Agile
Version e6.1

ORACLE

Oracle® Agile Engineering Data Management

Architecture Guide for Agile e6.1.1

Part No. E15603-01

August 2009

Copyright and Trademarks

CONTENTS

Preface

The Oracle documentation set includes Adobe® Acrobat™ PDF files. The Oracle Technology Network (OTN) Web site (http://www.oracle.com/technology/documentation/agile.html) contains the latest versions of the Oracle Agile EDM PDF files. You can view or download these manuals from the Web site, or you can ask your Agile administrator if there is an Oracle Documentation folder available on your network from which you can access the documentation (PDF) files.

Note	To read the PDF files, you must use the free Adobe Acrobat Reader™ version 7.0 or later. This program can be downloaded from the Adobe Web site (http://www.adobe.com).

Note	Before calling Agile Support about a problem with an Oracle Agile EDM manual, please have the full part number ready, which is located on the title page.

TTY Access to Oracle Support Services

Oracle provides dedicated Text Telephone (TTY) access to Oracle Support Services within the United States of America 24 hours a day, 7 days a week. For TTY support, call 800.446.2398. Outside the United States, call +1.407.458.2479.

Readme

Any last-minute information about Oracle Agile EDM can be found in the Release Notes file on the Oracle Technology Network (OTN) Web site (http://www.oracle.com/technology/documentation/agile_eseries.html)

Agile Training Aids

Go to the Oracle University Web page (http://www.oracle.com/education/chooser/selectcountry_new.html) for more information on Agile Training offerings.

Accessibility of Code Examples in Documentation

Screen readers may not always correctly read the code examples in this document. The conventions for writing code require that closing braces should appear on an otherwise empty line; however, some screen readers may not always read a line of text that consists solely of a bracket or brace.

Accessibility of Links to External Web Sites in Documentation

This documentation may contain links to Web sites of other companies or organizations that Oracle does not own or control. Oracle neither evaluates nor makes any representations regarding the accessibility of these Web sites.

Chapter 1

Architecture Overview

The Agile e6 system architecture is based on an object-relational database (Oracle). A central component is the object-oriented repository. It is unique in that it stores all metadata which define the application with respect to:

- Object model
- User interface
- Business logic
  The business logic consists of e.g.:
  - Lifecycle definitions
  - Workflow processes
  - Consistency Checks
  - Automation Scripts

The repository ensures the separation of system description and physical structure. Because all metadata is stored in the database, deployment and upgrade processeses are simplified.

Three types of clients are available, serving the different needs of casual users and power users (see chapter Clients). With each client process launched, an application server process is started in parallel. The application server process interprets the repository and can thus dynamically reflect any changes applied to the metadata. User data (stored in the object-relational database) is accessed by the application server process (The connection to the database is realized with the corresponding protocol of the database management system. This is regarded as standard and not described any further in this document.). This ensures that the clients do not require a direct database connection.

This separation of services is called 3-Tier architecture, where the client is responsible for the presentation logic, the application server process is responsible for the business logic and the database server takes care of the physical storage of all data.

However, some responsibilities of the application server process have been assigned to dedicated services. These include the File Management Service “FMS” (responsible for managing physical files), and the Business Services (consisting of the Workflow Engine, responsible for executing business workflows, and the Permission Manager, responsible to manage all user-role assignments and the resulting permissions). The File Management Service and the Workflow are described in chapter Server Components.

