# FY14 MAW Desktop Display Application – Programmer’s Overview

## Introduction

The purpose of this document is to provide a high-level overview of the MAW desktop display application and underlying code. It is intended for readers with extensive experience with the technologies employed in the development and deployment of the application, such as javascript, Sencha Touch, Aptana IDE, and web servers.

Note that the MAW desktop is a modified version of the EMDSS display, so the underlying code is largely identical. MAW desktop consists of a simplified user interface, yet with some additional time-selection functionality.

## Application Description

The MAW desktop display is a javascript application intended to demonstrate the capabilities of the VDT system developed under TOPR4.

The display exposes three primary views of the underlying VDT and vehicle data:

1. **Map View**  
   MAW desktop shows a map containing colored symbols at the locations where forecasts are available. The colored symbols indicate the worst alert provided by the VDT during the user-selected time period. The time period is selected via drop-down menus at the top of the map.
2. **Road Segment Detail View**  
   Clicking on a road segment symbol in the map view brings up a detail view of the selected object. This view shows the VDT-generated alert for each hour in the selected time period.

The display is designed to run on a desktop computer in a browser that supports HTML5. Because of the underlying javascript libraries used for development, the application should also run on mobile devices. However, the user interface was not tailored in this phase of development to the smaller format of mobile devices, so additional usability study and design would be needed to adapt the user interface to this use.

## Application Architecture

MAW desktop is the front-end component of a typical three-tier architecture: data, services, and display. Architectural decisions during development held the goal of keeping the display client as thin as possible in order to provide the best-possible user experience and to reduce complexity and brittleness of the display code.

The VDT backend integrates mobile observations into weather and road condition forecasts produced by Dicast. A service layer processes the VDT output and vehicle observations and derives display-specific datasets that are provided to MAW desktop on request. The display application does very little with the data returned from the services layer besides exposing it to the user.

Most data processing for the display is performed in the services layer. Display responsibilities are concentrated on user interactions and data refresh. The primary responsibilities of the display code are to:

* Poll the services layer for updates to the site data, and refresh the dataset when necessary.
* Derive the alert for each site to display on the map, based on the user’s selected time range.
* Display appropriate alerts on the map, based on user-selected alert time range.
* Handle user interactivity, such as changing the selected time range.
* Manage other user actions, such as clicking on a site to show the detail view.

MAW desktop employs the Sencha Touch javascript framework (version 2.2.1) as the basis for the application structure. Much of the application architecture and code design for MAW desktop is dictated by the conventions of Sencha Touch. Any developer working on MAW desktop will need extensive experience with Sencha Touch in order to understand and modify the code. The primary components of the display code are described here as means of introduction:

Controller  
MAW desktop utilizes a single controller – the Application controller. The Application controller starts up timers to perform the data refresh operations, manages the map, and handles most of the user interactions.

Views  
MAW desktop consists of two primary Sencha views:

* Main – the map view showing alerts for the selected time
* Road Segment – the list of alerts for each hour for a selected segment

Data  
Sencha Touch employs data models and data stores to manage data retrieved from the services layer. The primary data types are listed here:

* Latest data times – the timestamp of the latest available data for each type.
* Sites – weather and road condition data for all sites

Map  
The map display in MAW desktop is based on the GoogleMaps javascript API v3. When sites and vehicles are retrieved from the services layer, they are added to the map object as markers. Callbacks attached to those markers execute controller functions to respond to user actions.

### Client-Server Interaction

MAW desktop uses a REST-like protocol to request data from the services layer. Data is returned in JSON format. Below are example requests and responses from each of the services:

**Latest available data**  
Service Name: datatime  
  
URL: [http://www.ral.ucar.edu/projects/rdwx\_mdss/proxy.php?path=/datatime/&state=minnesota\_vdt](http://www.ral.ucar.edu/projects/rdwx_mdss/AK_mdss/proxy.php?path=/datatime/&state=minnesota_vdt)  
  
JSON Response:  
[

{

"latest\_time": "201402201855",

"dir": "latest\_vehicles"

}, {

"latest\_time": "201402201848",

"dir": "rec\_treatment"

}, {

"latest\_time": "201402201830",

"dir": "road\_wx\_dir"

}, {

"latest\_time": "201402201855",

"dir": "district\_alerts"

}

]

