

# **Deployment Guide for the Electronic Freight Management (EFM) Package**

Prepared for:

**U.S. Department of Transportation**

Prepared by:

**Battelle**

**July 17, 2009**



# TABLE OF CONTENTS

	<u>Page</u>
<b>BACKGROUND</b> .....	<b>1</b>
<b>INTENDED AUDIENCE</b> .....	<b>2</b>
<b>ARCHITECTURE OF EFM</b> .....	<b>2</b>
Communication Model .....	2
Deployment Model .....	3
Partner Instances .....	3
<b>DEPLOYMENT REQUIREMENTS</b> .....	<b>3</b>
Operating System.....	5
Servers.....	6
Software .....	6
Network Connections and Firewalls.....	6
<b>GLASSFISH AND MYSQL INSTALLATION</b> .....	<b>7</b>
Installation Steps.....	7
<b>REFERENCES</b> .....	<b>11</b>

## List of Figures

Figure 1. Prototypical EFM Partner Instance .....	4
---	---

## BACKGROUND

The EFM Initiative is intended to improve the efficiency and productivity of freight movements by evaluating and promoting innovative e-business concepts that support coordination and information sharing among supply chain partners. EFM is an open-architecture, Internet-based solution that promotes the use of standards to allow supply chain partners to efficiently track freight shipments as they move through the supply chain. EFM provides shippers—the supply chain owners—with visibility to meet very tight performance standards and improve operational efficiencies. EFM offers uniform access to existing customized database formats, computing platform independence, and adaptable services. EFM allows each supply chain partner to exchange data with other supply chain partners via Web services using eXtensible Markup Language (XML) data standards in a service oriented architecture (SOA) and open-source software products. The framework employs secure encryption and digital certificates, ensuring that any information exchanged between partners is authorized and secure, that data is not corrupted in-transit, and that the data transmitted is complete. EFM provides a gateway for automated interfaces and software capabilities that are designed to support computer-to-computer interactions over the Internet.

Accurate, visible information provides supply chain partners with the actionable intelligence they need to improve operational efficiency and increase agility in a fast-paced global business environment. Without this information, supply chain partners can face delayed shipments, disrupted assembly lines, congested cargo transfer points, and stressed inventories. Many large firms use logistics software and electronic data exchange to standardize data flows...but small to medium-sized firms often do not because of the high implementation cost and technical expertise required to effectively use the software and standards. The Electronic Freight Management (EFM) framework supports in-transit visibility to all supply chain owners, from the largest to the smallest. EFM can be used by all supply chain partners—from shippers to 3PLs to customs brokers—creating a truly integrated, “shared view” of the status of shipments across the globe and helping to increase the competitiveness and the effectiveness of the supply chain.

All supply chain partners can benefit from using supply chain visibility tools like EFM. While as few as two partners in the supply chain can benefit from using EFM, the value and operational efficiencies grow as more supply chain partner’s link into EFM. As more partners adopt EFM, fewer manual transactions are required; a more complete “shared picture” among partners enables better and more responsive decision-making.

As part of this EFM Initiative, USDOT has sponsored the development of what is known as the EFM Package. The EFM Package is a collection of documents and computer source code which may be downloaded, free of charge, for use by organizations wishing to take advantage of the benefits of EFM. The complete set of documents and the source code is available at the EFM website <http://efm.us.com>.

## INTENDED AUDIENCE

This document is one of several in the EFM Package and is provided as a guide for supply chain partners wishing to deploy web services under the EFM Package architecture. By choosing to deploy web services, a partner can either respond to a request by an EFM client seeking information from this partner, or, the partner can be a ‘listener’, waiting for the receipt of triggered ‘pushes’ on behalf of other partners.

This document summarizes the deployment environment and specifications as they pertain to the installation of the software modules provided as part of EFM. The information contained in this guide was based on a reference implementation of EFM, using the services included in the EFM package, but has been generalized and can serve as a starting point for all partners wishing to implement EFM. It assumes that the partner has performed the necessary work to tailor the existing services or build out additional services and have compiled the necessary software modules from the ‘tailored’ EFM package. This document will then serve to offer guidance on how to setup and configure the deployment environment.

## ARCHITECTURE OF EFM

### Communication Model

EFM deployments support supply-chain partner interaction between individual partner instances and between partners and/or the EFM operating environment as brokered by the EFM Platform.

