT_HOME/logs/catalina.out in your favorite text editor
        1. Or on linux/mac: tail -n 500 -f $TOMCAT_HOME/logs/catalina.out for up to date view of log view
5. If you run into errors:
    1. Ensure that $JAVA_HOME is set correctly

2.4 Test and Demo Web UI

ODE deploys a web UI for the purpose of internal development, testing and sometimes demoing the software to the client. This section describes the design and development of the ODE Web UI application.

There are five main components to the Web UI:

1. index.html
2. deposit.html
3. style.css
4. control.js
5. upload.xhtml

2.4.1 Index.html

Index.html defines the web page for setting up subscriptions or querying data from the ODE. It sets up three divs for each of the columns in the UI. What is shown is determined by control.js.

2.4.2 Deposit.html

Deposit.html defines the web page for publishing data to the ODE. It sets up three divs for each of the columns in the UI. What is shown is determined by control.js.

2.4.3 Style.css

Style.css defines the look and feel of the components in index.html and deposit.html.

2.4.4 Control.js

Control.js is a JavaScript program that implements the front business logic. It indicates what options are shown based on control selections and controls the map and incoming requests dialogue.

2.4.5 Upload.xhtml

Used for sending data as a subscription to a specific request ID.

2.5 GIT Version Control

The ODE software is maintained and version controlled using GIT version control system.

Recommend clients:

- Tortoise Git
- Source Tree
- Git Hub Windows App
- Git Extensions

It is recommended that GIT plug-ins are installed with your IDE so that your IDE is GIT "aware".

2.5.1 SSH and SCP Clients

In order to access ODE application and Hadoop servers command line shell, you will need to install and configure an SSH client if you don't have one already.

Recommend clients:

- Putty

In order to upload or download files from ODE application and Hadoop servers, you will need to install and configure an SCP client if you don't have one already.

Recommend clients:

- WinSCP

2.6 Build and Deploy

2.6.1 ASN.1 Java API

Currently the primary sources of data for the ODE are the US DOT Situation Data Clearinghouse (SDC), Situation Data Warehouse (SDW) and Situation Data Processing Center (SDPC). These entities are collectively referred to as US DOT Data Distribution System or DDS for short. The DDS provides data to the clients in ASN.1 format which is also the format that connected vehicles and the road-side equipment transmit and deposit their data into the system. In order for the ODE to utilize the data, it must be able to decode the data from ASN.1 format into a more generic format, in this case Plain Old Java Objects (POJOs). We have acquired ASN.1 compiler and run-time libraries tool from OSS Nokalva to accomplish the decoding/encoding of data from ASN.1/POJO to POJO/ASN.1.

2.6.1.1 ASN.1 Compiler and Run-time System

The ASN.1 compiler software is located at /ode/Development/ASN.1 in the ODE Git repository.

- **asn1pjav_win32_6.3.0.exe:** Windows installer
- **asn1pjav_linux_6.3.0.zip**: Linux package for installing the tool on Linux and integrating into a build automation tool.
- **ossinfo**: The software license for the ASN.1 compiler and run-time is a file called ossinfo which should be copied into the installation directory. NOTE: This license can be used ***ONLY by Research*** projects managed by the "US DOT JPO ITS Connected Vehicle Data Program". To obtain a copy of the ASN.1 compiler and runtime system license file, contact Ariel Gold at FHWA [ariel.gold@dot.gov].
- **SEMI_JavaAPI.a1sproj**: The project file for the OSS compiler
- **SEMI_ASN.1_Structures_2.2.asn**: ASN.1 schema definitions for the SEMI Test Bed. This file is used by the compiler to generate POJO files. This file was downloaded from CPS-VO site from which future updates must be retrieved.
- **DSRC_R36_Source.asn**: ASN.1 schema definition for the DSRC communication. This file is referenced by **SEMI_ASN.1_Structures_2.2.asn** file. This file was downloaded from CPS-VO site from which future updates must be retrieved.
- **JavaAPI**: This is the directory to which the compiler will generate POJO files. This directory is also a java eclipse project directory for building a jar file (**semi-asn.1.jar**) that can be used by the ODE or any other application that requires ASN.1 decoding/encoding capability.

2.6.1.2 Build Environment

2.6.1.2.1 Prerequisites

- JDK 1.8 **(32 bit)** Installed
- OSS Nokalva ASN.1 compiler and valid license are installed.
- Eclipse is installed
- Maven is installed

2.6.1.2.2 Build Procedure

2.6.1.2.2.1 Automated Build

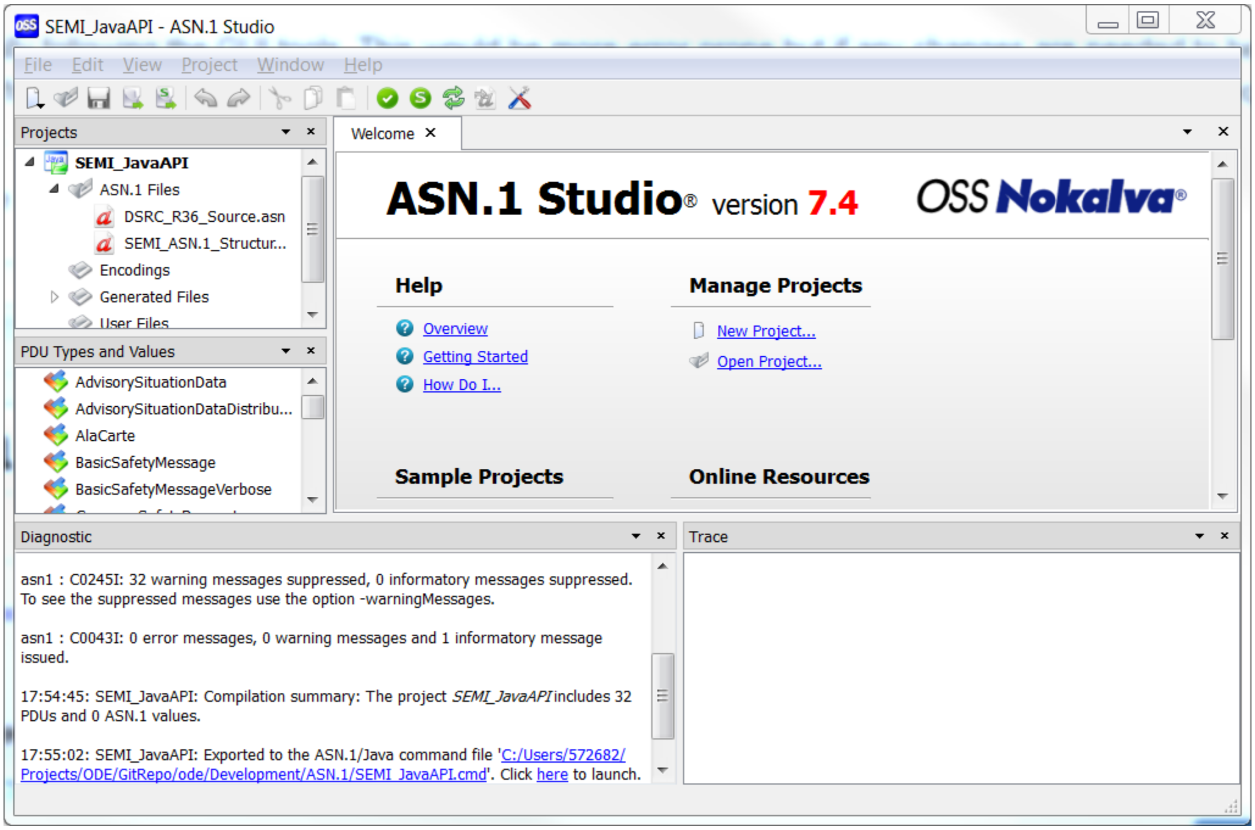
The easiest and surest way to build the Southeast Michigan Test Bed ASN.1 JavaAPI would be to run the batch file **build-semi-api.bat** located at **JavaAPI/semi** directory. This script runs the ASN.1 compiler to generate JavaAPI, installs the ASN.1 runtime on to the local Maven repository, builds the ASN.1 JavaAPI jar file and installs the jar file in the Maven local repository.

```
> build-semi-api.bat
```

2.6.1.2.2.2 Building Manually Using GUI

Alternatively, you can do all the above manually following the GUI tools. This would be more error prone but if any changes are needed to be made to the ASN.1 compiler options, it is a more user friendly approach. In that case, the chances must be carried over to the above script so subsequent builds can be done consistently and correctly.

1. Launch OSS ASN.1 Studio and load the **SEMI_JavaAPI.a1sproj** file.



```
asn1 : C0245I: 32 warning messages suppressed, 0 informatory messages suppressed.
To see the suppressed messages use the option -warningMessages.

asn1 : C0043I: 0 error messages, 0 warning messages and 1 informatory message
issued.

17:54:45: SEMI_JavaAPI: Compilation summary: The project SEMI_JavaAPI includes 32
PDUs and 0 ASN.1 values.

17:55:02: SEMI_JavaAPI: Exported to the ASN.1/Java command file 'C:/Users/572682/
Projects/ODE/GitRepo/ode/Development/ASN.1/SEMI_JavaAPI.cmd'. Click here to launch.
```

1. Click on the Green check-mark button to compile generate Java API source code. The project should already be set up to output the generated code to

"/ode/Development/ASN.1/JavaAPI/semi" directory. If not, please make sure that the source is generated or copied to that directory.

2. Run the following commands from the command line:
   > copy "C:\Program Files (x86)\OSS Nokalva\asn1pjav\win32\6.2.0\lib\osstoed.jar" lib\
   > mvn install:install-file -DcreateChecksum=true -Dpackaging=jar -Dfile=lib\osstoed.jar -DgroupId=asn.1 -DartifactId=oss.runtime -Dversion=6.2
   > mvn clean package
   > mvn install:install-file -DcreateChecksum=true -Dpackaging=jar -Dfile=target\semi-2.2.jar -DgroupId=asn.1 -DartifactId=semi -Dversion=2.2

2.6.2 ODE Deployment Artifact

The ODE deployment artifact, being a web application, is a WAR file. The WAR file will contain the ODE code as well as all ODE dependencies.

2.6.3 Building ODE WAR file locally

The ODE WAR file can be generated with Maven after the completion. The Maven build can be executed via Maven's command line interface or through Eclipse's Maven interface. The maven goals for building the WAR should be "clean install package"

Build Warning

3rd Party ODE specific dependencies such as the ASN.1 Java API, need to be installed into developers' local Maven repository prior to building the ODE Web Application. The ASN.1 dependency that the ODE application uses is not located in any remote Maven Repository. Follow the instructions on the ASN.1 Java API section to ensure that the JAR files have been installed into your local repository.

Smart Suite Team has an internal Nexus Maven repository server at http://dropbox.cnsi.com/nexus which houses the ASN.1 Jars. The ODE Bamboo CI Server Build 'plan' uses this repository when Packaging the WAR file.

When building the ODE application via the Maven command line interface or through the use of pre-made batch scripts, certain environmental variable may need to be configured in order for maven to work. The pre-made windows batch files will check to see if a file called set-env.bat exists in the current directory and call it prior to calling the maven executable. The set-env.bat script can be used to set Environmental Variables prior to the execution of the "mvn" command.

build-hdp2.3.bat

```
REM this will build the ODE-WEB application for use on HDP 2.3 as installed on the ODE.

REM if the HDP Stack version changes 2.3.0.0-2557  to 2.3.x.y-z, then pom properties or arguments will need to be updated.

REM If certain environmental variables do not exist
REM create a file called set-env.bat and set the environmental variables
there.
set "CURRENT_DIR=%cd%"
if not exist "%CURRENT_DIR%\set-env.bat" goto setEnvDone
call "%CURRENT_DIR%\set-env.bat"
:setEnvDone
call mvn clean install  package
pause
```

For example, If you have multiple versions of Java installed, you can set your JAVA_HOME variable in set-env.bat file for the version of java you wish you use when building the project.

set-env.bat

```
REM Set any environmental variables here that may
REM be required in order to build the application on your system.
REM set JAVA_HOME
REM set MAVEN_HOME
set JAVA_HOME=C:\Program Files (x86)\Java\jdk1.7.0_79\
```

Currently ODE WAR file is built using the Eclipse IDE environment. The ODE project is a Maven project however and can be built outside eclipse using Maven tool. That said, the pom file is not currently set up correctly to produce a complete WAR file, so the build cannot be created using Maven at this time but that would be something that can be readily be fixed.