View Server

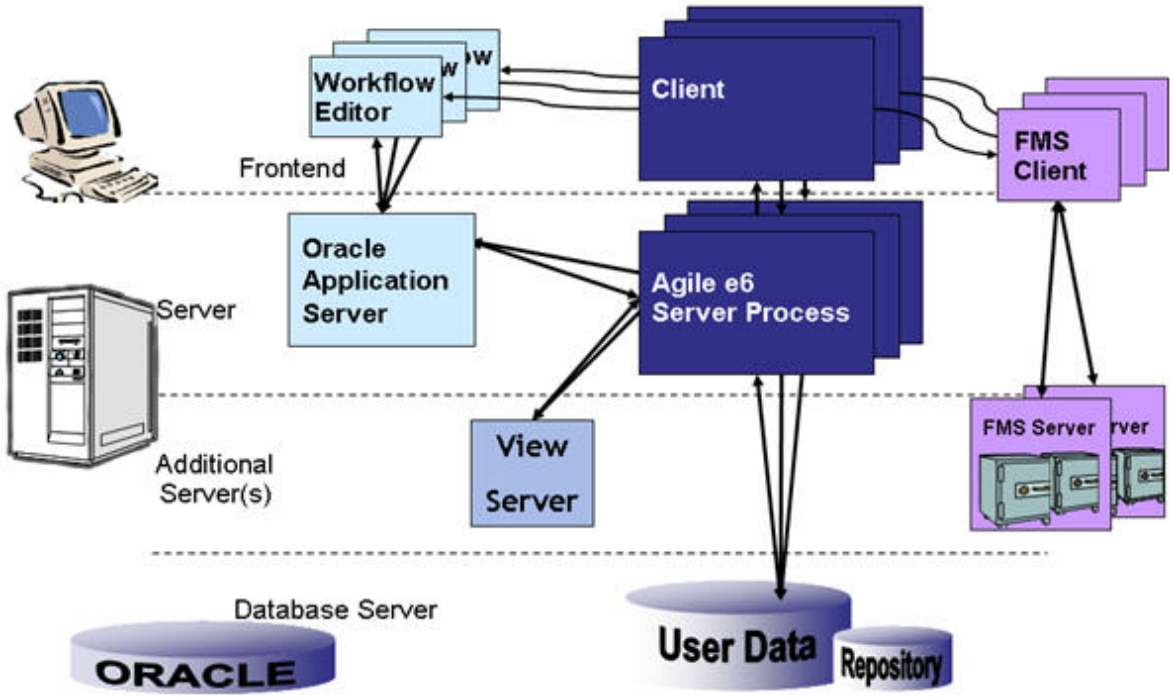
Note We recommend using the View Server as a separate server.

Contains the Oracle AutoVue Server and the VueLink service, which transfer and proceed files from the File Server to the Java Client. It displays different types of files in the Java Client, e.g. 2D/3D CAD files or Microsoft Office documents.

These separate services can reside on the same server where the application server processes are

executed (this is recommended for the Workflow Engine and the Permission Manager) or can reside on any other computer in the network (this is especially true for the FMS server). Due to this additional separation of services, Agile e6 actually features a distributed architecture.

The following picture gives an overview of the main services which are used in the architecture of Agile e6:



For reasons of simplicity some services (e.g. External Mail client) are not displayed

The following chapters explain the main services and their communication in more detail and show the communication types that affect the configuration of a firewall.

The entire communication is based on TCP/IP. Only unprivileged ports (above 1024) are used. Once a TCP/IP connection is established the port will not be changed dynamically.

Lines of communication, that are local to the Client, or the Application Server, or between the Application Server and the database server (or additional servers), are not described here since they are not relevant for the configuration of a firewall.

Chapter 2

Server Components

File Management Service

Files are stored on any server in the network under control of the File Management Service (FMS). The File Management Service manages documents (referred to as “Files”) in vaults, thus implementing the “check-in” and “check-out” functionality provided by the Document Management System in Agile e6.

The File Management Service consists of an FMS Client and a FMS Server. An FMS client communicates with the corresponding FMS server to check-in and check-out files.

Detailed descriptions about the access from the different clients can be found further on in this document in the respective client communication chapters. In case of the Java Client, the FMS Client is embedded and does not need to be installed separately. In case of the Web Client, the FMS Client is executed on the Web Server and does not require any installation on the front-end.

The FMS Server is installed on one or more server computers. Each FMS Server can manage one or more vaults.

