

**Multi Modal Intelligent Traffic Signal System**

**Field Deployment – User Manual**

Revision 0.0

(Initial Release)

July 12, 2020

**Table of Contents**

1. Purpose of Document 3

2. Requirements: 3

3. Simulation Model 3

3.1. Configuring Connected Vehicle 3

3.2. Signal Controller 4

3.2.1. Configuring Econolite ASC/3 Signal Controller 4

3.2.2. Configuring MaxTime Signal Controller 6

4. VISSIM Driver Model DLL 7

5. Simulation-Tools 8

5.1. Message-Distributor 9

5.2. Priority-Request-Generator-Server 10

6. Deployment – Docker Containers 13

**List of Figures:**

Figure 1: Vissim simulation model of multiple intersections 3

Figure 2:Defining Connected Vehicle Type 4

Figure 3: Prepare to setup the parameters 5

Figure 4: Define the parameters 5

Figure 5: Starting MaxTime signal Controller 6

Figure 6: Defining the IP address in MaxTime Signal Controller 6

Figure 7: Flow diagram of Priority-Request-Generator-Server 12

# Purpose of Document

This document is an instruction guide for deploying Multi-Modal Intelligent Traffic Signal System (MMITSS) applications in the real world. The document contains the detailed configuration and usage instructions for deploying the MMITSS software components in the docker container.

# Deployment – Docker Containers

**2.1. MRP containers**

To deploy MMITSS in the simulation environment, following steps can be followed:

**Step1:** Create configuration file

It is required to create mmitss-phase3-master-config.json, mmitss-data-external-clients.json configuration files for mrp container and mmitss-phase3-master-config.json for vsp container.

{

"HostIp": "xxx.xxx.xxx.xxx",

"SourceDsrcDeviceIp": "xxx.xxx.xxx.yyy",

"IntersectionName": "xxx",

"MapPayload":001283fe38083020315abe2149d0eecf1800a0000271c4fcbd028280",

"IntersectionID" : XXXX,

"RegionalID" : 0,

"DataCollectorIP": "xxx.xxx.xxx.xyx",

"HMIControllerIP": "xxx.xxx.xxx.yxx",

"MessageDistributorIP": " xxx.xxx.xxx.zzz ",

"PriorityRequestGeneratorServerIP": "xxx.xxx.xxx.zzz",

"VehicleType" : 6,

"Logging" : "True",

"SRMTimedOutTime" : 10.0,

"PortNumber":

{

"MessageTransceiver":

{

"MessageSender": 10003,

"MessageReceiver": 10002,

"MessageEncoder": 10003,

"MessageDecoder": 10002

},

"MessageDistributor": 5000,

"RsmDecoder": 10006,

"OBUBSMReceiver": 10005,

"HostBsmDecoder": 10005,

"TrajectoryAware": 20001,

"PriorityRequestServer": 20002,

"PrioritySolver": 20003,

"PriorityRequestGenerator": 20004,

"TrafficControllerInterface": 20005,

"TrafficControllerCurrPhaseListener": 20006,

"TrafficControllerTimingPlanSender": 20007,

"PerformanceObserver": 20008,

"HMIController": 20009,

"PrioritySolverToTCIInterface": 20010,

"SignalCoordination": 20011,

"MapSPaTBroadcaster": 6053,

"DsrcImmediateForwarder": 1516,

"PriorityRequestServer\_SendSSM": 50003,

"DataCollector": 30006,

"SnmpEngine": 20020,

"SnmpEngineInterface": 20021,

"PriorityRequestGeneratorServer": 20022

},

"psid":

{

"map": "E0000017",

"spat": "8002",

"rsm": "8003",

"srm": "E0000019",

"ssm": "E0000020",

"bsm": "20"

},

"msgId":

{

"map": "0012",

"spat": "0013",

"rsm": "0021",

"srm\_lower": "001d",

"srm\_upper": "001D",

"ssm\_lower": "001e",

"ssm\_upper": "001E",

"bsm": "0014"

},

"SignalController":

{

"IpAddress": " xxx.xxx.xxx.yyy",

"NtcipPort": 502,

"TimingPlanUpdateInterval\_sec": 60,

"NtcipBackupTime\_sec": 300,

"Vendor": "Econolite",

"TimingPlanMib": "/nojournal/bin/EconoliteMib.py",

"InactiveVehPhases":[],

"InactivePedPhases":[],

"SplitPhases":

{

"1": 6,

"3": 8,

"5": 2,

"7": 4

},

"PermissiveEnabled":