To build the ODE WAR file (ode.war) on eclipse:

1. Launch eclipse and import ode webapp project if not done already
2. Do a clean build: Project > Clean... and select ode project. Then Press OK.
3. When clean build is completed, right-click the ode project on the Project Explorer and select Export > WAR file
4. Select a destination for the war file (usually under ode/Development/deploy/ode.war

2.6.4 Bamboo CI Server Build of ODE WAR File

The ODE WAR File is currently being built on each Commit by Booz Allen SmartSuite's Bamboo CI Server. The Bamboo Server will run unit tests automatically and create a deployable WAR file. This is the preferred approach for building and deploying the ODE WAR File.

The Bamboo CI Server Build also populates a version.txt (located in ode-web/src/main/webapp) which is deployed in the ODE WAR root directory. When packaged by the CI Server, version.txt looks like this:

Version.txt from a CI Server Build

```
BuildDate: 2016-01-29T15:00:13.599+0000               # Build Date and Time
BuildNumber: 73                                       # Bamboo Build Number
Commit:  76b52fc45318d8bafa566ae1f2e2df8faf5b82b4     # Git Commit Hash ID
User: bamboouser
```

Versus building a local version of the ODE WAR file:

Version.txt from a local Dev Build

```
BuildDate: 2016-02-01T21:14:41.588+0000
BuildNumber: Localbuild
Commit:  Localbuild
User: <BAH Employee ID>
```

Version.txt is updated on each mvn build. The values used in version.txt are added to the file via maven's filtering feature. Most values used in the version are set in the ODE parent pom.xml file. The Bamboo CI build sets these values via command line arguments.

Version.txt with variables

```
BuildDate: ${timestamp}
BuildNumber: ${bamboo.build.number}
Commit:  ${bamboo.git.revision}
User: ${user.name}
```

2.6.5 Deploying ODE

To deploy the ODE, there are several options. If deploying the ODE Application on an existing Yarn cluster, you will need to ensure there is adequate capacity on Yarn cluster to launch the ODE Spark Jobs. Prior to launching the ODE Application, you will need to stop an existing yarn applications, shut off tomcat, remove any existing artifacts, and restart tomcat. Upon tomcat restart, the ODE will submit new Spark Jobs to Yarn.

2.6.5.1 Deploy using Tomcat Manager Web Admin

Tomcat Manager is available at https://ip:8080/manager . Using the Manager, you can deploy, undeploy, start, stop and reload applications. To deploy an application use the "WAR file deploy" section of the manager page and select the WAR file to be deployed. Then press the Deploy button. The WAR file will be uploaded to tomcat server and loaded.

2.6.5.2 Deploy manually using FTP

Transfer the file to the Tomcat server machine via SFTP. The SSH to the server using putty and copy the WAR file from the SFTP folder that received it on to Tomcat's webapps directory. Tomcat will detect the presence of the WAR file and automatically deploy the application. To copy the war file, you will likely need to use sudo cp command in order to have elevated privileges to copy a file to the webapps folder.

2.7 Emulated Test Applications

2.7.1 Overview

Sample client application that can assist with query and real time data subscription testing. The client is written in Python 2.6 and depends on the web-socket library.

2.7.2 Capabilities

Supports the following:

1. Real Time Subscriptions
2. Query Subscriptions
3. Deposit Test Data to simulate a Real time Subscription

Subscription Record types:

1. Vehicle Data
2. Intersection Data
3. Advisory Message Data
4. Aggregate Data

2.7.3 Prerequisites

This Program was written to work with Python 2.6 and Python 2.7. It has not been tested with Python 3.

In order to run the program, you will need to install the necessary dependencies using python package management tools.

2.7.3.1 Windows

To run this program on a Windows based computer, you will need to execute the following command in at the command line to install the python web-socket library:

```
# Assumes that python is on the system Path
python -m pip install websocket-client
```

2.7.3.2 Linux

To run this program on a Linux based computer, you will need to install the Python websocket client by executing the following command:

```
sudo pip install websocket-client
```

2.7.4 Command Line arguments:

The command line can be used to use execute pre-defined queries and subscriptions for the different data types. To fully leverage the functionality of the client application, it is best to use the config file as all parameters can be overridden via the config file compared to a limited set of parameters that can be specified via command line.

Simulated Client Options

```
> simulatedclient.py -?

Usage: simulatedclient.py [options]
Situation Data App Example
Options:
  --help                  show this help message and exit

Required Parameters:
                       -t type, --type=type
                        Request Type Query - qry, Subscription = sub )
                       -d data, --data=data
                        Data Type (Vehicle, Intersection, Aggregate, Advisory)
                       -h host, --host=host Hostname
Optional Parameters:
                       -f file, --file=file
                        Full Path to a File containing JSON output that will
                        be used to validate ODE output instead of
                        default JSON files
                       -c config_file, --config=config_file
                        Full path to config file that can be used to
                        override all Settings. Config file will override all
                        command line parameters
```

2.7.5 Config File

While a subset of options can be specified via the command line, all configuration options can be specified via a configuration file. The configuration file is comprised of the following sections:

- ode
- serviceRegion
- queryParams

The configuration file will aid the ODE team with testing and validating of the ODE capabilities as different config files can be created that that can be used to test different capabilities.

2.7.5.1 ode

Describes basic information needed to connect to an ODE and to receive the data.

Value	Description	Input Type Supported
host	DNS name or IP address of the ODE application on a given host	Example: `ec2-52-6-61-205.compute-1.amazonaws.com:8080/ode`
userName	ODE UserName	User Name as string
password	ODE Password	Password as string
requestType	Type of request that you wish to place with the ODE	Valid Input Strings: • qry - Query Subscription • sub - Real-time Subscription
dataType	ODE record or data type that will be returned to the client	Valid Input Strings: • veh - Vehicle Data • int - Intersection Data • agg - Aggregate Data (Real Time Subscription only) • adv - Advisory Message Data (Query Subscriptions Only
uploadData	Boolean value used to indicate whether the test app should upload a test Input file to the ODE to simulate a real-time Data Subscription.	• true • false
inputFile	File on the local system that will be used to test real time data subscriptions by being uploaded to the	

	ODE via an Upload API that is used for Testing.	Absolute (or relative) path to file containing JSON record(s), with one JSON record per line.
validationFile	File on the local system that will be used to validate the output from the ODE.	

2.7.5.2 serviceRegion

Describes a geographic box by specifying the North West and South East Corners of that box

Value	Description	Input Type Supported
nwLat	Northwest Latitude Point	Decimal Formatted Latitude or Longitude values
nwLon	Northwest Longitude Point	
seLat	Southeast Latitude Point	
seLon	Southeast Longitude Point	

2.7.5.3 queryParams

Parameters that are only used for Query / Historical subscriptions to define the time interval for the subscription

Value	Description	Input Types Supported
endDate	End time of the Query	ISO 8601 DateTime values which includes UTC Offset Examples: • ~~yyyy-mm-ddThh:mm:ss.mmm+000~~ • ~~yyyy-mm-ddThh:mm:ss.mmm-000~~ • yyyy-mm-ddThh:mm:ss.mmmZ
startDate	Start time of the Query	
skip		Integer

limit		Integer

Sample Config File

```
[ode]
host=ec2-52-6-61-205.compute-1.amazonaws.com:8080/ode
userName=joeSmith@example.com
password=someComplexPasswod
subscriptionType=sub
dataType=veh
uploadData=true
inputFile=.//ODE_test_run_2015-07-28T12_20_22.txt
validationFile=.//ODE_test_run_2015-07-28T12_20_22.txt


[serviceRegion]
nwLat=43.652969118285434
nwLon=-85.94707489013672
seLat=36.4765153148293
seLon=-74.53468322753901

# endDate/startDate Format: "yyyy-mm-ddThh:mm:ss.mmmZ"
# Time Zone is UTC
#
[queryParams]
endDate=2015-06-18T15:52:45.500Z
startDate=2015-06-17T15:52:45.500Z
skip=0
limit=10
```

2.8 Warehouse Client

2.8.1 Overview

This application was provided by the SE MI Warehouse team as a sample client application to interface with SE MI Situation Data Clearinghouse (SDC), Situation Data Warehouse (SDW) and Situation Data Processing Center (SDPC). This application was then modified by us to a multi-threaded application allowing multiple parallel subscriptions to the SE MI test bed as well as capture multiple streams of data in parallel. The application can be found on the ODE Git repository at /ode/Development/apps/fedgov-cv-websocket-client

Please refer to the documents deposited in the Cyber-Physical Systems Virtual Organization (CPS-VO) site at http://cps-vo.org/. To gain access to the documents, you must first register an account with CPS-VO at https://cps-vo.org/user/register?destination=home. Once logged in, you will be able to browse through all the documents shared by the various groups, including the SE Michigan Test Bed team located at http://cps-vo.org/node/8937/browser. Instructions about this tool are contained in the "Readme V2.2 20150218.docx" located in the "Design > Tools > Connected Vehicle Warehouse Tools" folder.

2.8.2 Application Configuration

The application config file resides in the **config** folder. The only parameter than needs to be changed in order to start the application would be:

```
"casUserName": "Your email addressed as registered with the
Warehouse"
"casPassword": "Your Warehouse password",
```

To obtain your credentials to access the warehouse fill out the registration form at https://ec2-54-211-88-131.compute-1.amazonaws.com/casadmin/ and wait for a day or two and try accessing the site using your chosen credentials.

The easiest run the application would be from within eclipse IDE by launching gov.usdot.cv.whtools.client.WarehouseClient Java application. Once the application is running, it monitors the directory specified by the "requestDir" config parameter. If any new file is dropped into this directory OR modified, the tool will read the file and interpret it as the request data for a subscription or a query. The request is then sent to the warehouse and response to the request in the form of streaming vehicle, intersection or advisory data is received and saved into a file in the directory specified by the "responseDir" config parameter.

The tool also monitors the directory specified by the "depositFileDir" directory. Files dropped into this directory must be in valid ASN.1 format (Hex, Base64 or binary). The tool will deposit the data contained in the deposited file according to the content of the messages as well as the "depostxxx) config parameters and will deposit it to the warehouse.

Please refer to the README files located at /ode/Development/apps/fedgov-cv-websocket-client/docs for instructions on how to use the application.

2.9 Lessons Learned

See “Southeast Michigan Testbed Advanced Data Capture Field Testing Lessons Learned Repository v3.0” document.

3 Operational Environment

3.1 Virtual Private Cloud

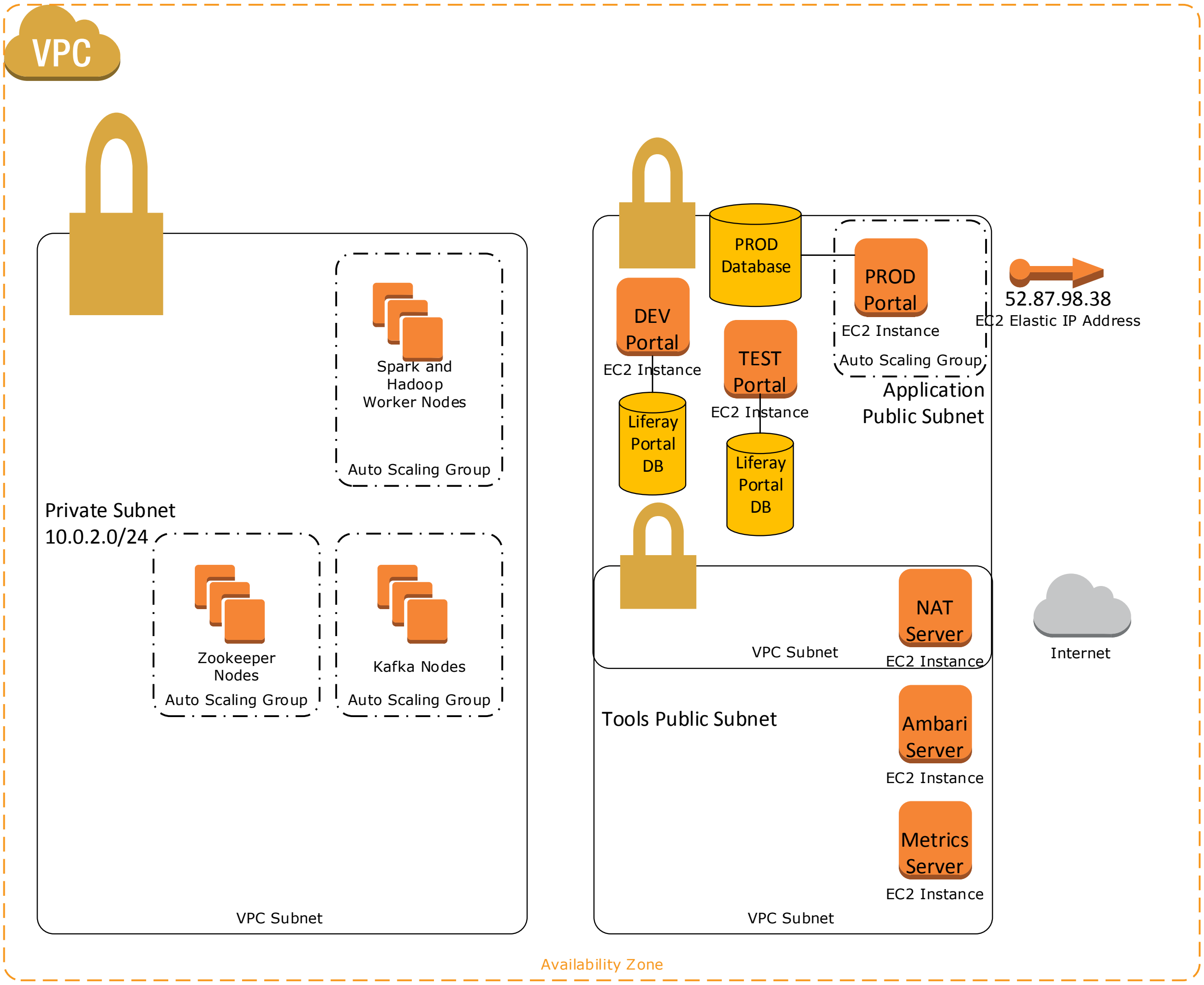


Figure 1 - SEMI ODE Virtual Private Cloud

3.2 Servers and Services

Table below shows EC2 instance names, their type, number CPU cores and memory as well as the services running on each.