- If the FMS Server is installed on a Windows platform, this server must be NTFS based. FAT does not work.
- The vaults managed by a FMS Server have to be created on local hard discs of the computer where the corresponding FMS Server is running or on a SAN. The SAN has to be configured in a way that IO from the file and the database server are separated (separated IO channel).
- The FMS Client communicates with the Agile e6 application server process using sockets.
- The FMS Server and the FMS Client communicate with Remote Procedure Calls (RPC) and sockets. The FMS Server does not communicate directly with the Agile e6 application server process.

In a very simple configuration, there is only one FMS Server with a single vault (e.g. when Agile e6 is installed on a single workgroup server or on a laptop).

In large installations of Agile e6, there may be FMS Servers running on several computers, each FMS Server managing multiple vaults.

Installations with remote users that connect to an Agile e6 Server through a Wide Area Network (WAN), e.g. an external office that is connected to the headquarters, would usually be limited when accessing files by a small bandwidth. To improve performance in such a configuration, a FMS Server and one or more vaults can also be installed at the remote location (even though the Agile e6 server and the database are running at the headquarters). Files can be checked-in and checked-out into these vaults by any FMS Client that is able to communicate with this FMS Server.

Web File Services

Introduction

In an internet environment it is necessary to provide a file storage access via HTTP/HTTPS. The Web File Services are providing a scaleable access to the file vaults which are managed by the Core File Server.

Additionally, the Internet access to the PLM system needs a high security solution for the full environment. The usage of the HTTPS protocol is only the first step. The system administrators are installing firewalls to protect the internal network. These firewalls are blocking the most ports so that only well knows ports are available to communicate through the firewall. The best case for an administrator is that only one port is necessary.

This article covers the implementation of a HTTPS environment for the Web-Client and the Web File Service. Furthermore, the article describes the high security enhancements and shows how they are could be implemented on the customer side.

System Overview

Required hardware: All supported platforms for PLM

Required Software: TOMCAT web server, JSSE/JCE framework

Required Knowhow: Administration of a TOMCAT web server

Enable Secure Socket Layer Support (HTTPS)

The web servers are support the HTTPS protocol, which encrypts the data and let the caller know who is the server where I get the data.

This section describes the steps, which are necessary to setup a HTTPS environment.

Server Certificate

For testing purposes, you can generate your own certificate (This works not for services like the Web File Service, but you can test the Web-Client without file access).

For production use, you need to obtain a Server Certificate from a trusted source, such as Verisign or Thawte.

If you have more than one domain name you wish to secure, you should get multiple certificates.

Digital certificates are host and domain name specific, so you will need as many certificates as you have domain names. For additional informations please see the TOMCAT web page or the information provided by the Certificate Authorities.

Installation

The following describe the steps needed to configure Tomcat SSL. In all of the sample code, the symbol <plm_root> is defined as the PLM installation directory.

Edit server_web.xml

Edit <plm_root>/epclt/webplm/config/server_web.xml. Add or uncomment the following sample XML snippet:

```
<Connector className="org.apache.tomcat.service.PoolTcpConnector">
    <Parameter name="handler"
value="org.apache.tomcat.service.http.HttpConnectionHandler"/>
    <Parameter name="port" value="8443"/>
    <Parameter name="socketFactory"
value="org.apache.tomcat.net.SSLSocketFactory"/>
    <Parameter name="keystore"
value="<plm_root>/ext/tomcat/conf/keystore" />
    <Parameter name="keypass" value="changeit"/>
</Connector>
```

In this example the keystore is the file <plm_root>/ext/tomcat/conf/keystore with the password change it.

Edit java.security (Java 1.3.x only)

Edit the file <plm_root>/ext/jre/<machine_platform>/lib/security/java.security and add the following line:

```
security.provider.2=com.sun.net.ssl.internal.ssl.Provider
```

Note There may already be a security.provider.2 entry. Adjust the above line if needed.

Generate Certificate (for Testing only)

This step is for testing purposes only. Skip to the next step for production systems.

Generate a certificate for Test purposes only. Run the Java keytool utility that is included with every Eigner PLM installation.