The general pattern is as follows:

1. The buyer party in the supply chain initiates a consignment along with relevant information about that consignment (shipping container information, order information, etc.). This data gets sent via ForwardingInstruction to all relevant partners in the supply chain. Receipt of this message is an indication that TransportationStatus messages should be forwarded to the buyer’s EFM instance.
2. As consignments move through the intermodal supply chain, EFM partner instances receive TransportationStatus from supply chain partner production systems in the form of either (a) legacy data formats or (b) direct web service calls to ReceiveTransportationStatus.
3. As EFM partner instances receive TransportationStatus data feeds from their production system counterparts, the data is transformed into UBL-based TransportationStatus messages and forwarded, via a web service call to the buyer’s EFM instance.
4. The buyer’s EFM instance receives the TransportationStatus message and saves it in the instance database.

The EFM package of software is designed to encourage Web service communication between supply chain partner enterprises while providing mechanisms for legacy file format capture with various transport methods (email, FTP, etc).

## Deployment Model

The EFM environment is a deployment of servers, platform software and application code supporting supply-chain partner transactions. The environment can provide either (a) proxy hosts for simulated supply chain partner interaction based on data feeds of actual partner supply chain data and/or (b) connections to actual partner enterprises.

As an example, a typical consignee may need to exchange messages with the following partners:

- One or more ocean carriers
- One or more drays
- One or more rails
- A customs broker

EFM can be installed directly in each partner's data center to serve as a common set of software for exchanging UBL-based messages. If it is not possible to install EFM software (for any reason), proxy servers can be deployed and hosted to stand in for the actual partner.

## Partner Instances

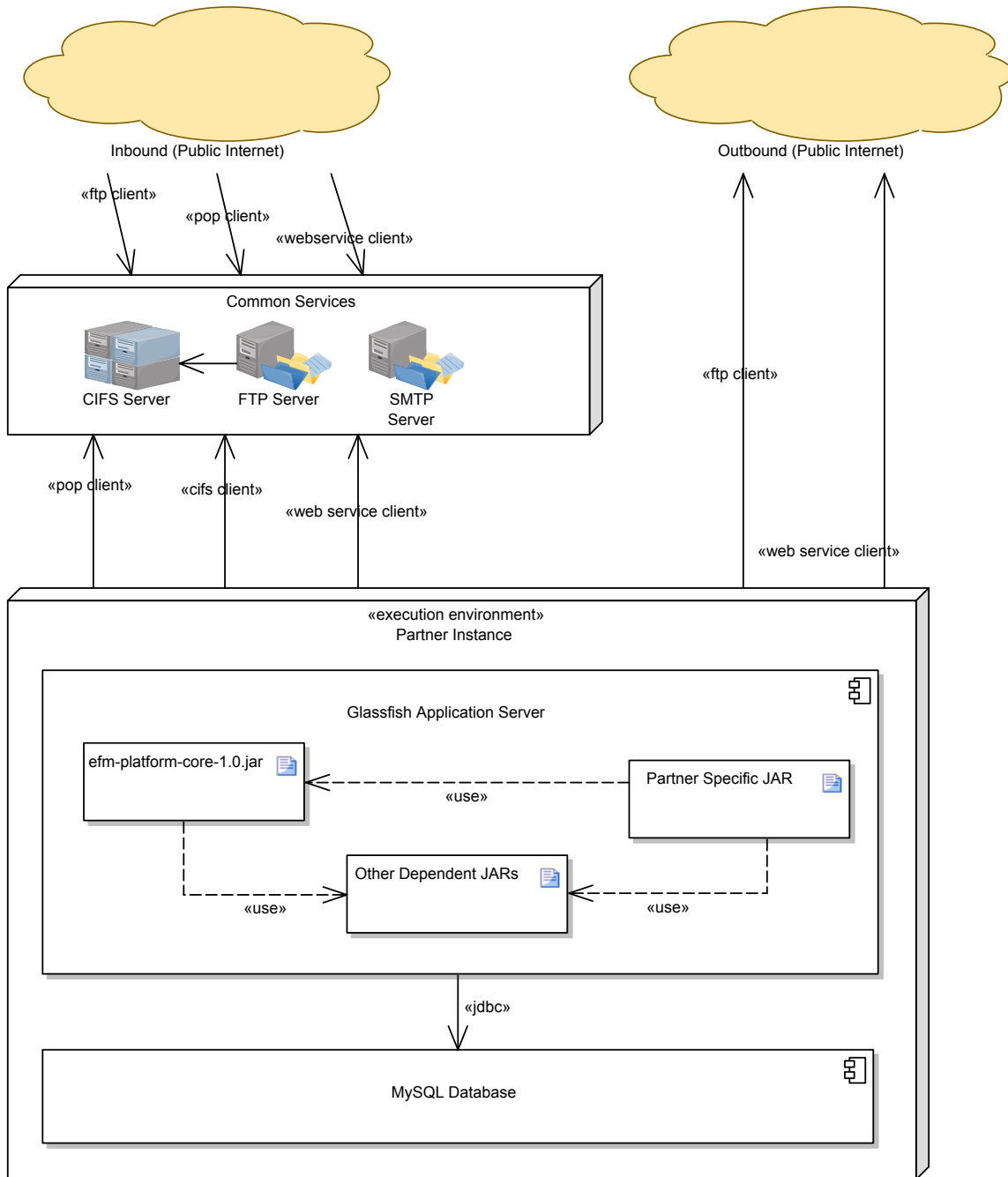
Whether hosted in an actual partner enterprise or via proxy, the EFM software stack looks similar. Figure 1 depicts a typical EFM partner instance. A fully fleshed out EFM environment would require one instance per supply chain partner.