EC2	Instance Type	# CPUs	Mem (GB)	SSD (GB)	Ambari	HDFS/Name Node	HDFS/Data Node	MapReduce 2	YARN/Node Manager	Yarn/ResourceManager	ZooKeeper Server	ZooKeeper Client	Ambari Metrics	Kafka Broker	Spark History	Graphite	ODE	NAT
Ambari Server	r3.large	2	15.25	16 GB + 20 GB	x					x		x	x		x			
HDFS Primary Name Node, Kafka-Zookeeper-Metrics	r3.large	2	15.25	3x16 GB		x		x			x	x	x	x				
HDFS Secondary Name Node, Kafka-Zookeeper-Metrics	r3.large	2	15.25	3x16 GB		x					x	x	x	x				
Worker Node 1: Yarn NodeManager, HDFS DataNode, Metrics	r3.xlarge	4	30.5	16 GB + 32 GB			x		x			x	x					
Kafka-Zookeeper-Metrics	r3.large	2	15.25	3x16 GB							x	x	x	x				
Worker Node 2: Yarn NodeManager, HDFS DataNode, Metrics	r3.xlarge	4	30.5	16 GB + 32 GB			x		x			x	x					
NAT Server	t2.small	1	2	1 x 8 GB														x

EC2	Instance Type	# CPUs	Mem (GB)	SSD (GB)	Ambari	HDFS/Name Node	HDFS/Data Node	MapReduce 2	YARN/Node Manager	Yarn/ResourceManager	ZooKeeper Server	ZooKeeper Client	Ambari Metrics	Kafka Broker	Spark History	Graphite	ODE	NAT
Graphite Monitoring Server	m3.medium	1	3.75	1 x 32 GB												X		
DEV Portal Server	c3.large	2	3.75	2 x 16 GB									x				x	
TEST Portal Server	c3.large	2	3.75	2 x 16 GB													x	
PROD Portal Server	C3.large	2	3.75	2 x 16 GB													X	

Table 1 - EC2 Instances Running Services

3.3 Application Server

On your local environment, you can download and install Liferay Portal 6.2 bundled with Tomcat 7. For the linux servers, insatiate a bitnami LAMP Stack with Liferay 6.2 AMI on Amazon and then install other software such as Spark and Yarn Client software.

3.4 Hadoop Cluster

The Hadoop Cluster consists of the following servers:

- Ambari Server - Serves as the Management component of Hadoop Cluster applications
- Hadoop Masters
- Hadoop Slaves

Each server is configured in a similar manner to simplify server and configuration management.

OS: Centos

Java - 1.8 (installed by Ambari)

3.4.1 Ambari Server

- Runs Ambari as distributed by Hortonworks
- Capable of provisioning Hadoop applications on Servers

3.4.2 Hadoop Masters

These Servers normally run "master" or the server related components that management and coordinate all the other Hadoop related applications. For instance, HDFS NameNode Service, or Yarn Resource manager may run on a "Master" Node.

3.4.3 Hadoop Slaves

Hadoop slaves perform the 'grunt' work of the Hadoop Cluster. Hadoop slaves can serve as Spark worker nodes or serve has HDFS Data Nodes.

3.5 Spark on Hadoop (Outside of HDP platform)

3.5.1 Installation

For each Hadoop Cluster Server, Download Spark pre-compiled packages from the Spark web site http://spark.apache.org. Un-zip the spark file into /the /opt directory.

3.5.2 Deployment Types:

Spark jobs can run in two ways: on Spark’s own standalone scheduler or on Yarn.

Spark standalone means that one server will be designated the Spark Master, and the Spark Master will handle the coordination of work among the worker spark nodes. The Spark Master will also provision resource allocation and scheduling of jobs on the Spark cluster nodes.

Spark on Yarn means that the Yarn resource manager will handle the provisioning, and scheduling of worker nodes and coordination of work among the worker nodes. In addition to managing Spark worker nodes Yarn will manage other Hadoop cluster resources as well.

4 System Configuration

4.1 Tomcat

Download and install local Tomcat. Configure tomcat using setenv.sh file in the tomcat installation's bin directory.

setenv

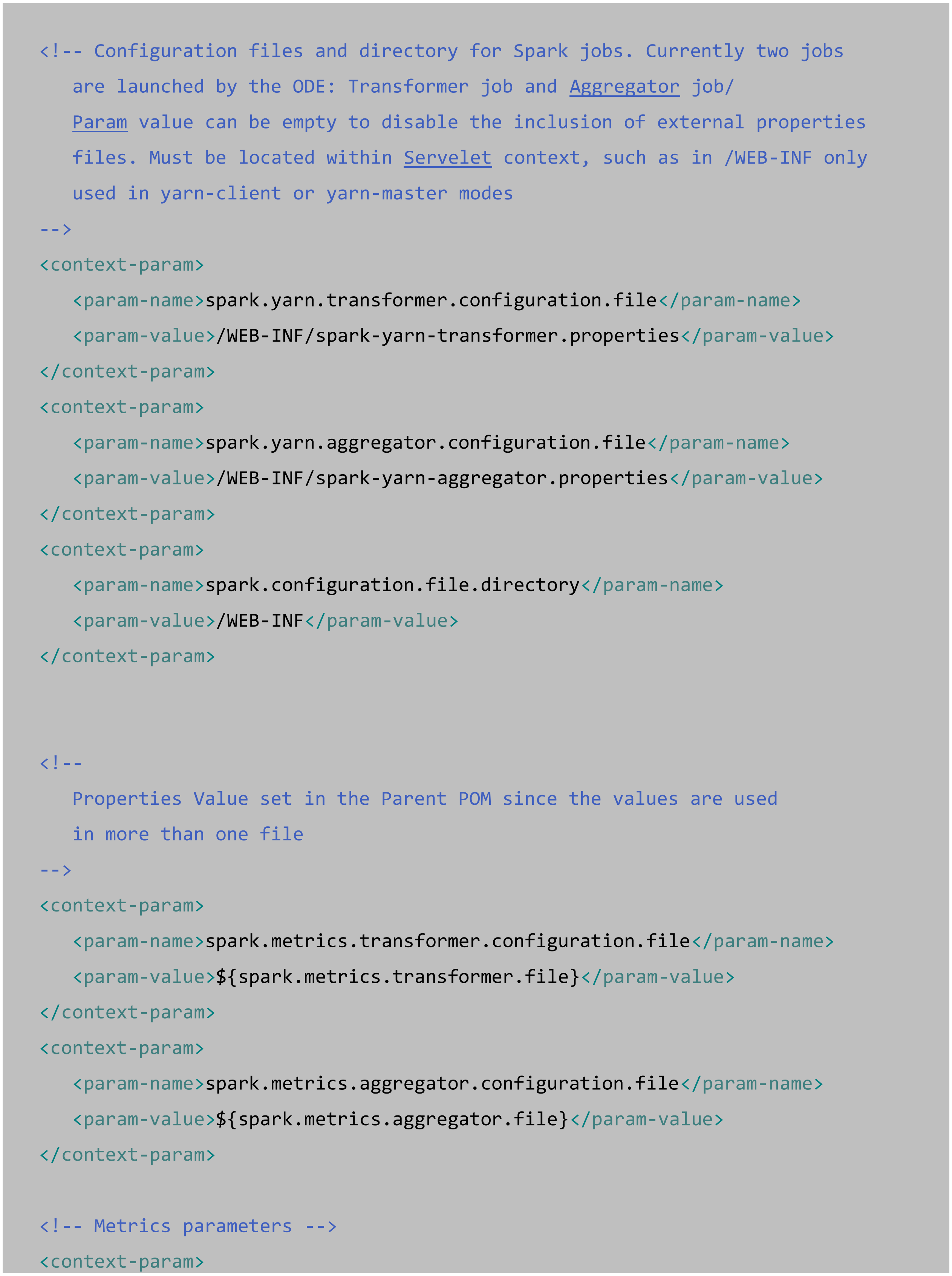
```
set JAVA_OPTS=-Dfile.encoding=UTF-8 -Xms512m -Xmx2048m
```

Also export JRE_HOME and JAVA_HOME environment variables in the same setenv file to point to the JRE and JDK home directories on the server.

4.2 Web.xml

SEMI ODE data services configuration parameters reside in the ode/WEB-INF/web.xml file shown below.

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xmlns="http://java.sun.com/xml/ns/javaee"
   xsi:schemaLocation="http://java.sun.com/xml/ns/javaee
http://java.sun.com/xml/ns/javaee/web-app_2_5.xsd"
   id="WebApp_ID" version="2.5">
   <display-name>ode</display-name>

   <context-param>
      <param-name>contextConfigLocation</param-name>
      <param-value>classpath:spring/application-config.xml</param-value>
   </context-param>

   <context-param>
      <!-- The root context for ODE (NOT USED)-->
      <param-name>web.server.root</param-name>
      <param-value>ode</param-value>
   </context-param>
   <context-param>
      <!-- The root context for Liferay portal -->
      <param-name>liferay.ws.serverhost</param-name>
      <param-value>http://localhost:8080</param-value>
   </context-param>
   <context-param>
      <!-- Company ID of the Liferay portal instance. Will vary based on Destination installation -->
      <param-name>liferay.ws.companyId</param-name>
      <param-value>10157</param-value>
   </context-param>
   <context-param>
      <!-- Liferay Portal database name -->
      <param-name>liferay.db.name</param-name>
      <param-value>lportal</param-value>
   </context-param>
   <context-param>
      <!-- Liferay Portal database host name -->
      <param-name>liferay.db.host</param-name>
      <param-value>localhost</param-value>
   </context-param>
   <context-param>
      <!-- smtp mail server host name (NOT USED)-->
      <param-name>mail.smtp.host</param-name>
      <param-value></param-value>
   </context-param>
   <context-param>
      <!-- smtp mail server port number (NOT USED)-->
      <param-name>mail.smtp.port</param-name>
      <param-value>25</param-value>
   </context-param>
   <context-param>
      <!-- smtp mail server socket factory port number (NOT USED)-->
      <param-name>mail.smtp.socketFactory.port</param-name>
      <param-value>25</param-value>
   </context-param>
   <context-param>
      <!-- FROM email address used for notifications (NOT USED)-->
      <param-name>mail.from</param-name>
      <param-value></param-value>
   </context-param>
   <context-param>
      <!-- smtp mail server username (NOT USED)-->
      <param-name>mail.user</param-name>
      <param-value></param-value>
   </context-param>
   <context-param>
      <!-- smtp mail server password (NOT USED)-->
      <param-name>mail.password</param-name>
      <param-value></param-value>
   </context-param>
   <context-param>
      <!-- USDOT CAS URL used for authenticating requests -->
      <param-name>dds.cas.url</param-name>
      <param-value>https://cas.connectedvcs.com/accounts/v1/tickets</param-value>
   </context-param>
   <context-param>
      <!-- USDOT CAS username used for authenticating requests -->
      <param-name>dds.cas.username</param-name>
      <param-value>${dds.cas.username}</param-value>
   </context-param>
   <context-param>
      <!-- USDOT CAS password used for authenticating requests -->
      <param-name>dds.cas.password</param-name>
      <param-value>${dds.cas.password}</param-value>
   </context-param>
   <context-param>
      <!-- USDOT Warehouse Tools WebSocket URI -->
      <param-name>dds.websocket.uri</param-name>
      <param-value>wss://webapp.connectedvcs.com/whtools22/websocket</param-value>
   </context-param>
   <context-param>
      <!-- Spark Master specifies if spark job is to run locally with K worker
           threads (local[k]) or on yarn (yarn-client or yarn-cluster).
           If on yarn, whether to deploy your driver on the worker nodes (yarn-cluster)
           or locally as an external client (yarn-client) (default: client)
      -->
      <param-name>spark.master</param-name>
      <param-value>yarn-client</param-value>
   </context-param>
   <context-param>
      <!-- Spark microbatch duration in milliseconds -->
      <param-name>spark.streaming.microbatch.duration.ms</param-name>
      <param-value>1000</param-value>
   </context-param>
   <context-param>
      <!-- Spark streaming sliding window duration in number of microbatches.
           multiply spark.streaming.window.microbatches by
           spark.streaming.microbatch.duration.ms to get the sliding window
           duration in milliseconds.
       -->
      <param-name>spark.streaming.window.microbatches</param-name>
      <param-value>60</param-value>
   </context-param>
   <context-param>
      <!-- Spark streaming sliding window slide duration in number of microbatches.
           multiply spark.streaming.slide.microbatches by
           spark.streaming.microbatch.duration.ms to get the slide duration
           in milliseconds.
       -->
      <param-name>spark.streaming.slide.microbatches</param-name>
      <param-value>30</param-value>
   </context-param>