Replace <keystore> with the keystore attribute value defined in server_web.xml above. Use the password defined in the keypass attribute, and you do not need to sign the certificate.

```
cd <plm_root>/ext/jre/<machine_platform>/bin
./keytool -genkey -alias tomcat -keyalg RSA –keystore <keystore>
```

In this example a digital certificate is generated with the alias tomcat using RSA algorithm and stored in the file <keystore>. RSA is required for Netscape and IE browsers.

Import Certificate

This step required for production systems after you have obtained a digital certificate from a trusted source. Please see the information at the Certificat Authority how to create a certificate request.

Use the keytool utility to import a certificate obtained from a trusted source. The following example imports a certificate file CERT.pem to the file <keystore>. Use the keystore attribute from server_web.xml defined above.

```
cd <plm_root>/ext/jre/<machine_platform>/bin
./keytool -import -v -trustcacerts -alias tomcat -file CERT.pem –
keystore <keystore>
```

Enable the HTTPS protocol

Then you need to add the following tomcat startup option (TOMCAT_OPTS) to support the HTTPS protocol for the URL class.

```
TOMCAT_OPTS = -
Djava.protocol.handler.pkgs=com.sun.net.ssl.internal.www.protocol
```

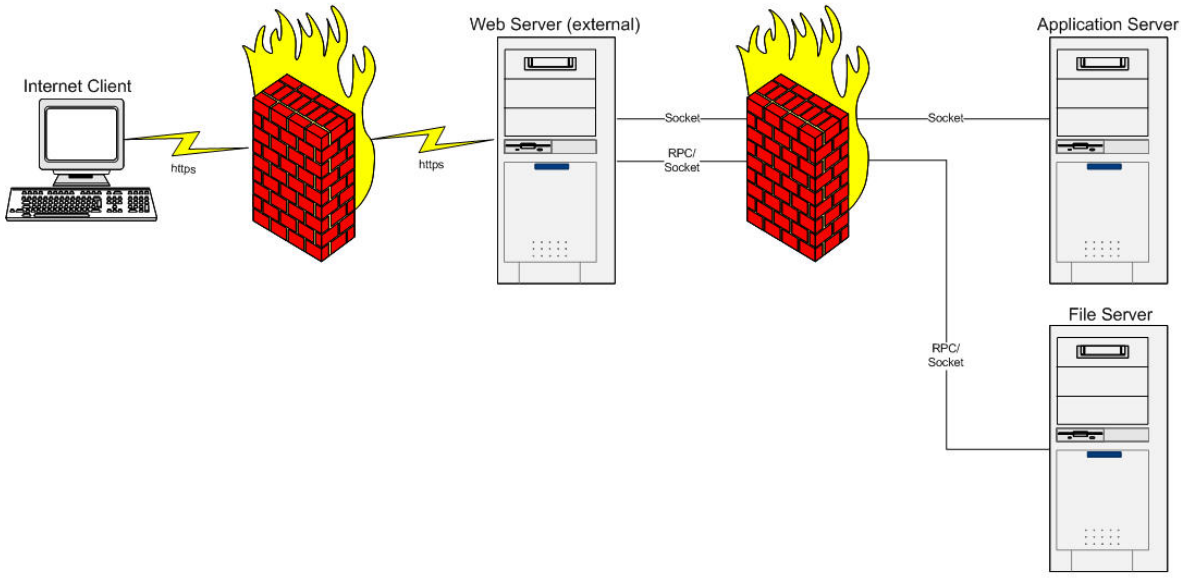
Security Concepts

The PLM system should provide a solution to access files in a medium secure environment and in high-end security environments. The files are stored in a vault, which is managed by a file server. This file server provides the files for every application, which has access to the Meta data. The Meta data is stored in the PLM system.

In the web environment the Web Client has the access to the PLM system via an ECI connection. The Web Client provides the access to the file access operations, which are executed by the Web File Service. The Meta data is only available within the web server; an access from the Internet is possible, but the system uses encrypted tickets to grand the access to the Meta data.

