# FY14 MAW Desktop Display Application – Installation and Configuration

## Introduction

The purpose of this document is to provide instructions for installation and configuration of the MAW desktop display application code delivered to FHWA in February 2014. Installation requires experience with web servers and some knowledge of debugging javascript applications in web browsers.

Note that the MAW desktop display is a modified version of the EMDSS display. The procedure for installation and configuration of MAW desktop is therefore identical to that used for setting up EMDSS.

## Source Code Bundle

Source code for MAW desktop is bundled in a file named “MAW\_Desktop\_FY14\_src.tgz”. The files it contains must be installed on a web server accessible to any intended users of the application.

## Services Layer

In order to provide MAW desktop with data, the service layer must be installed and available on the web. This is provided as part of the VDT. Please install according to the instructions and take note of the location of these services. For the purposes of this document, the variable $LOCAL\_SERVICES\_URL will be used to designate the valid web address of the services layer from the host where the MAW desktop display is deployed. Note that $LOCAL\_SERVICES\_URL need not be a publicly-accessible web address – it can be behind a firewall that prevents public access. But the services have to be accessible from the web host where the MAW desktop display is deployed.

## Same-Origin Policy

Javascript applications are constrained by the “Same-Origin Policy” that requires all services accessed by an application be on the same site as the application. MAW desktop uses a proxy on the originating site to forward service requests, in case your configuration requires that the services be hosted on another site, or from behind a firewall. The provided file proxy.php must be configured to point to the host and port where the service layer is deployed. Edit this file and change the variables $destinationURL and $port as appropriate. The values assigned to these variables should be a host name and port that can be resolved from the host where the MAW desktop display code is deployed. The specified host name and port must point to the services host.

## Configuring the Services Layer Web Server

The services layer is comprised of a set of python scripts driven by Web Services Gateway Interface (wsgi) scripts managed by the web server. Below are some Apache configuration parameters (in the file httpd.conf) showing an example of how to get this set up.

The services configuration at the Reasearch Applications Laboratory accepts requests on port 8080. This is specified by the following directive in the Apache configuration file:

**Listen** 8080

The Apache wsgi module is loaded on web server startup with this directive:

LoadModule wsgi\_module modules/mod\_wsgi.so

The python home and python path are specified here:

**WSGIPythonHome** /d1/local   
**WSGIPythonPath** /home/dicast/mdss\_view:/home/dicast/scripts/python:/home/dicast/aiew:/home/vii/scripts/python

Directories holding scripts run by the web server must be granted access as follows:

<**Directory** /home/dicast/mdss\_view>  
 **Order** allow,deny  
 **Allow** from all  
</**Directory**>

Each wsgi script is given an alias via the following directives:

**WSGIScriptAlias** /latest\_vehicles /home/dicast/mdss\_view/latest\_vehicles.wsgi  
**WSGIScriptAlias** /datatime /home/dicast/mdss\_view/data\_time.wsgi  
**WSGIScriptAlias** /district\_alerts\_vdt /home/dicast/mdss\_view/district\_alerts\_vdt.wsgiModule authn\_file\_module modules/mod\_authn\_file.so  
**WSGIScriptAlias** /plots /home/dicast/mdss\_view/plots.wsgi  
**WSGIScriptAlias** /plots\_vdt /home/dicast/mdss\_view/plots\_vdt.wsgi

## Deploying the Application

The MAW desktop display is a javascript application, so it requires no compilation. It is deployed by unpacking the source code in a location accessible to a web server, then changing the contents of a single file to configure the application. The following sections describe how to deploy the application, first in a debug environment for testing and modification, then in a production environment for wider use.

## Deploying the Application in a Debug Environment

For development, testing, and debugging, the application should be installed on a local host. The example shown here is for debug installation on an Apple MacBook Pro with MAMP installed.

In order to comply with the constraints of the same-origin policy, an additional proxy will be necessary for this configuration, directing all service requests first to the debug host (where the application is deployed), which are then forwarded to a public web server and then finally forwarded to the services host itself.

## Installation and Configuration

1. cd ~/
2. mkdir test\_maw
3. cd test\_maw
4. tar –zxvf MAW\_Desktop\_FY14\_src.tgz
5. copy ./proxy.php to a web server where you plan to deploy the production version of MAW desktop. Ensure it is placed where it has the path: WebRoot/projects/rdwx\_mdss/proxy.php (note, this location can be changed, but any changes must be reflected in the Config.ps URLs edited below)
6. Edit the proxy.php file you placed on the production web site as follows:
   1. Change $destinationUrl to point to the services host
   2. Change $port to the Listen port configured for the services host
7. edit ./projects/rdwx\_mdss/proxy.php as follows:
   1. Change $site to point to the production server domain
   2. Change $port to point to the production server http port
   3. Change $uri to point to the path to the proxy.php file placed on the production server in 5 and 6, above
8. start MAMP
9. open MAMP Preferences and configure as follows:
   1. Set the Apache Document Root to ~/test\_maw
   2. Set the Apache port to 8888
10. click ‘Start Servers’ in MAMP
11. edit the file ./app/util/Config.js as follows:
    1. Comment out the seven lines labeled “RAL Production” by placing two forward slashes at the beginning of each line
    2. Un-comment the seven lines labeled “Localhost Production” by removing the two forward slashes at the beginning of each line
    3. Leave the forceSourceDomain: as ‘localhost’
    4. Modify the URLs if you have changed the location of the proxy.php file
12. open GoogleChrome and browse to <http://localhost:8888/?state=minnesota>
13. the javascript console and debug utilities are available in GoogleChrome by pressing command-option-J

## Deploying the Application in a Production Environment

Deploying the application on a production server is slightly simpler due to the fact that MAMP is not used and the second proxy is not needed in this environment. A single proxy will direct service requests to the service host.

## Installation and Configuration

1. log in to the production server
2. cd [INSTALL\_DIR]
3. mkdir maw
4. cd maw
5. tar –zxvf MAW\_Desktop\_FY14\_src.tgz
6. copy ./proxy.php to the following location on your production server. Ensure it is placed where it has the path: WebRoot/projects/rdwx\_mdss/proxy.php (note, this location can be changed, but any changes must be reflected in the Config.ps URLs edited below)
7. Edit the proxy.php file as follows:
   1. Change $destinationUrl to point to the services host
   2. Change $port to the Listen port configured for the services host
8. edit the file ./app/util/Config.js as follows:
   1. Edit out the seven lines labeled “RAL Production”
9. Change the forceSourceDomain: to your production domain
10. Modify the URLs so they reflect your production domain
11. Modify the URLs if you have changed the location of the proxy.php file
12. open GoogleChrome and browse to <http://[YOUR_DOMAIN]/[INSTALL_URL]/maw/?state=minnesota>
13. the javascript console and debug utilities are available in GoogleChrome by pressing command-option-J