{

"1": true,

"3": true,

"5": true,

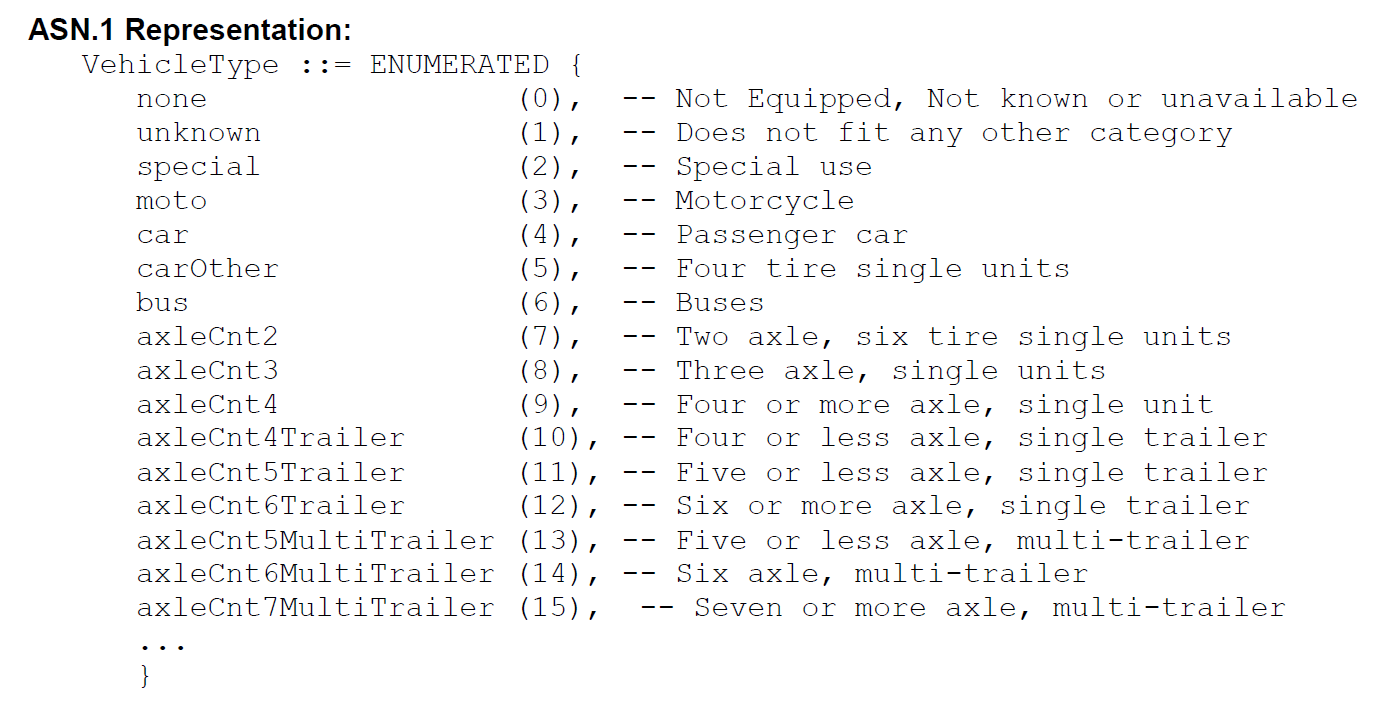
"7": true

}

}

}

1. For mrp container, “*HostIp”,* “*SourceDsrcDeviceIp”, “IntersectionName”, “MapPayload”, “IntersectionID”*, *signal controller* “*IpAddress”, “NtcipPort”,* and “*NtcipBackupTime\_sec”, “Vendor”, “TimingPlanMib”* are required to be specified. The *HostIP* must match the ip address of the connected vehicle co-processor (CVCP). The “*SourceDsrcDeviceIp”*  The map payload can be obtained by creating an intersection map using USDOT map tool (<https://webapp.connectedvcs.com/isd/>).
2. For vsp container, *“HostIp”,* “*SourceDsrcDeviceIp”,* “*VehicleType”* are required to be specified. The vehicle type has to be specified based on *J2735 2016* standard.



1. Create a log folder in the nojournal/bin directory for each intersection and simulation tools.

The structure of mmitss-message-distributor-config.json is discussed in the section 5.1.

**Step2:** Define the docker files

It is required to specify two dockerfiles: 1. For MMITSS roadside software component Dockerfiles and 2. For simulation tools Dockerfile\_simulation-tools. The dockerfiles are required to be placed in the mmitss/bin directory.