Medium Secure Environment

The medium secure environment has one or more firewalls and could have a standard proxy before the external web server. The first firewall grants access for one port only. The possible second firewall blocks the most ports, but let pass some dedicated ports.



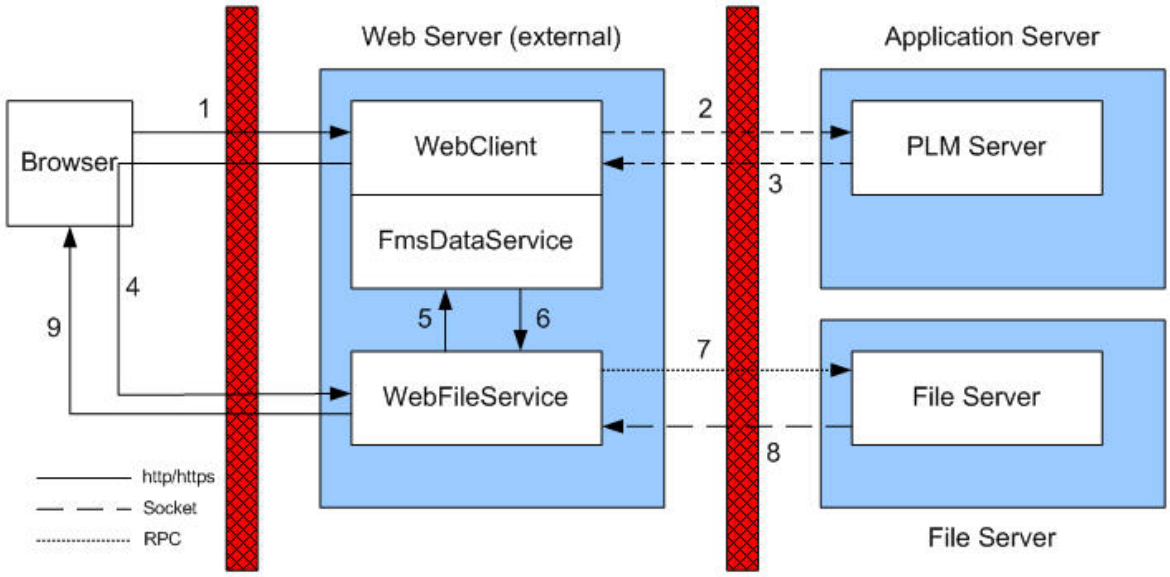
The first firewall let pass the HTTP or HTTPS port only to access the web server all other ports are blocked. The second firewall let pass the ports needed by the ECI, the RPC port and the file server ports, which are necessary to exchange the file data.

The following table shows the necessary ports:

Module	Ports	configurable	Description
ECI	16067	yes	PLM Java Daemon
ECI	5100-5150	no	ECI communication socket
FMS	111 (RPC)	no	RPC port mapper
FMS	52517-53517	no	FMS communication socket, the port range depends on the max. Number of concurrent users

Communication path during file viewing

This section describes the communication between the several components in a medium and high-end security environment.



Software components

- **Browser** The browser is used to access the web site on the Web Server
- **Web Client** The Web Client is a web service, which gains access to the PLM system
- **FMS Data Service** The FMS Data Service is an internal service, which provides the Meta data for the file access
- **Web File Service** The Web File Service provides the file access for the user
- **PLM Server** The PLM Server represents the PLM system
- **File Server** The File Server manages the files within the PLM system