   <context-param>
      <!-- Weather integration file
         CAN BE HDFS OR LOCAL
         Local example: /home/centos/semi-weather.csv</param-value
         HDFS example: hdfs://ip-10-0-16-10.ec2.internal:8020/user/hdfs/semi-weather.csv
      -->
      <param-name>spark.static.weather.file.location</param-name>
      <param-value>hdfs://ip-10-0-16-10.ec2.internal:8020/user/hdfs/semi-weather.csv</param-value>
   </context-param>
   <context-param>
      <!-- Sanitization file
         CAN BE HDFS OR LOCAL
         Local example: /home/centos/sanitization.json</param-value
         HDFS example: hdfs://ip-10-0-16-10.ec2.internal:8020/user/hdfs/sanitization.json
      -->
      <param-name>spark.static.sanitization.file.location</param-name>
      <param-value>hdfs://ip-10-0-16-10.ec2.internal:8020/user/hdfs/sanitization.json</param-value>
   </context-param>
   <context-param>
      <!-- Validation file
         CAN BE HDFS OR LOCAL
         Local example: /home/centos/validation.json</param-value
         HDFS example: hdfs://ip-10-0-16-10.ec2.internal:8020/user/hdfs/validation.json
      -->
      <param-name>spark.static.validation.file.location</param-name>
      <param-value>hdfs://ip-10-0-16-10.ec2.internal:8020/user/hdfs/validation.json</param-value>
   </context-param>
   <context-param>
      <param-name>spark.road.segment.snapping.tolerance</param-name>
      <param-value>20</param-value>
   </context-param>



   <context-param>
      <!-- Spark Assembly to be uploaded to Yarn Cluster Assumed to be installed
         on the Web application Server This param specifies the name based on HDP
         naming convention
      -->
      <param-name>spark.assembly.jar</param-name>
      <param-value>spark-assembly-1.3.1.2.3.0.0-2557-hadoop2.7.1.2.3.0.0-2557.jar</param-value>
   </context-param>


   <context-param>
      <!-- List of Kafka broker server host names ad port numbers-->
      <param-name>kafka.metadata.broker.list</param-name>
      <param-value>ip-10-0-16-115.ec2.internal:6667,ip-10-0-16-143.ec2.internal:6667,ip-10-0-16-10.ec2.internal:6667</param-value>
   </context-param>
   <context-param>
      <param-name>data.sequence.reorder.delay</param-name>
      <param-value>1000</param-value>
   </context-param>
   <context-param>
      <param-name>data.sequence.reorder.period</param-name>
      <param-value>1000</param-value>
   </context-param>
   <context-param>
      <!-- List of Zookeeper server host names and port numbers-->
      <param-name>zk.connection.strings</param-name>
      <param-value>ip-10-0-16-115.ec2.internal:2181,ip-10-0-16-10.ec2.internal:2181,ip-10-0-16-143.ec2.internal:2181</param-value>
   </context-param>
   <context-param>
      <!-- jar file name containing Spark ODE applications -->
      <param-name>ode.spark.jar</param-name>
      <param-value>ode-spark-0.0.1-SNAPSHOT.jar</param-value>
   </context-param>
   <context-param>
      <!-- RSA Keystore file used in ODE authentication -->
      <param-name>token.key.rsa.pem</param-name>
      <param-value>mykey.pem</param-value>
   </context-param>
```



```
<!-- Configuration files and directory for Spark jobs. Currently two jobs
   are launched by the ODE: Transformer job and Aggregator job/
   Param value can be empty to disable the inclusion of external properties
   files. Must be located within Servelet context, such as in /WEB-INF only
   used in yarn-client or yarn-master modes
-->
<context-param>
   <param-name>spark.yarn.transformer.configuration.file</param-name>
   <param-value>/WEB-INF/spark-yarn-transformer.properties</param-value>
</context-param>
<context-param>
   <param-name>spark.yarn.aggregator.configuration.file</param-name>
   <param-value>/WEB-INF/spark-yarn-aggregator.properties</param-value>
</context-param>
<context-param>
   <param-name>spark.configuration.file.directory</param-name>
   <param-value>/WEB-INF</param-value>
</context-param>



<!--
   Properties Value set in the Parent POM since the values are used
   in more than one file
-->
<context-param>
   <param-name>spark.metrics.transformer.configuration.file</param-name>
   <param-value>${spark.metrics.transformer.file}</param-value>
</context-param>
<context-param>
   <param-name>spark.metrics.aggregator.configuration.file</param-name>
   <param-value>${spark.metrics.aggregator.file}</param-value>
</context-param>

<!-- Metrics parameters -->
<context-param>
```

```
        <param-name>metrics.prefix</param-name>
        <param-value>ode-</param-value>
    </context-param>
    <context-param>
        <param-name>metrics.polling.rate.seconds</param-name>
        <param-value>10</param-value>
    </context-param>
    <context-param>
        <param-name>metrics.graphite.host</param-name>
        <param-value>ip-10-0-8-41.ec2.internal</param-value>
    </context-param>
    <context-param>
        <param-name>metrics.graphite.port</param-name>
        <param-value>2003</param-value>
    </context-param>

    <!-- *** PERFORMANCE PARAMS *** -->
    <context-param>
        <param-name>dds.num.vsr.in.bundle.to.use</param-name>
        <param-value>10</param-value>
    </context-param>
    <context-param>
        <param-name>spark.yarn.transformer.driver.cores</param-name>
        <param-value>1</param-value>
    </context-param>
    <context-param>
        <param-name>spark.yarn.transformer.driver.memory</param-name>
        <param-value>512m</param-value>
    </context-param>
    <context-param>
        <param-name>spark.yarn.transformer.executor.cores</param-name>
        <param-value>2</param-value>
    </context-param>
    <context-param>
        <param-name>spark.yarn.transformer.executor.memory</param-name>
        <param-value>9216m</param-value>
```

```
   </context-param>

   <context-param>
      <param-name>spark.yarn.aggregator.driver.cores</param-name>
      <param-value>1</param-value>
   </context-param>
   <context-param>
      <param-name>spark.yarn.aggregator.driver.memory</param-name>
      <param-value>512m</param-value>
   </context-param>
   <context-param>
      <param-name>spark.yarn.aggregator.executor.cores</param-name>
      <param-value>1</param-value>
   </context-param>
   <context-param>
      <param-name>spark.yarn.aggregator.executor.memory</param-name>
      <param-value>2048m</param-value>
   </context-param>


   <!-- This param can be used to disable ODE Aggregator. Default = true -->
   <context-param>
      <param-name>spark.ode.aggregator.enabled</param-name>
      <param-value>true</param-value>
   </context-param>
   <!--
      This param can be used to run the aggregator in a sliding windnow
      of size: spark.streaming.window.microbatches * 
spark.streaming.microbatch.duration.ms
      OR
      run it in a microbatch size of:
      spark.streaming.window.microbatches * spark.streaming.microbatch.duration.ms

      Default: true
   -->
   <context-param>
```

```
        <param-name>spark.run.aggregator.in.sliding.window</param-name>
        <param-value>false</param-value>
    </context-param>
    <!--
        This param can be used to run the aggregator in the same Spark
        application as the Vehicle Data Transformer application.
        Default: false
    -->
    <context-param>
        <param-name>spark.run.ode.aggregator.in.vdp</param-name>
        <param-value>false</param-value>
    </context-param>
    <!-- Number of kafka consumer threads -->
    <context-param>
        <param-name>kafka.consumer.threads</param-name>
        <param-value>9</param-value>
    </context-param>
    <!--
        This param can be used to change the spark scheduler mode of operation.
        Valid values are FIFO and FAIR. Default = FIFO
    -->
    <context-param>
        <param-name>spark.scheduler.mode</param-name>
        <param-value>FAIR</param-value>
    </context-param>
    <!--
        This param can be used to run the spark apps with a single kafka receiver
        supporting multiple topic partitions or route each kafka partition to
        a separate Spark receiver. Default = false

        When false, one receiver is used to receive all partitions. When true,
        multiple receivers are dispatched to worker nodes to receive data in
        parallel streams. the streams are then unified by spark before processing.
        This param should probably be set to true if a single receiver
        is the cause of data bottleneck.
    -->
```

```
   <context-param>
      <param-name>spark.use.parallel.receivers</param-name>
      <param-value>false</param-value>
   </context-param>

   <listener>
      <listener-
class>org.springframework.web.context.ContextLoaderListener</listener-class>
   </listener>
   <listener>
      <listener-class>com.bah.ode.server.InitServeletListener</listener-class>
   </listener>

   <servlet>
      <servlet-name>dispatcherServlet</servlet-name>
      <servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-
class>
      <init-param>
         <param-name>contextConfigLocation</param-name>
         <param-value>/WEB-INF/mvc-config.xml</param-value>
      </init-param>
      <load-on-startup>1</load-on-startup>
   </servlet>
   <servlet-mapping>
      <servlet-name>dispatcherServlet</servlet-name>
      <url-pattern>/web/*</url-pattern>
   </servlet-mapping>
   <!-- <servlet> <description>InitServlet is intended to perform eager
initializations
      for ODE application.</description> <display-name>InitServlet</display-name>
      <servlet-name>InitServlet</servlet-name> <servlet-
class>com.bah.ode.server.InitServlet</servlet-class>
      <load-on-startup>0</load-on-startup> </servlet> <servlet-mapping> <servlet-
name>InitServlet</servlet-name>
      <url-pattern>/api/sample/javax/*</url-pattern> </servlet-mapping> -->
   <servlet>
      <servlet-name>RESTfulServices</servlet-name>
      <servlet-class>org.glassfish.jersey.servlet.ServletContainer</servlet-class>
      <init-param>
         <param-name>javax.ws.rs.Application</param-name>
         <param-value>com.bah.ode.api.OdeResourceConfig</param-value>
      </init-param>

      <init-param>
         <param-name>jersey.config.server.provider.packages</param-name>
         <param-value>com.bah.ode.api</param-value>
      </init-param>
      <init-param>
         <!-- Default value is true for Jersery -->
         <param-name>jersey.config.server.provider.scanning.recursive</param-name>
         <param-value>true</param-value>
      </init-param>
      <load-on-startup>1</load-on-startup>
   </servlet>

   <servlet-mapping>
      <servlet-name>RESTfulServices</servlet-name>
      <url-pattern>/api/*</url-pattern>
   </servlet-mapping>

   <servlet>
      <servlet-name>ViewStatusMessages</servlet-name>
      <servlet-class>ch.qos.logback.classic.ViewStatusMessagesServlet</servlet-class>
   </servlet>

   <servlet-mapping>
      <servlet-name>ViewStatusMessages</servlet-name>
      <url-pattern>/lbClassicStatus</url-pattern>
   </servlet-mapping>
</web-app>
```

4.3 spark-transformer-metrics.properties