## DEPLOYMENT REQUIREMENTS

The deployment environment must support the following requirements. These requirements are based on the Open Source products used to build EFM as well as the integration test environment.

It is not the intent of this document to prescribe specific server hardware quantities or capabilities but (a) what software is required (b) what OS is required and (c) some recommendations on configuration. In the specifications below, reference to a 'dedicated' server indicates a requirement to install and operate the application/functionality within its own dedicated stack, whether this is a standalone hardware device, or a VM with the necessary memory/disk. Those servers listed without the qualifier may be hosted in such a way to best utilize the hardware/resources available, keeping in mind, the requirements for this particular app/functionality.

Unless otherwise stated, the term *server* can be taken to mean either real or virtual.



**Figure 1. Prototypical EFM Partner Instance**

## Operating System

1. EFM has been tested with CentOS 5.2 and should be the operating system for all servers. Since CentOS 5.2 is re-packaged Redhat Enterprise Linux 5.2, Redhat will also suffice.
2. The following security parameters should be checked / set:

In `/etc/security/limits.conf`, set the following parameters:

```
*          -          core           0
*          soft      nofile        14335
*          hard      nofile        16383
```

3. The following kernel / boot parameters must be set:  
If the O/S is run in a VM under VMware Server / VMware ESX / VMware ESXi, the following kernel options should be appended in the boot loader configuration (i.e. `/etc/grub.conf`):

```
nosmp noapic nolapic acpi=off clocksource=acpi_pm elevator=noop
```

Also if the O/S is run in a VM, please make sure the guest has plenty of memory (above recommended number), and set the kernel swappiness setting to the smallest setting so it rarely attempts to swap memory pages to disk – unless it's absolutely necessary.

In `/etc/sysctl.conf`:

```
# Set Kernel's Tendency To Swap Memory Pages To Disk
vm.swappiness = 0
```

The following kernel parameters should be set, regardless, in `/etc/sysctl.conf` (per recommendation of the glassfish tuning guidelines for linux)

```
#####
# Glassfish Recommended Parameters #
#####
# Max File Descriptors Kernel Will Allocate
fs.file-max = 65535
net.core.rmem_max = 262143
net.core.rmem_default = 262143
net.ipv4.ip_local_port_range = 1024 65000
net.ipv4.tcp_rmem = 4096 131072 262143
net.ipv4.tcp_wmem = 4096 131072 262143
net.ipv4.tcp_sack = 0
net.ipv4.tcp_timestamps = 0
```

We recommend ipv6 is turned off by changing the `/etc/modprobe.conf` by appending the following:

```
# Turn of IPV6
#####
alias net-pf-10 off
alias ipv6 off
#####
```

## Servers

The number of servers needed depends on a number of factors including:

- The number of partners involved
- The mechanism used to implement the partner EFM deployment (are they real or hosted?)
- Scalability / transaction volume / performance requirements
- Security / enterprise boundaries

Generally, it's a good idea to host the database and application server on 2 separate servers (real or virtual), however this is not a requirement.

Ideally, each partner in the supply chain hosts their own instance of EFM, however, if you are providing *proxy EFM instances*, you will need servers for each partner being represented. Once again, it is recommended that each partner have its own server, but this is not a requirement.

Servers for other ancillary services may also be needed – for example FTP, file shares, and email servers.