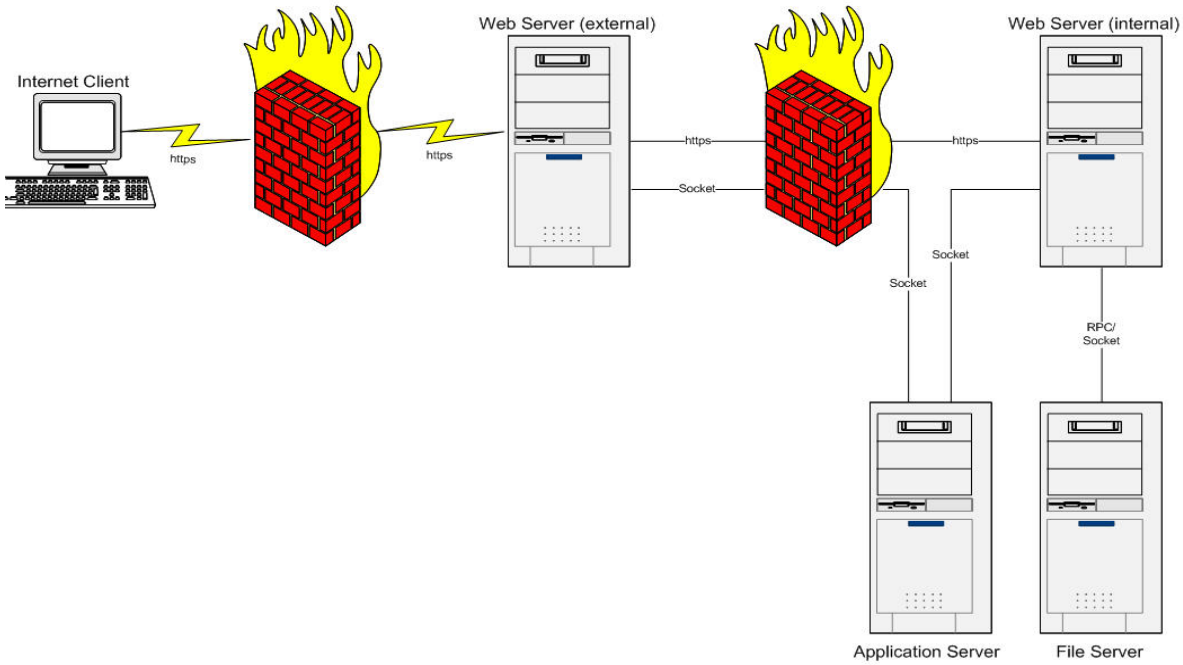
Communication steps:

1. The user pushes the viewing button to view a file

2. The Web Client calls the xfile userexit over ECI
3. The PLM Server calls the FMS callable over ECI (The Meta data will be stored in the Cache of the FMS Data Service)
4. The Web Client responds a redirection URL to the browser and the browser sends a GET request to the Web File Service (The request contains a access ticket)
5. The Web File Service decrypts the access ticket and sends a signed request envelope to the FMS Data Service to gain the Meta data
6. The FMS Data Service responds with a signed envelope, which contains the Meta data
7. The Web File Service calls the File Server via RPC by using the FMS client library. The RPC return packet contains the socket for the file transfer
8. The File Server creates a new thread for the file transfer and sends the file via socket to the calling Web File Service.
9. The Web File Service sends the receiving data without caching as respond of the redirection GET request to the browser.

High-end Security Environment

The high-end security environment has two firewalls or more firewalls and could have a standard proxy before the external web server. The firewall, which protects the external web server let pass one port only. This port is a HTTP or HTTPS port to access the web server. At the current state of development the Java ECI does not support the HTTP protocol, so that the ECI ports are necessary (see the table above).