```
# Enable Graphite
*.sink.graphite.class=org.apache.spark.metrics.sink.GraphiteSink
*.sink.graphite.host=${spark.metrics.server.host}
*.sink.graphite.port=${spark.metrics.server.port}
*.sink.graphite.period=10
*.sink.graphite.prefix=aggregator

# Enable jvm source for instance master, worker, driver and executor
master.source.jvm.class=org.apache.spark.metrics.source.JvmSource
worker.source.jvm.class=org.apache.spark.metrics.source.JvmSource
driver.source.jvm.class=org.apache.spark.metrics.source.JvmSource
executor.source.jvm.class=org.apache.spark.metrics.source.JvmSource
```

4.4 spark-yarn-transformer.properties

```
# Enable Graphite
*.sink.graphite.class=org.apache.spark.metrics.sink.GraphiteSink
*.sink.graphite.host=${spark.metrics.server.host}
*.sink.graphite.port=${spark.metrics.server.port}
*.sink.graphite.period=10
*.sink.graphite.prefix=transformer

# Enable jvm source for instance master, worker, driver and executor
master.source.jvm.class=org.apache.spark.metrics.source.JvmSource
worker.source.jvm.class=org.apache.spark.metrics.source.JvmSource
driver.source.jvm.class=org.apache.spark.metrics.source.JvmSource
executor.source.jvm.class=org.apache.spark.metrics.source.JvmSource
```

4.5 spark-yarn-aggregator.properties

```
# Properties to Tune Spark when in Yarn Mode
# Work in progress.
#

#spark.yarn.containerLauncherMaxThreads  25
spark.shuffle.service.enabled true
spark.dynamicAllocation.enabled true
spark.dynamicAllocation.minExecutors 2
spark.dynamicAllocation.initialExecutors 2
spark.dynamicAllocation.maxExecutors 5
spark.dynamicAllocation.executorIdleTimeout 120

spark.metrics.conf ${spark.metrics.aggregator.file}

# In milliseconds
spark.locality.wait 2000
spark.streaming.blockInterval 500

#logging parameters
spark.executor.logs.rolling.strategy ${spark.executor.logs.rolling.strategy}
spark.executor.logs.rolling.maxSize ${spark.executor.logs.rolling.maxSize}
spark.executor.logs.rolling.maxRetainedFiles
${spark.executor.logs.rolling.maxRetainedFiles}
```

4.6 spark-aggregator-metrics.properties

```
# Properties to Tune Spark when in Yarn Mode
# Work in progress.
#
#spark.yarn.containerLauncherMaxThreads  25

spark.shuffle.service.enabled true
spark.dynamicAllocation.enabled true
spark.dynamicAllocation.minExecutors 2
spark.dynamicAllocation.initialExecutors 2
spark.dynamicAllocation.maxExecutors 4
spark.dynamicAllocation.executorIdleTimeout 120

spark.metrics.conf ${spark.metrics.transformer.file}

# In milliseconds
spark.locality.wait 2000
spark.streaming.blockInterval 500
```

```
#logging parameters
spark.executor.logs.rolling.strategy ${spark.executor.logs.rolling.strategy}
spark.executor.logs.rolling.maxSize ${spark.executor.logs.rolling.maxSize}
spark.executor.logs.rolling.maxRetainedFiles
${spark.executor.logs.rolling.maxRetainedFiles}
```

5 Interfaces

The ODE provides two types of API's to its subscribers, 1) a RESTful API for administrative functions and 2) a streaming API for data services. Both APIs communicate through HTTPS.

ODE provides support for WebSocket as defined by RFC 6455. The ODE will interact with the client application using the standard WebSocket interface (RFC-6455 http://tools.ietf.org/html/rfc6455). DDS and other data sources through the available network interfaces provided by these sources. The following section highlights the key components and features that will support the ODE's interaction with the DDS.

Note: Initial access to these interfaces is currently restricted to the ODE development team, but once sufficient testing is complete, any individual who successfully registers at the ODE Portal web site will be able to leverage the ODE interfaces. The word individual is intended to identify a person who will develop an application leveraging ODE data stream(s).

5.1 ODE RESTful API

The ODE RESTful API employs *synchronous* calls to ODE services for use by clients and is used primarily for sending administrative commands to the ODE. Standard HTTP verbs such as GET, POST, PUT, DELETE, etc., may be deployed for various relevant functions.

```
host: ec2-52-20-100-211.compute-1.amazonaws.com

basePath: /ode

schemes:

  - http

  - https
```

5.1.1 Login

<table>
<tr><td><b>Path</b></td><td>/auth/login</td></tr>
<tr><td><b>Verb</b></td><td>GET</td></tr>
<tr><td><b>Summary</b></td><td>Authenticates the user given username/password and returns a security token</td></tr>
<tr><td><b>Produces</b></td><td>- application/json</td></tr>
<tr><td><b>Headers</b></td><td>
<table>
<tr><th>Name</th><th>Value</th><th>Description</th><th>Required</th></tr>
<tr><td>Authorization</td><td>&lt;username>:<password&gt;</td><td>The user name/password combination in the standard Basic HTTP Authentication format 'username:password'</td><td>True</td></tr>
</table>
</td></tr>
</table>

Responses	Code	Description	Schema
	200 - OK	successful operation	Token
	401 - File Not Found	Invalid username/password supplied	**See return response below.**

Table 2 – Login API

If login fails due to invalid username/password the following response will be returned to the client:

```
Status: error

Error Thrown: Unauthorized

Details: {"readyState":4,

"responseText":"{\"code\":\"FAILURE\",\"message\":\"Unable to Authenticate.
Error java.lang.NumberFormatException: For input string:
\\\"{\\\"exception\\\":\\\"Authenticated access
required\\\"}\\\"\",\"dataType\":\"status\",\"version\":1}","responseJSON":{"co
de":"FAILURE","message":"Unable to Authenticate. Error
java.lang.NumberFormatException: For input string:
\"{\"exception\":\"Authenticated access
required\"}\"","dataType":"status","version":1},"status":401,"statusText":"Unau
thorized"}
```

5.1.2 Logout

<table>
<tr><td><b>Path</b></td><td>/auth/logout</td></tr>
<tr><td><b>Verb</b></td><td>POST</td></tr>
<tr><td><b>Summary</b></td><td>Revokes a security token</td></tr>
<tr><td><b>Produces</b></td><td>- text/plain</td></tr>
<tr><td><b>Headers</b></td><td><table>
<tr><th>Name</th><th>Value</th><th>Description</th><th>Required</th></tr>
<tr><td>Authorization</td><td>Bearer <token&gt;</td><td>The authorization token to be revoked</td><td>True</td></tr>
</table></td></tr>
<tr><td><b>Parameters</b></td><td><table>
<tr><th>Name</th><th>Value</th><th>Description</th><th>Required</th><th>In</th></tr>
<tr><td></td><td></td><td></td><td></td><td></td></tr>
</table></td></tr>
<tr><td><b>Responses</b></td><td><table>
<tr><th>Code</th><th>Description</th><th>Schema</th></tr>
<tr><td><b>200 - OK</b></td><td>Successful Logout</td><td></td></tr>
<tr><td><b>401 - Unauthorized</b></td><td>Invalid token</td><td></td></tr>
</table></td></tr>
</table>

Table 3 - Logout API

5.2 ODE Streaming API

The Streaming API utilizes the standard Web Socket interface as defined by RFC 6455 (http://tools.ietf.org/html/rfc6455). The ODE Client applications will be able to submit one or more subscription and/or query requests to the ODE via this interface.

The following parameters are required for establishing a WebSocket client connection.

```
host: <Host server IP address or DNS name>

basePath: /ode/api/ws
```

```
schemes:
  - ws
  - wss
```

The ODE Streaming API employs *asynchronous* calls to ODE services which are forwarded to the SE-MI clearinghouse and other real-time data sources. The streamed data is received indefinitely by the client application until the application deliberately disconnects from the ODE or connection between ODE and the application endpoint is disrupted abnormally.

The data from either the SDW or the SDPC are historical by definition and not "real-time". SDW retains data for a depositor selected period of time as specified by the timeToLive parameter of the data deposit message (an enumerated value indicating 1 minute, 30 minutes, 1 day, 1 week, 1 month, or 1 year). This optional parameter is available only for ISD and TSD messages. If not specified and for EVSD messages, the data will be retained in the SDW for a default duration of 30 minutes. SDPC on the other hand retains data for a much longer period of time[1]. Applications requiring historical data must specify the source of data in the data query request: SDW or SDPS. Since queries submitted to the ODE are historical by nature, a start and end date and time can be provided to limit the scope of the search, otherwise all data available in the SDW or SDPC database will be returned. Nevertheless, the time series data returned by SDW and SDPC are finite when compared to clearinghouse subscriptions, which will stream data indefinitely to the client application (or until the underlying WebSocket connection linking the ODE and endpoint is broken for whatever reason).

5.2.1 ODE Control Messages

The response to a Subscription request may contain control messages such as CONNECT, START, etc, as well as data messages containing the requested data. The response to Query requests may contain control messages such as CONNECT, START, STOP. A query request may receive two STOP messages. One will contain the number of "bundles" and the number of records received from the data source. A bundle may contain 1 or more records.

[1] *As of this writing, the SDPC is configured to retain data for 6-months*

```
{"metadata":{"payloadType":"control","version":1},"payload":{"dataSourceBundleCount":20,"receivedRecordCount":203,"tag":"STOP","dataType":"Control","version":1},"version":1}
```

The other contains the actual number of records sent to the client.