An example of the Dockerfile is as follows:

#---------------------------------------------------------------------------#

# Dockerfile to build an x86 platform image for an intersection #

#---------------------------------------------------------------------------#

FROM mmitssuarizona/mmitss-base-x86:1.3

COPY TrafficControllerInterface/x86/M\_TrafficControllerInterface /mmitss

COPY SnmpEngine/x86/M\_SnmpEngine /mmitss

COPY PriorityRequestSolver/x86/M\_PriorityRequestSolver /mmitss

COPY PriorityRequestServer/x86/M\_PriorityRequestServer /mmitss

COPY MapSpatBroadcaster/x86/M\_MapSpatBroadcaster /mmitss

COPY supervisord.conf /mmitss

CMD ["/usr/bin/supervisord"]

An example of the Dockerfile\_simulation-tools is as follows:

#---------------------------------------------------------------------------#

# Dockerfile to build an x86 platform image for an intersection #

#---------------------------------------------------------------------------#

FROM mmitssuarizona/mmitss-base-x86:1.3

COPY PriorityRequestGeneratorServer/x86/M\_PriorityRequestGeneratorServer /mmitss

COPY MessageDistributor/x86/M\_MessageDistributor /mmitss

RUN mkdir -p /mmitss/map

COPY supervisord\_simulation-tools.conf /mmitss/supervisord.conf

CMD ["/usr/bin/supervisord"]

**Step 3:** Define supervisrod.conf and supervisor\_simulation-tool.conf file

To monitor the applications in the container, it is required to define two supervisord.conf files. For MMITSS roadside software components supervisrod.conf and for simulation tools supervisrod-simulation-tools.conf. They has to be placed in the same level directory structure of Dockerfile (mmitss/bin).

**Step 4:** Define docker-compose.yml file

To build and run all the software components in the docker container, docker-compose.yml file has to be specified. The yml file has to be placed in the same level directory structure of Dockerfile (mmitss/bin).

1. For each intersection, container name, source, and ipv4\_address are required to be specified. Container name can be intersection name, source must be the directory of configuration files and ipv4 address has to match the Host IP address which is defined in the mmitss-phase3-master-config.json file.
2. For simulations tools dockerfile name has to defined along with container name, source, ipv4\_address. Container name can be simulation-tools, source must be the directory of configuration files and ipv4 address has to match the *message-distributor* IP address which is defined in the mmitss-phase3-master-config.json file.
3. The ethernet interface of the computer and subnet has to be defined under the network. The ethernet interface of the computer that will be used to communicate with the containers can be obtained by using following command in a terminal on the computer:

ifconfig

An example of the docker-compose.yml file for two intersections and simulation tools is as follows:

version: "3.8"

services:

daisy-gavilan:

build:

context: ./

container\_name: daisy-gavilan

volumes:

- type: bind

source: $MMITSS\_ROOT/mmitss/bin/corridors/Anthem/Daisy-Gavilan/nojournal

target: /nojournal

networks:

mmitss\_vlan:

ipv4\_address: xxx.xxx.xxx.xxx

daisy-dedication:

build:

context: ./

container\_name: daisy-dedication

volumes:

- type: bind

source: $MMITSS\_ROOT/mmitss/bin/corridors/Anthem/Daisy-Dedication/nojournal

target: /nojournal

networks:

mmitss\_vlan:

ipv4\_address: xxx.xxx.xxx.yyy

simulation-tools:

build:

context: ./

dockerfile: Dockerfile\_simulation-tools

container\_name: simulation-tools

volumes:

- type: bind

source: $MMITSS\_ROOT/mmitss/bin/corridors/simulation-tools/nojournal

target: /nojournal

networks:

mmitss\_vlan:

ipv4\_address: xxx.xxx.xxx.zzz

networks:

mmitss\_vlan:

driver: macvlan

driver\_opts:

parent: eno2

ipam:

config:

- subnet: xxx.xxx.xxx.0/24

**Step 5:** Build and run docker container

1. Run the VISSIM simulation model
2. Open a terminal in Linux box and directed to mmitss/scripts directory
3. Run following script to build all the software components:

mmitss\_docker\_make\_all\_for\_x86.sh

1. To build and run the docker containers go to mmitss/bin directory and execute the following command:

docker-compose up --build

1. To monitor the containers execute the following command:

docker-compose exec <container\_name> /bin/bash

It will direct inside the container. Following command can be executed to monitor the applications using supervisor.

supervisordctl

1. To stop all the containers, execute the following command:

docker-compose stop

To stop a specific container, execute the following command

docker-compose stop <container\_name>

1. To start container the execute the following command:

docker-compose start

To start a specific container, execute the following command

docker-compose start <container\_name>

1. To down the whole macvlan network of docker containers, execute the following command

docker-compose down

1. To up the whole macvlan network of docker containers, execute the following command

docker-compose up