**Site data**  
Service Name: district\_alerts  
  
URL: [http://www.ral.ucar.edu/projects/rdwx\_mdss/proxy.php?path=/district\_alerts&state=minnesota\_vdt](http://www.ral.ucar.edu/projects/rdwx_mdss/AK_mdss/proxy.php?path=/district_alerts&state=minnesota_vdt)  
  
JSON Response:  
{

"data\_time": "201402201900",

"districts": [

{

"district\_name": "Minnesota",

"max\_lat": 49.70000076293945,

"max\_lon": -88.5,

"min\_lat": 43.40000152587891,

"min\_lon": -97.69999694824219,

"sites": [

{

"desc": "MN ROAD SEGMENT Interstate 94 1",

"hr06\_alert\_code": "alert",

"hr24\_alert\_code": "clear",

"hr72\_alert\_code": "clear",

"is\_road\_cond\_site": true,

"is\_rwis\_site": false,

"is\_wx\_obs\_site": false,

"lat": 46.84318161010742,

"lon": -96.63314819335938,

"obs\_alert\_code": "clear",

"site\_id": "M00001",

"site\_num": 72753066,

"time\_series": [

{

"alert\_code": "clear",

"chemical": "apply chem",

"pavement": "dry",

"plow": "plow",

"precip": "none",

"road\_temp": 32.0,

"time": "201402201905",

"treatment\_alert\_code": "alert",

"visibility": "normal"

}, {

"alert\_code": "alert",

"chemical": "none",

"pavement": "slick, icy",

"plow": "none",

"precip": "moderate snow",

"road\_temp": 32.0,

"time": "201402202000",

"treatment\_alert\_code": "clear",

"visibility": "normal"

},

...

{

"alert\_code": "clear",

"chemical": "none",

"pavement": "dry",

"plow": "none",

"precip": "none",

"road\_temp": 24.0,

"time": "201402211800",

"treatment\_alert\_code": "clear",

"visibility": "normal"

}

]

},

...

{

"desc": "Effie MN-1 Mile Post 194",

"hr06\_alert\_code": "alert",

"hr24\_alert\_code": "warning",

"hr72\_alert\_code": "warning",

"is\_road\_cond\_site": false,

"is\_rwis\_site": true,

"is\_wx\_obs\_site": false,

"lat": 47.84040069580078,

"lon": -93.48519897460938,

"obs\_alert\_code": "warning",

"site\_id": "MN052",

"site\_num": 72747030,

"time\_series": [

{

"alert\_code": "warning",

"chemical": "apply chem",

"pavement": "ice possible",

"plow": "plow",

"precip": "",

"road\_temp": 38.0,

"time": "201402201905",

"treatment\_alert\_code": "alert",

"visibility": "normal"

}, {

"alert\_code": "warning",

"chemical": "apply chem",

"pavement": "wet",

"plow": "plow",

"precip": "moderate snow",

"road\_temp": 37.0,

"time": "201402202000",

"treatment\_alert\_code": "alert",

"visibility": "normal"

},

...

{

"alert\_code": "warning",

"chemical": "apply chem",

"pavement": "slick, snowy",

"plow": "plow",

"precip": "moderate snow",

"road\_temp": 20.0,

"time": "201402211800",

"treatment\_alert\_code": "alert",

"visibility": "normal"

}

]

}

]

}

],

"hr06\_alert\_summary\_code": "alert",

"hr24\_alert\_summary\_code": "alert",

"hr72\_alert\_summary\_code": "alert",

"obs\_alert\_summary\_code": "alert"

}

## Development Environment

Many options are available to developers working with Sencha Touch to develop javascript displays, from working on the command line with a text editor to employing integrated development environments (IDEs) purchased from developers of software tools. The following list details the toolset used at the Research Applications Laboratory to develop, test, and run the MAW desktop display:

* Platform: Apple MacbookPro 2.53 GHz Core i5 with 8GB RAM
* OS: OS X 10.8.5
* UI Framework: Sencha Touch version 2.2.1
* Mapping: Google Maps API v3
* CSS Compilation: Compass 0.12.2
* IDE: AptanaStudio 3.0 build 3.3.2.201302081546
* Run Environment: Google Chrome Version 32.0.1700.107
* Local Web Services: MAMP Version 2.1.1
* Debugging: Ripple Emulator Version 0.9.15