```
{"metadata":{"payloadType":"control","version":1},"payload":{"dataSourceBundleCount":20,"receivedRecordCount":203,"sentRecordCount":10,"tag":"STOP","dataType":"Control","version":1},"version":1}
```

For detailed schema of ODE Control messages see section 5.3.2.3.3

5.2.2 Vehicle Data Subscription Request

<table>
<tr><td><b>Path</b></td><td>/sub/veh</td></tr>
<tr><td><b>Summary</b></td><td>Subscription request for vehicle data</td></tr>
<tr><td><b>Consumes</b></td><td>- application/json</td></tr>
<tr><td><b>Produces</b></td><td>- application/json</td></tr>
<tr><td><b>Parameters</b></td><td><table>
<tr><th>Name</th><th>Value</th><th>Description</th><th>Required</th><th>In</th></tr>
<tr><td>request</td><td>Subscription Request (see section 5.3.1.1)</td><td>The subscription request parameters</td><td>true</td><td>body</td></tr>
<tr><td>token</td><td>string</td><td>A valid security token obtained from login service</td><td>true</td><td>query-param</td></tr>
</table></td></tr>
<tr><td><b>Responses</b></td><td><table>
<tr><th>Code</th><th>Description</th><th>Schema</th></tr>
</table></td></tr>
</table>

	0	SUCCESS	OdeVehicleDataFlat
	1	FAILURE	OdeStatus
	2	SOURCE_CONNECTION_ERROR	OdeStatus
	3	INVALID_REQUEST_TYPE_ERROR	OdeStatus
	4	INVALID_DATA_TYPE_ERROR	OdeStatus

Table 4 - Vehicle Data Subscription Request

5.2.3 Vehicle Data Query Request

Path	/qry/veh				
Summary	Query request for vehicle data				
Consumes	- application/json				
Produces	- application/json				
Parameters	**Name**	**Value**	**Description**	**Required**	**In**
	request	Query Request (see Section 5.3.1.2)	The subscription request parameters	true	body
	token	string	A valid security token obtained from login service	true	Query-param
Responses	**Code**	**Description**	**Schema**		

<table>
<tr><td></td><td>0</td><td>SUCCESS</td><td>OdeVehicleDataFlat</td></tr>
<tr><td></td><td>1</td><td>FAILURE</td><td>OdeStatus</td></tr>
<tr><td></td><td>2</td><td>SOURCE_CONNECTION_ERROR</td><td>OdeStatus</td></tr>
<tr><td></td><td>3</td><td>INVALID_REQUEST_TYPE_ERROR</td><td>OdeStatus</td></tr>
<tr><td></td><td>4</td><td>INVALID_DATA_TYPE_ERROR</td><td>OdeStatus</td></tr>
</table>

Table 5 - Vehicle Data Query Request

5.2.4 Intersection, Map and SPaT Data Subscription Request

<table>
<tr><td><b>Path</b></td><td colspan="5">/sub/int OR /sub/map OR /sub/spat</td></tr>
<tr><td><b>Summary</b></td><td colspan="5">Subscription request for intersection data</td></tr>
<tr><td><b>Consumes</b></td><td colspan="5">- application/json</td></tr>
<tr><td><b>Produces</b></td><td colspan="5">- application/json</td></tr>
<tr><td rowspan="3"><b>Parameters</b></td><td><b>Name</b></td><td><b>Value</b></td><td><b>Description</b></td><td><b>Required</b></td><td><b>In</b></td></tr>
<tr><td>request</td><td>Subscription Request (see section 5.3.1.1)</td><td>The subscription request parameters</td><td>true</td><td>body</td></tr>
<tr><td>token</td><td>string</td><td>A valid security token obtained from login service</td><td>true</td><td>query-param</td></tr>
<tr><td><b>Responses</b></td><td><b>Code</b></td><td colspan="2"><b>Description</b></td><td colspan="2"><b>Schema</b></td></tr>
</table>

0	SUCCESS	OdeIntersectionData
1	FAILURE	OdeStatus
2	SOURCE_CONNECTION_ERROR	OdeStatus
3	INVALID_REQUEST_TYPE_ERROR	OdeStatus
4	INVALID_DATA_TYPE_ERROR	OdeStatus

Table 6 - Intersection Data Subscription Request

5.2.5 Intersection Data Query Request

<table>
<tr><td><b>Path</b></td><td>/sub/int, /sub/map, /sub/spat</td></tr>
<tr><td><b>Summary</b></td><td>Query request for intersection data</td></tr>
<tr><td><b>Consumes</b></td><td>- application/json</td></tr>
<tr><td><b>Produces</b></td><td>- application/json</td></tr>
<tr><td><b>Parameters</b></td><td><table>
<tr><th>Name</th><th>Value</th><th>Description</th><th>Required</th><th>In</th></tr>
<tr><td>request</td><td>Query Request (see Section 5.3.1.2)</td><td>The subscription request parameters</td><td>true</td><td>body</td></tr>
<tr><td>token</td><td>string</td><td>A valid security token obtained from login service</td><td>true</td><td>query-param</td></tr>
</table></td></tr>
<tr><td><b>Responses</b></td><td><table>
<tr><th>Code</th><th>Description</th><th>Schema</th></tr>
</table></td></tr>
</table>

	0	SUCCESS	OdeIntersectionData
	1	FAILURE	OdeStatus
	2	SOURCE_CONNECTION_ERROR	OdeStatus
	3	INVALID_REQUEST_TYPE_ERROR	OdeStatus
	4	INVALID_DATA_TYPE_ERROR	OdeStatus

Table 7 - Intersection Data Query Request

5.2.6 Aggregate Vehicle Data Subscription Request

Path	/sub/agg				
Summary	Subscription request for aggregate vehicle data				
Consumes	- application/json				
Produces	- application/json				
Parameters	**Name**	**Value**	**Description**	**Required**	**In**
	request	Subscription Request (see section 5.3.1.1)	The subscription request parameters	true	body
	token	string	A valid security token obtained from login service	true	query-param
Responses	**Code**	**Description**	**Schema**		

	0	SUCCESS	OdeAggregateData
	1	FAILURE	OdeStatus
	2	SOURCE_CONNECTION_ERROR	OdeStatus
	3	INVALID_REQUEST_TYPE_ERROR	OdeStatus
	4	INVALID_DATA_TYPE_ERROR	OdeStatus

Table 8 - Aggregate Data Subscription Request

5.2.7 Aggregate Vehicle Data Query Request

Path	/qry/veh				
Summary	Query request for aggregate vehicle data				
Consumes	- application/json				
Produces	- application/json				
Parameters	**Name**	**Value**	**Description**	**Required**	**In**
	request	Query Request (see Section 5.3.1.2)	The subscription request parameters	true	body
	token	string	A valid security token obtained from login service	true	Query-param
Responses	**Code**	**Description**	**Schema**		

	0	SUCCESS	OdeAggregateData
	1	FAILURE	OdeStatus
	2	SOURCE_CONNECTION_ERROR	OdeStatus
	3	INVALID_REQUEST_TYPE_ERROR	OdeStatus
	4	INVALID_DATA_TYPE_ERROR	OdeStatus

Table 9 - Aggregate Data Query Request

5.3 Schemas

The ODE API consists of data “request” and “responses”. Section 5.3.1 specifies the structure of the data contained in the body of a “request”. Section 5.3.2 specifies the structure of the data contained in the body of the “response” records.

5.3.1 ODE Request Schemas

5.3.1.1 Subscription Data Request

Name	**Required/Optional (R/O)**	**Type**	**Description**	**Units**	**Valid Min**	**Valid Max**	**Default**
dataSource	O	String	SDC (Optional)	Enumeration	SDC	SDC	SDC
nwLat	R	decimal	Latitude of the northwest corner of the request service region.	Degrees	-90.0000000	+90.0000001	none
nwLon	R	decimal	Longitude of the northwest corner of the request service region.	Degrees	-180.0000000	+180.0000001	none

`polyline`	O	`Array`	Array of OdeRoadSegment. See 5.3.1.3		N/A	N/A	none
`seLat`	R	`decimal`	Latitude of the southeast corner of the request service region.	Degrees	-90.0000000	+90.0000000	none
`seLon`	R	`decimal`	Longitude of the southeast corner of the request service region.	Degrees	-180.0000000	+180.0000001	none

Table 10 - OdeSubscriptionRequest

5.3.1.2 Query Data Request and Test Data Request

Name	**Required/Optional (R/O)**	**Type**	**Description**	**Units**	**Valid Min**	**Valid Max**	**Default**
`dataSource`		`String`	SDW or SDPC (optional)	enum	N/A	N/A	SDPC
`endTime`	O	`string`	End time of the query. If not provided, the most recent record available will be returned.	ISO 8601 Date-Time			
`limit`	R	`integer`	Maximum number of records to send in the response.	count	0	2147483647	2147483647
`nwLat`	R	`decimal`		Degrees	-90.0000000	+90.0000000	

nwLon	R	decimal		Degrees	-180.0000000	+180.0000000	
polyline	O	Array	Array of OdeRoadSegment		N/A	N/A	none
seLat	R	decimal		Degrees	-90.0000000	+90.0000000	
seLon	R	decimal		Degrees	-180.0000000	+180.0000000	
skip	R	integer	Maximum number of records to skip	count	0	2147483647	0
startTime	O	String	Start time of the query. If not provided, the earliest record available will be returned.	ISO 8601 Date-Time			

Table 11 - OdeQueryRequest

5.3.1.3 OdeRoadSegment

Name	Required/Optional (R/O)	Type	Description	Units	Valid Min	Valid Max	Default
endPoint	R	OdePoint	Road segment end point. See 5.3.1.4				
id	R	string	Road segment Identifier				
prevSegment	O if startPoint specified	string	ID of the previous road segment connected to this road segment. If this field is specified, it is assumed the endPoint of the previous road				

			segment is used as the startPoint of this road segment, hence specifying the start point will not be required.				
startPoint	O if prevSegment specified, otherwise R	OdePoint	Road segment start point				

Table 12 - OdeRoadSegment

5.3.1.4 OdePoint

Name	**Required/Optional (R/O)**	**Type**	**Description**	**Units**	**Valid Min**	**Valid Max**	**Default**
latitude	R	decimal	Point latitude	Degrees	-180.0000000	+180.0000000	
longitude	R	decimal	Point longitude	Degrees	-90.0000000	+90.0000000	

Table 13 - OdePoint

5.3.2 ODE Data Schemas

5.3.2.1 ODE Data Message

The top level structure of ODE Data Message is defined in the table below:

Name	**Type**	**Description**	**Units**	**Valid Min**	**Valid Max**
metadata	OdeMsgMetadata	See 5.3.2.2			

payload	One of OdeMsgPayload types.	See section 5.3.2.3 and its subsections			
version	Integer	The version number for this schema. If new data elements are added or removed from the schema, the version number will increment.			

Table 14 - OdeDataMessage

5.3.2.2 ODE Data Message Metadata

Name	Type	Description	Units	Valid Min	Valid Max
payloadType	string	One of: "status", "control", "auth", "veh", "int", "adv", "map", "spat", "agg", "other"	enum		
latency	Integer	Amount of time it took to process this record from the time it was received (based on receivedAt payload field)	Milliseconds	0	9223372036854775807

violations	Array of violation objects	This is an array of OdePayloadViolation objects. See section 5.3.2.2.1 for details.			
version	Integer	The version number for this schema. If new data elements are added or removed from the schema, the version number will increment.			

Table 15 – OdeMsgMetadata

5.3.2.2.1 ODE Payload Violation

Name	Type	Description	Units	Valid Min	Valid Max
fieldName	string	The name of the payload field for which violation was detected.	N/A		
validMax	Decimal	Maximum value considered to be valid.	The Unit of measure of the fieldName		
validMin	Decimal	Minimum value considered to be valid	The Unit of measure of the fieldName		

5.3.2.3 ODE Data Message Payload

OdeMsgPayload is the parent class for all payload data structures, each of which contain the following data elements, datatype and version. For

Name	Type	Description	Units	Valid Min	Valid Max
dataType	string	One of: "status", "control", "auth", "veh", "int", "adv", "map", "spat", "agg", "other"	Enum		
version	Integer	The version number for this schema. If new data elements are added or removed from the schema, the version number will increment.			

Table 16 - OdeMsgPayload

5.3.2.3.1 ODE Authorization Response Message

Name	Type	Description	Units	Valid Min	Valid Max
dataType	string	Authorization response type	Enum	"auth"	"auth"
version	Integer	See 5.3.2.3		1	1

Table 17 - OdeAuthentication

5.3.2.3.2 ODE Status Message

Name	Type	Description	Units	Valid Min	Valid Max
status	String	Request status	Enum	"status"	"status"
version	Integer	See 5.3.2.3			

Table 18 - OdeStatus