## Software

1. Vsftpd, packaged with CentOS shall serve as the FTP server
2. CIFS, packaged with CentOS shall serve as the file sharing server
3. MySQL 5.0 shall serve as the database server
4. Glassfish v2 shall serve as the application server
5. An EFM package war file per partner instance. The war file contains the EFM platform core software, a partner-specific jar of software, and all 3<sup>rd</sup> party Open Source dependent jar files.
6. JRE 1.6, Update 11 shall be deployed on all servers

## Network Connections and Firewalls

Depending on the EFM environment for the particular situation being deployed, partners may be exchanging data using a variety of ports and protocols. Routers should be configured to open up (allow traffic) over the ports used by FTP, email, CIFS, https, etc.

## GLASSFISH AND MYSQL INSTALLATION

- Install Operating System (CentOS 5.2 / RedHat EL 5.2) -- standard build / image / installation.
- Do any steps to integrate authentication / standardization for your environment.
- Set up any local user accounts needed.
- Patch with latest updates, remove old kernel packages.
- Do kernel tuning, boot parameters, and security hardening as specified in the deployment document.
- If installing in a VMware virtual server, install the VMware tools.
- Install mysql packages - i.e., for CentOS you might issue the command:

```
yum install mysql mysql-server
```

MySQL distributions come with several default configurations that can be used. We prefer the default “medium” configuration be used as the base of configuration. EFM installation script will take care of the rest of the configuration options. To get the default medium configuration in place on CentOS linux, you'll need to execute the following comand:

```
cat /usr/share/mysql/my-medium.cnf > /etc/my.cnf
```

After this is done, set the server to start automatically:

```
chkconfig mysqld on  
service mysqld start
```

Make sure to set up a password for the root user:

```
mysqladmin -u root password <YOUR_PASSWORD>
```

- Reboot.
- Download the following:
  - sun-jdk-1.6.0-11.1.i586.rpm
  - apache-ant-1.7.0-0.noarch.rpm
  - glassfish-init-scripts-1.0-0.noarch.rpm
  - glassfish-installer-v2ur2-b04-linux.jar
  - mysql-connector-java-5.1.7.jar
- Place the SSL certificate / key pkcs12 bundle for this server in the same directory. Name the pkcs12 bundle <partner>-cert.p12. (Note: If this is not done at this point, this can be done manually at a later time)

### Installation Steps

The following are the general steps to set up glassfish and MySQL. They will almost assuredly need tuned to local installations. There are scripts in the EFM source code tree that are needed (and will need tuned). The following steps are the same basic steps as found in *install\_efm.sh*.

## 1. Install JDK

```
rpm -ivh sun-jdk-1.6.0-11.1-i586.rpm apache-ant-1.7.0-0-noarch.rpm \  
glassfish-init-scripts-1.0-0-noarch.rpm
```

## 2. Make sure MySQL is set to start automatically:

```
chkconfig mysqld on
```

## 3. Configure MySQL cnf file

```
chmod 755 ${WHENCE}/configure_my_cnf.pl  
cat /etc/my.cnf | ${WHENCE}/configure_my_cnf.pl > /etc/my.cnf.new  
cat /etc/my.cnf > /etc/my.cnf.orig  
cat /etc/my.cnf.new > /etc/my.cnf  
rm -f /etc/my.cnf.new
```

## 4. Start MySql

```
service mysqld restart
```

## 5. Install Glassfish

```
pushd /opt  
java -Xmx256m -jar ${WHENCE}/glassfish-installer-v2ur2-b04-linux.jar  
mv glassfish glassfish-v2ur2  
pushd glassfish-v2ur2  
ant -f setup-cluster.xml  
bin/asadmin delete-domain domain1  
chown -R glassfish:glassfish .  
find . -type f -print | xargs chmod g=u  
find . -type d -print | xargs chmod g=u,g+s  
  
# When prompted, enter password to be used for the administrators password.  
# leave master password blank  
su glassfish -c 'bin/asadmin create-domain --adminport 4848 --user administrator \  
--savelogin=true --profile=cluster ${efm.partner.name}'
```

## 6. Configure Glassfish init scripts

```
cat /etc/sysconfig/glassfish-config | sed \  
"s/REPLACE_PARTNER_NAME_HERE/${efm.partner.name}/" > \  
/etc/sysconfig/glassfish-config.new  
cat /etc/sysconfig/glassfish-config.new > /etc/sysconfig/glassfish-config  
rm -f /etc/sysconfig/glassfish-config.new  
chkconfig glassfish on
```

## 7. Install database library

```
cat ${WHENCE}/mysql-connector-java-5.1.7.jar > \  
domains/${efm.partner.name}/lib/ext/mysql-connector-java-5.1.7.jar
```