5.3.2.3.3 ODE Control Message

Name	Type	Description	Units	Valid Min	Valid Max
dataSourceBundleCount	Integer	Number of bundles reported by the data source	Count	0	9223372036854775807
receivedRecordCount	Integer	Number of messages received by the ODE	Count	0	9223372036854775807
sentRecordCount	Integer	Number of records sent by the ODE to the client	Count	0	9223372036854775807
tag	Enum	The control message event tag	Enum	One of CONNECTED, START, STOP, ERROR	One of CONNECTED, START, STOP, ERROR
version	Integer	See 5.3.2.3			

Table 19 - OdeControlData

5.3.2.3.4 ODE Data Payload

OdeData is a subclass of OdeMsgPayload and a parent class for all source data structures. All ODE Data structures that contain source data will have the following data elements:

Name	Type	Description	Units	Valid Min	Valid Max
receivedAt	iso-date-time	ISO Date and Time string include time zone. The format of this element is **yyyy-MM-ddThh:mm:ss.sssZ**			
serialId	string	A unique ID for each data record. The serialId consists of three parts, separated by a period: **streamId.bundleId.recordId**			

Table 20 - OdeData

5.3.2.3.4.1 ODE Advisory Data

Name	Type	Description	Units	Valid Min	Valid Max
advisoryMessage	OdeAdvisoryDetails	See 5.3.2.3.4.7.8			
centerPosition	OdePosition3D	The center point of the serviceRegion. See 5.3.2.3.4.7.16			

dataType	string	This message contains traveler advisory data.	Enum	"AdvisoryData"	"AdvisoryData"
groupID	String	unique ID used to identify an organization			
receivedAt	iso-date-time	See 5.3.2.3.4			
serialId	string	See 5.3.2.3.4			
serviceRegion	OdeGeoRegion	See 5.3.2.3.4.7.1			
timeToLive	Enum	One of: minute, halfHour, day, week, month, year			
version	Integer	See 5.3.2.3			

Table 21 – OdeAdvisoryData

5.3.2.3.4.2 ODE Aggregate Data

Name	Type	Description	Units	Valid Min	Valid Max
avgSpeed	decimal	Average speed in the ‘key’ group	m/s	0	163.8
count	integer	Number of records	each	0	9223372036854775807

dataType	string	This payload contains the Aggregate Vehicle Data calculated by the ODE from receiving vehicle data.	Enum	"AggregateData"	"AggregateData"
key	string	The key by which the data is grouped for aggregation	N/A		
maxSpeed	decimal	Maximum speed in the ‘key’ group	m/s	0	163.8
minSpeed	decimal	Minimum speed in the ‘key’ group	m/s	0	163.8
receivedAt	iso-date-time	See 5.3.2.3.4			
serialId	string	See 5.3.2.3.4			
version	Integer	See 5.3.2.3			

Table 22 - OdeAggregateData

5.3.2.3.4.3 ODE Intersection Data

Name	Type	Description	Units	Valid Min	Valid Max
centerPosition	OdePosition3D	The center point of the serviceRegion. See 5.3.2.3.4.7.16			
dataType	string	This payload contain ODE Intersection Data	Enum	"IntersectionData"	" IntersectionData "

groupID	String	unique ID used to identify an organization			
mapData	OdeMapData	See 5.3.2.3.4.4			
receivedAt	iso-date-time	See 5.3.2.3.4			
serialId	string	See 5.3.2.3.4			
serviceRegion	OdeGeoRegion	See 5.3.2.3.4.7.1			
spatData	OdeSpatData	See 5.3.2.3.4.5			
timeToLive	Enum	One of: minute, halfHour, day, week, month, year			
version	Integer	See 5.3.2.3			

Table 23 – OdeIntersectionData

5.3.2.3.4.4 Ode Map Data

Name	Type	Description	Units	Valid Min	Valid Max
dataParameters	OdeDataParameters				
intersections	Array of OdeIntersection	See 5.3.2.3.4.7.15			
layerId	Integer				

layerType	Enum	One of: NONE, MIXED_CONTENT, GENERAL_MAP_DATA, INTERSECTION_DATA, CURVE_DATA, ROADWAY_SECTION_DATA, PARKING_AREA_DATA, SHARED_LANE_DATA	Enum		
msgCnt	Integer				
msgID	Enum	One of: reserved, alaCarteMessage, basicSafetyMessage, basicSafetyMessageVerbose, commonSafetyRequest, emergencyVehicleAlert, intersectionCollisionAlert, mapData, nmeaCorrections, probeDataManagement, probeVehicleData, roadSideAlert, rtcmCorrections,	Enum		

		signalPhaseAndTimingMessage, signalRequestMessage, signalStatusMessage, travelerInformation			
name	String				

Table 24 - OdeMapData

5.3.2.3.4.5 Ode SPaT Data

Name	Type	Description	Units	Valid Min	Valid Max
dateTime	OdeDateTime	See 5.3.2.3.4.7.7			
intersections	OdeIntersectionState	See 5.3.2.3.4.7.16			
timestamp	iso-date-time	ISO Date and Time string include time zone. The format of this element is **yyyy-MM-ddThh:mm:ss.sssZ**			

Table 25 - OdeSpatData

5.3.2.3.4.6 ODE Vehicle Data

Name	Type	Description	Units	Valid Min	Valid Max
accelLat	decimal	Latitudinal acceleration, i.e. acceleration perpendicular to the direction of travel.	m/s^2	-20.00	+20.00
accelLong	decimal	Longitudinal acceleration, i.e. acceleration in the direction of travel.	m/s^2	-20.00	+20.00
accelVert	decimal	Vertical acceleration, i.e. acceleration along the z-axis	G	-3.4	+1.54
accelYaw	decimal	Rotational acceleration around the z axis	Degrees/sec	-327.67	+327.67
dataType	string	This payload has OdeVehicleDataFlat structure	Enum	"VehicleData"	"VehicleData"
dateTime	iso-date-time				
day	integer	Day of the month		1	31
elevation	decimal	Elevation	meters	-409.5	6143.9

groupId	string		unique ID used to identify an organization		
heading	decimal		Degrees	0.000	360.000
hour	integer	Hour of the day		0	23
latitude	decimal	Position latitude	Degrees	-90.0000000	+90.0000000
longitude	decimal	Position longitude	Degrees	-180.0000000	*180.0000000
minute	integer	Minute of the hour		0	59
month	integer	Month of the year		1	12
receivedAt	iso-date-time	See 5.3.2.3.4			
roadSeg	string	Road Segment ID on which vehicle is traveling	N/A	N/A	N/A
second	integer	Seconds of the minute. To account for leap seconds, the maximum value may be greater than 60.		0.000	62.000
serialId	string	See 5.3.2.3.4			
sizeLength	integer		Centimeter	0	16383
sizeWidth	integer		Centimeter	0	1023

speed	decimal		m/s	0	163.8
tempId	string				
version	Integer	See 5.3.2.3			
weatherAirPres	decimal		hectopascals (hPa)	580.0	1090.0
weatherAirTemp	decimal		Centigrade (Celcius)	-40.0	151.0
year	integer	Year		1970	9999

Table 26 - OdeVehicleDataFlat

5.3.2.3.4.7 Supporting Data Structures

The data structures described in this section and subsections are referenced by top level ODE data message schemas.

5.3.2.3.4.7.1 OdeApproach

Name	Type	Description	Units	Valid Min	Valid Max
barriers	Array of OdeBarrierLane	See 5.3.2.3.4.7.3			
computedLanes	Array of OdeVehicleComputedLane	See 5.3.2.3.4.7.21			
crosswalks	Array of OdeCrosswalkLane	See 5.3.2.3.4.7.5			
drivingLanes	Array of OdeVehicleReferenceLane	See 5.3.2.3.4.7.21			

id	Integer				
name	String				
trainsAndBuses	Array of OdeSpecialLane	See 5.3.2.3.4.7.21			

Table 27 - OdeApproach

5.3.2.3.4.7.2 OdeApproachObject

Name	Type	Description	Units	Valid Min	Valid Max
approach	OdeApproach	See 5.3.2.3.4.7.1			
egress	OdeApproach	See 5.3.2.3.4.7.1			
laneWidthCm	Integer		Centimeter		
refPoint	OdePosition3D	See 5.3.2.3.4.7.16			

Table 28 - OdeApproachObject

5.3.2.3.4.7.3 OdeArea

Only one of the following elements will be present:

Name	Type	Description	Units	Valid Min	Valid Max
circle_chosen	OdeCircle	See 5.3.2.3.4.7.13			
regionPointSet_chosen	OdeRegionPointSet	See 5.3.2.3.4.7.22			
shapePointSet_chosen	OdeShapePointSet	See 5.3.2.3.4.7.25			

Table 29 - OdeArea

5.3.2.3.4.7.4 OdeBarrierLane

Name	Type	Description	Units	Valid Min	Valid Max
barrierAttributes	Hex Encoded Octet String	With bits as defined: noData = 0 median = 1 whiteLine = 2 strippedLines = 4 doubleStrippedLines = 8 trafficCones = 16 constructionBarrier = 32 trafficChannels = 64 noCurbs = 128 lowCurbs = 256 highCurbs = 512 hovDoNotCross = 1024 hovEntryAllowed = 2048 hovExitAllowed = 4096	Centimeter		
laneNumber	Integer				
laneWidth	Integer		Centimeters		

nodeList	Array of OdeLaneOffsets				

Table 30 - OdeBarrierLane

5.3.2.3.4.7.5 OdeCrosswalkLane

Name	Type	Description	Units	Valid Min	Valid Max
connectsTo	Array of OdeLaneOffsets				
keepOutList	Integer		Centimeter		
laneAttributes	Hex Encoded Octet String	Bit Masks: noData (0) twoWayPath (1) pedestrianCrosswalk (2) bikeLane (4) railRoadTrackPresent (8) oneWayPathOfTravel (16) pedestrianCrosswalkTypeA (32) pedestrianCrosswalkTypeB (64) pedestrianCrosswalkTypeC (128)	Centimeter		
laneNumber	Integer				
laneWidth	Integer		Centimeters		

nodeList	Array of OdeLaneOffsets				

Table 31 - OdeCrosswalkLane

5.3.2.3.4.7.6 OdeDataParameters

Name	Type	Description	Units	Valid Min	Valid Max
geiodUsed	String	See DSRC Specs			
lastCheckedDate	String	See DSRC Specs			
processAgency	String	See DSRC Specs			
processMethod	String	See DSRC Specs			

Table 32 - OdeDataParameters

5.3.2.3.4.7.7 OdeDateTime

Name	Type	Description	Units	Valid Min	Valid Max
day	Integer				
hour	Integer				
minute	Integer				
month	Integer				
second	Decimal				

year	Integer				

Table 33 - OdeDateTime

5.3.2.3.4.7.8 OdeAdvisoryDataFrame

Name	Type	Description	Units	Valid Min	Valid Max
commonAnchor	OdePosition3D	See 5.3.2.3.4.7.16			
commonDirectionality	Enum	One of: forward, reverse, both			
commonLaneWidth	Integer		Centimeter		
content	OdeAdvisoryDataFrame.MessageContent	See 5.3.2.3.4.7.9			
duratonTime	Integer	Minutes duration	Minutes		
frameType	Enum	One of: unknown, advisory, roadSignage, commercialSignage			

msgId	OdeAdvisoryDataFrame.MessageId	See 5.3.2.3.4.7.9			
priority	Integer	The relative importance of the sign, a scale from zero (least important) to seven (most important).		0	7
regions	Array of: OdeValidRegion	See 5.3.2.3.4.7.30			
startTime	Integer	Minutes of the year	Minutes		
startYear	Integer		Year		
url	String	URL may link to image or other content			

Table 34 - OdeAdvisoryDataFrame

5.3.2.3.4.7.9 OdeAdvisoryDataFrame.MessageContent

Only one of the following will be present

Name	Type	Description	Units	Valid Min	Valid Max
advisory_chosen	Array of OdeAdvisoryDataFrame.MessageContent.CodeOrText				
exitService_chosen	Array of OdeAdvisoryDataFrame.MessageContent.CodeOrText				
genericSign_chosen	Array of OdeAdvisoryDataFrame.MessageContent.CodeOrText				

speedLimit_chosen	Array of OdeAdvisoryDataFrame.MessageContent.CodeOrText				
workZone_chosen	Array of OdeAdvisoryDataFrame.MessageContent.CodeOrText				

Table 35 - OdeAdvisoryDataFrame.MessageContent

5.3.2.3.4.7.10 OdeAdvisoryDataFrame.MessageContent.CodeOrText

Only one of the following elements will be present.

Name	Type	Description	Units	Valid Min	Valid Max
code_chosen	Integer				
text_chosen	String				

Table 36 - OdeAdvisoryDataFrame.MessageContent.CodeOrText

5.3.2.3.4.7.11 OdeAdvisoryDataFrame.MessageId

Only one of the following will be present.

Name	Type	Description	Units	Valid Min	Valid Max
furtherInfoID_chosen	String	This data element provides a link number to other messages (described here and in other message set standards) which relate to the same event. Use			

		zero when unknown or not present.			
roadSignID_chosen	OdeRoadSignId	See 5.3.2.3.4.7.21			

Table 37 - OdeAdvisoryDataFrame.MessageId

5.3.2.3.4.7.12 OdeAdvisoryDetails

Name	Type	Description	Units	Valid Min	Valid Max
id	String				
distType	String				
startTime	iso-date-time	ISO Date and Time string include time zone. The format of this element is **yyyy-MM-ddThh:mm:ss.sssZ**			
stopTime	iso-date-time				
travelerInfo	OdeTravelerInfo				
type	Enum	One of: spatAggregate, map, tim, ev			

Table 38 – OdeAdvisoryDetails

5.3.2.3.4.7.13 OdeCircle

Name	Type	Description	Units	Valid Min	Valid Max
center	OdePosition3D	See 5.3.2.3.4.7.16			
Radius	OdeRadius Choice Object				

Table 39 – OdeCircle

5.3.2.3.4.7.14 OdeGeoRegion

Name	Type	Description	Units	Valid Min	Valid Max
nwCorner	OdePosition3D	See 5.3.2.3.4.7.16			
seCorner	OdePosition3D	See 5.3.2.3.4.7.16			

Table 40 - OdeGeoRegion

5.3.2.3.4.7.15 OdeIntersection

Name	Type	Description	Units	Valid Min	Valid Max
approaches	Array of OdeApproachObject	See 5.3.2.3.4.7.1			
id	String				
laneWidth	Integer		Centimeters		
name	String				

orientation	Decimal	Heading	Degrees	0.000	360.000
preemptZones	Array of OdeSignalControlZone	See 5.3.2.3.4.7.21			
priorityZones	Array of OdeSignalControlZone	See 5.3.2.3.4.7.21			
refInterNum	String				
refPoint	OdePosition3D	See 5.3.2.3.4.7.16			
type	OdeIntersectionStatusObject	See 5.3.2.3.4.7.17			

Table 41 - OdeIntersection

5.3.2.3.4.7.16 OdeIntersectionState

Name	**Type**	**Description**	**Units**	**Valid Min**	**Valid Max**
id	String				
lanesCnt	Integer				
name	String				
priority	Hex encoded Octet	the active priority state data, if present	-- With bits set as follows: -- Bit 7 (MSB) Set if the state is currently active		

preempt	Hex encoded Octet	the active preemption state data, if present	-- only one active state can exist at a time, and -- this state should be sent first in any sequences -- Bits 6~4 The preempt or priority value that is -- being described. -- Bits 3~0 the state bits, indicating either a -- preemption or a priority use as follows: -- If a preemption: to follow the -- preemptState object of NTCIP 1202 v2.19f -- See PreemptState for bit definitions. -- If a prioirty to follow the -- tspInputStatus object utilized in the -- NYC ASTC2 traffic controller -- See PriorityState for bit definitions		
states	Array of OdeMovementState objects	See 5.3.2.3.4.7.19			
status	OdeIntersectionStatusObject	See 5.3.2.3.4.7.17			
timeMark	Integer				

Table 42 - OdeIntersectionState

5.3.2.3.4.7.17 OdeIntersectionStatusObject

Type	**Description**
Hex Encoded Octet	Bit Masks:

	NONE = 0
	MANUAL_CONTROL_ENABLED = 1
	STOP_TIME_ACTIVATED = 2
	IN_CONFLICT_PATH = 4
	PREEMPT_IS_ACTIVE = 8 TRANSIT_SIGNAL_PRIORITY_IS_ACTIVE = 16
	RESERVED_1 = 32
	RESERVED_2 = 64
	RESERVED_3 = 128

Table 43 - OdeIntersectionStatusObject

5.3.2.3.4.7.18 OdeLaneOffsets

Name	Type	Description	Units	Valid Min	Valid Max
LaneWidthCm	Integer		Centimeter		
xOffsetCm	Integer		Centimeter		
yOffsetCm	Integer		Centimeter		
zOffsetCm	Integer		Centimeter		

Table 44 - OdeLaneOffsets

5.3.2.3.4.7.19 OdeMovementState

Name	Type	Description	Units	Valid Min	Valid Max
currState	Integer	The current state of a Motorized lane. See DE_SignalLightState in SAE j2735 specs	For the current movement State, one of these will be present		
pedState	Enum	The current state of a Pedestrian type lane. One of: unavailable, stop, caution, walk			
specialState	Enum	The current state of a special type lane such as a dedicated train lane. One of: unknown, notInUse, arriving, present, departing			
laneCnt	Integer				
laneSet	Array of Integers	An array of lane numbers in an intersection.			
movementName	String				
pedCount	Integer			0	6000

pedDetect	Enum	One of: none, maybe, one, some			
stateConfidence	Enum	One of: unKnownEstimate, minTime, maxTime, timeLikeklyToChange			
timeToChange	Integer	the point in time this state will change	-- In units of 1/10th second from local UTC time -- A range of 0~600 for even minutes, 601~1200 for odd minutes -- 12001 to indicate indefinite time -- 12002 to be used when value undefined or unknown	0	12002
vehicleCount	Integer			0	6000
yellPedState	Enum	The next state of a			

		Pedestrian type lane. One of: unavailable, stop, caution, walk			
yellState	String	The next state of a Motorized lane. See DE_SignalLightState in SAE j2735 specs			
yellStateConfidence	Enum	One of: unKnownEstimate, minTime, maxTime, timeLikeklyToChange			
yellTimeToChange	Integer	the point in time this state will change	-- In units of 1/10th second from local UTC time -- A range of 0~600 for even minutes, 601~1200 for odd minutes -- 12001 to indicate indefinite time -- 12002 to be used when value		

			undefined or unknown		

Table 45 - OdeMovementState

5.3.2.3.4.7.20 OdePosition3D

Name	Type	Description	Units	Valid Min	Valid Max
elevation	decimal	Elevation	meters	-409.5	6143.9
latitude	decimal	Position latitude	Degrees	-90.0000000	+90.0000000
longitude	decimal	Position longitude	Degrees	-180.0000000	*180.0000000

Table 46 - OdePosition3D

5.3.2.3.4.7.21 OdeRadius

Only one of the following elements will be present:

Name	Type	Description	Units	Valid Min	Valid Max
Cm_chosen	Decimal		Centimeter	0.0	81917.5
km_chosen	Integer		Kilometers	1	5000
miles_chosen	Integer		Miles	1	2000

Table 47 - OdeRadius

5.3.2.3.4.7.22 OdeRegionOffsets

Name	Type	Description	Units	Valid Min	Valid Max
xOffsetCm	Integer		Centimeter		
yOffsetCm	Integer		Centimeter		
zOffsetCm	Integer		Centimeter		

Table 48 – OdeRegionOffsets

5.3.2.3.4.7.23 OdeRegionPointSet

Name	Type	Description	Units	Valid Min	Valid Max
anchor	OdePosition3D	See 5.3.2.3.4.7.16			
nodeList	Array of OdeRegionOffsets	See 5.3.2.3.4.7.22			

Table 49 – OdeRegionPointSet

5.3.2.3.4.7.24 OdeRoadSignId

Name	Type	Description	Units	Valid Min	Valid Max
mutcdCode	Enum	One of:			

		none, regulatory, warning, maintenance, motoristService, guide, rec			
position	OdePosition3D	See 5.3.2.3.4.7.16			
viewAngles	Array of Slice Mask Enums	One or more of the following: noHeading(0x0000), from000_0to022_5degrees(0x0001), from022_5to045_0degrees(0x0002), from045_0to067_5degrees(0x0004), from067_5to090_0degrees(0x0008), from090_0to112_5degrees(0x0010), from112_5to135_0degrees(0x0020), from135_0to157_5degrees(0x0040), from157_5to180_0degrees(0x0080), from180_0to202_5degrees(0x0100), from202_5to225_0degrees(0x0200), from225_0to247_5degrees(0x0400), from247_5to270_0degrees(0x0800), from270_0to292_5degrees(0x1000), from292_5to315_0degrees(0x2000), from315_0to337_5degrees(0x4000), from337_5to360_0degrees(0x8000), allHeadings(0xFFFF);			

Table 50 - OdeRoadSignId

5.3.2.3.4.7.25 OdeShapePointSet

Name	Type	Description	Units	Valid Min	Valid Max
anchor	OdePosition3D	See 5.3.2.3.4.7.16			
directionality	Enum	One of: forward, reverse, both			
laneWidth	Integer		Centimeter		
nodeList	Array of OdeLaneOffsets	See 5.3.2.3.4.7.16			

Table 51 – OdeShapePointSet

5.3.2.3.4.7.26 OdeSignalControlZone

Name	Type	Description	Units	Valid Min	Valid Max
data	OdeSignalControlZone.Data				
name	String				
pValue	String				

Table 52 - OdeSignalControlZone

5.3.2.3.4.7.27 OdeSignalControlZone.Data

Only one of the following will be present.

Name	Type	Description	Units	Valid Min	Valid Max
laneSet_chosen	Array of Integers				
zones_chosen	Array of Zones				

Table 53 - OdeSignalControlZone.Data

5.3.2.3.4.7.28 OdeSpecialLane

Name	Type	Description	Units	Valid Min	Valid Max
connectsTo	Array of OdeLaneOffsets				
keepOutList	Integer		Centimeter		

laneAttributes	Hex Encoded Octet String	Bit Masks:
		NO_DATA = 0;
		EGRESS_PATH = 1;
		RAILROAD_TRACK = 2;
		TRANSIT_ONLY_LANE = 4;
		HOV_LANE = 8;
		BUS_ONLY = 16;
		VEHICLES_ENTERING = 32;
			Centimeter		

		VEHICLES_LEAVING = 64;
		RESERVED = 128;			

laneNumber	Integer				
laneWidth	Integer		Centimeters		
nodeList	Array of OdeLaneOffsets				

Table 54 - OdeSpecialLane

5.3.2.3.4.7.29 OdeTravelerInfo

Name	Type	Description	Units	Valid Min	Valid Max
dataFrameCount	Integer				
dataFrames	Array of OdeAdvisoryDataFrame	See 5.3.2.3.4.7.8			
msgID	Enum	One of: reserved, alaCarteMessage, basicSafetyMessage, basicSafetyMessageVerbose, commonSafetyRequest, emergencyVehicleAlert, intersectionCollisionAlert, mapData, nmeaCorrections, probeDataManagement, probeVehicleData, roadSideAlert, rtcmCorrections, signalPhaseAndTimingMessage,			

		signalRequestMessage, signalStatusMessage, travelerInformation			
`packetID`	`Hex encoded octet string`	A message link value used to connect to other supporting messages in other formats			
`urlB`	`String`	A valid internet style URI / URL in the form of a text string which will form the base of a compound string which, when combined with the URL-Short data element, will link to the designated resource.			

Table 55 - OdeTravelerInfo

5.3.2.3.4.7.30 OdeValidRegion

Name	**Type**	**Description**	**Units**	**Valid Min**	**Valid Max**
area	OdeArea Choice Object	See 5.3.2.3.4.7.3			
directions	Array of Slice Mask Enums	One or more of the following: noHeading(0x0000), from000_0to022_5degrees(0x0001), from022_5to045_0degrees(0x0002),			

		from045_0to067_5degrees(0x0004), from067_5to090_0degrees(0x0008), from090_0to112_5degrees(0x0010), from112_5to135_0degrees(0x0020), from135_0to157_5degrees(0x0040), from157_5to180_0degrees(0x0080), from180_0to202_5degrees(0x0100), from202_5to225_0degrees(0x0200), from225_0to247_5degrees(0x0400), from247_5to270_0degrees(0x0800), from270_0to292_5degrees(0x1000), from292_5to315_0degrees(0x2000), from315_0to337_5degrees(0x4000), from337_5to360_0degrees(0x8000), allHeadings(0xFFFF);			
Extent	Enum	One of: useInstantlyOnly, useFor3meters, useFor10meters, useFor50meters, useFor100meters, useFor500meters, useFor1000meters, useFor5000meters, useFor10000meters, useFor50000meters, useFor100000meters, forever			

Table 56 - OdeValidRegion

5.3.2.3.4.7.31 OdeVehicleComputedLane

Name	Type	Description	Units	Valid Min	Valid Max
connectsTo	Array of OdeLaneOffsets	See 5.3.2.3.4.7.16			
keepOutList	Array of OdeLaneOffsets	See 5.3.2.3.4.7.16			
laneAttributes	OdeVehicleLaneAttributes	See 5.3.2.3.4.7.32			
laneNumber	Integer				
laneWidth	Integer		Centimeters		
lineOffset	Integer				
nodeList	Array of OdeLaneOffsets	See 5.3.2.3.4.7.16			
refLaneNum	Integer				

Table 57 – OdeVehicleComputerLane

5.3.2.3.4.7.32 OdeVehicleLaneAttributes

Type	Description
Hex Encoded Octet	Bit Masks:

	NO_LANE_DATA = 0;
	EGRESS_PATH = 1;
	MANEUVER_STRAIGHT_ALLOWED = 2;
	MANEUVER_LEFT_ALLOWED = 4;
	MANEUVER_RIGHT_ALLOWED = 8;
	YIELD = 16;
	MANEUVER_NO_U_TURN = 32;
	MANEUVER_NO_TURN_ON_RED = 64;
	MANEUVER_NO_STOP = 128;
	NO_STOP = 256;
	NO_TURN_ON_RED = 512;
	HOV_LANE = 1024;
	BUS_ONLY = 2048;
	BUS_AND_TAXI_ONLY = 4096;
	MANEUVER_HOV_LANE = 8192;
	MANEUVER_SHARED_LANE = 16384; MANEUVER_BIKE_LANE = 32768

Table 58 - OdeVehicleLaneAttributes

5.3.2.3.4.7.33 OdeVehicleReferenceLane

Name	Type	Description	Units	Valid Min	Valid Max
`connectsTo`	`Array of OdeLaneOffsets`	See 5.3.2.3.4.7.16			

keepOutList	Array of OdeLaneOffsets	See 5.3.2.3.4.7.16			
laneAttributes	OdeVehicleLaneAttributes	See 5.3.2.3.4.7.32			
laneNumber	Integer				
laneWidth	Integer		Centimeters		
nodeList	Array of OdeLaneOffsets				

Table 59 - OdeVehicleReferenceLane

5.3.2.3.4.7.34 Zone

Name	Type	Description	Units	Valid Min	Valid Max
enclosed	Array of Integers	Array of lane numbers			
laneWidth	Integer				
nodeList	Array of OdeLaneOffsets	See 5.3.2.3.4.7.16			

Table 60 - Zone

5.3.2.3.5 ODE Full Message

When ODE receives a message that is not one of the known data messages described above, the message is sent to the client in full as an OdeFullMessage.

Name	Type	Description	Units	Valid Min	Valid Max

`dataType`	string		Enum	"other"	"other"
`fullMessage`	`string`	A full data message received and passed on verbatim.			
`version`	`Integer`	See 5.3.2.3			

Table 61 - OdeFullMessage