## 8. Start glassfish

```
service glassfish start
```

## 9. Create a directory for class-prefix items

```
mkdir -p domains/${efm.partner.name}/config/ext
```

## 10. Set glassfish jvm options

```
su glassfish -c 'bin/asadmin delete-jvm-options "\-Xmx512m"  
su glassfish -c 'bin/asadmin create-jvm-options "\-Xms768m:\-Xmx768m"  
su glassfish -c 'bin/asadmin set "server-config.java-config.classpath-prefix"=\  
  \${com.sun.aas.instanceRoot}/config/ext'  
popd  
popd
```

## 11. Import CA certificate

```
su glassfish -c 'keytool -importcert -keystore \  
${GLASSFISH_HOME}/domains/${efm.partner.name}/config/cacerts.jks -alias \  
org.efm_ca -file CA_org.efm-cert.pem -storepass changeit -noprompt -trustcacerts'  
unzip -j pkcs12import-20080923.zip  
export ASADMIN_SSL_OPTIONS=""
```

## 12. Install SSL certs

```
java -classpath ${WHENCE}/pkcs12import.jar com.sun.xml.wss.tools.PKCS12Import -file \  
${efm.partner.name}-cert.pl2 -alias ${efm.partner.name}_cert -keypass changeit \  
-pass changeit -keystore \  
${GLASSFISH_HOME}/domains/${efm.partner.name}/config/keystore.jks -storepass changeit  
export ASADMIN_SSL_OPTIONS="--secure --interactive=false"
```

```
pushd ${GLASSFISH_HOME}  
su glassfish -c 'bin/asadmin create-ssl --type http-listener --certname \  
${efm.partner.name}_cert admin-listener'  
su glassfish -c 'bin/asadmin set \  
"server-config.http-service.http-listener.admin-listener.security-enabled"=true'  
su glassfish -c 'bin/asadmin set \  
"server.http-service.http-listener.admin-listener.security-enabled"=true'  
su glassfish -c 'bin/asadmin set \  
"server-config.http-service.http-listener.http-listener-2.ssl.cert-\  
nickname"=${efm.partner.name}_cert'  
su glassfish -c 'bin/asadmin set \  
"server.http-service.http-listener.http-listener-2.ssl.cert-\  
nickname"=${efm.partner.name}_cert'  
su glassfish -c 'bin/asadmin set \  
"server-config.http-service.http-listener.http-listener-2.ssl.client-auth-\  
enabled"=true'  
su glassfish -c 'bin/asadmin set \  
"server.http-service.http-listener.http-listener-2.ssl.client-auth-enabled"=true'  
su glassfish -c 'bin/asadmin set \  
"server-config.security-service.auth-realm.certificate.property.assign-\  
groups"=EFMAuthorized'  
su glassfish -c 'bin/asadmin set \  
"server.security-service.auth-realm.certificate.property.assign-\  
groups"=EFMAuthorized'  
su glassfish -c 'bin/asadmin set \  
"server-config.http-service.property.accessLoggingEnabled"=true'  
su glassfish -c 'bin/asadmin set \  
"server.http-service.property.accessLoggingEnabled"=true'  
popd
```

## 13. Restart Glassfish

```
service glassfish restart
```

## 14. Run database deploy scripts

```
cat ${efm.partner.name}-db/bin/deploy_database.sh > \  
${efm.partner.name}-db/mysql/deploy_database.sh  
pushd ${efm.partner.name}-db/mysql  
chmod 755 deploy_database.sh  
./deploy_database.sh  
popd
```

## 15. Deploy the web app

```
pushd ${GLASSFISH_HOME}
su glassfish -c 'bin/asadmin add-resources --user administrator --host localhost \
--port 4848 ${ASADMIN_SSL_OPTIONS} ${WHENCE}/glassfish-resources.xml'
popd

if [[ -f efm.properties ]]
then
  su glassfish -c 'cat efm.properties > ~/efm.properties'
fi

su glassfish -c 'bin/asadmin deploy --user administrator --host localhost \
--port 4848 ${ASADMIN_SSL_OPTIONS} --name ${efm.partner.name} --contextroot \
${efm.partner.name} ${WHENCE}/${efm.partner.name}-web.war'
popd
```

After installation completes you may need to edit the `/home/glassfish/efm.properties` file in order to set up any special parameters for the partner, then restart the glassfish server.

## REFERENCES

- [1] Electronic Freight Management,
- [2] EFM Concept of Operations.
- [3] EFM Whitepaper
- [4] FIH Architecture
- [5] CEFM Concept of Operations
- [6] CEFM Detailed design document:
- [7] Demdaco Business Requirements
- [8] Demdaco Deployment Project Plan
- [9] EFM Platform Architecture
- [10] EFM Design