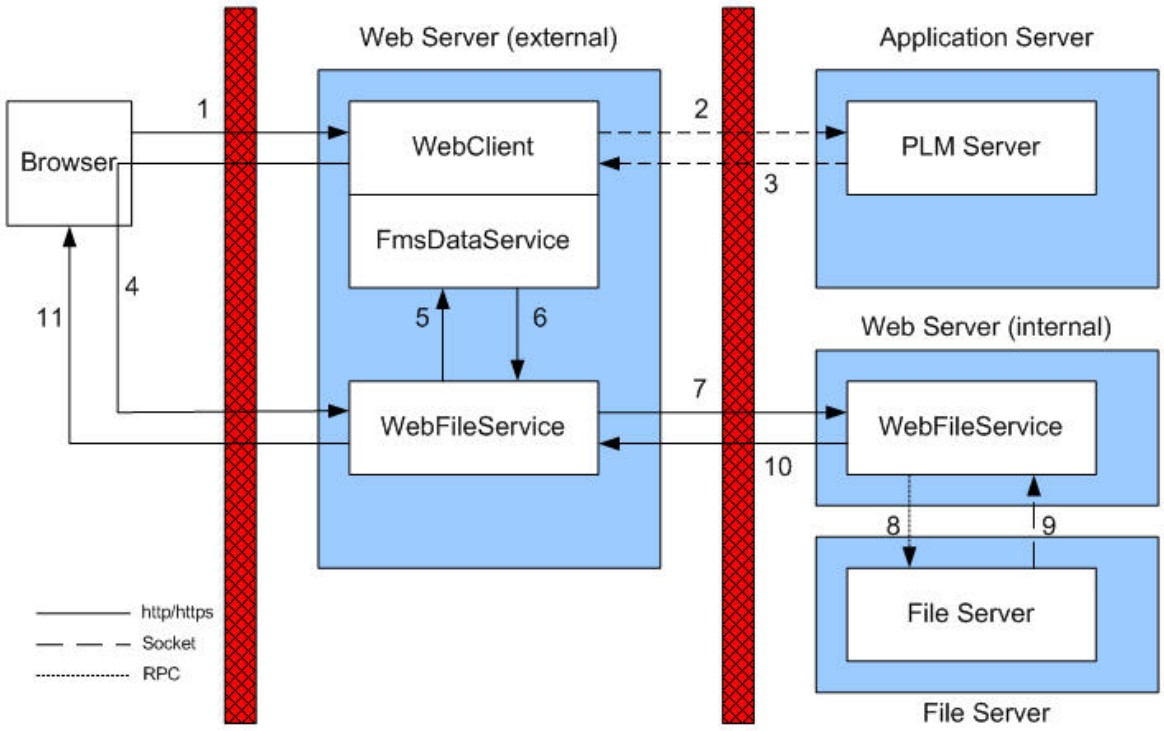


The most customers are having a Intranet web server, which provides a web access to the employees. The high-end solution uses this internal web server to provide the access to the files for Internet users. The Internet Web File Service works as a proxy, which sends the request to the

internal Web File Service. The internal Web File Service requests the file from the file server and sends the file data to the external Web File Service. The external Web File Service sends the file to the Internet user. Any Web File Service does not cache the file data, the incoming file data is sent as response to the user or calling Web File Service.

Note This environment is supported for file viewing only!

Communication path during file viewing



Software components

- **Browser** The browser is used to access the web site on the Web Server
- **Web Client** The Web Client is a web service, which gains access to the PLM system
- **FMS Data Service** The FMS Data Service is an internal service, which provides the Meta data for the file access
- **Web File Service** The Web File Service provides the file access for the user
- **PLM Server** The PLM Server represents the PLM system
- **File Server** The File Server manages the files within the PLM system

Communication steps:

1. The user pushes the viewing button to view a file
2. The Web Client calls the xfile userexit over ECI (The PLM Server checks if a proxy

configuration is available for the Web Client site and adds the proxy data to the Meta data)

3. The PLM Server calls the FMS callable over ECI (The Meta data will be stored in the Cache of the FMS Data Service)
4. The Web Client responds a redirection URL to the browser and the browser sends a GET request to the Web File Service (The request contains a access ticket)
5. The Web File Service decrypts the access ticket and sends a signed request envelope to the FMS Data Service to gain the Meta data
6. The FMS Data Service responds with a signed envelope, which contains the Meta data
7. The Web File Service generates a POST request to the Web File Service, which is running on the internal Web Server to get the file. The POST request contains an encrypted envelope with the Meta data.
8. The Web File Service calls the File Server via RPC by using the FMS client library. The RPC return packet contains the socket for the file transfer
9. The File Server creates a new thread for the file transfer and sends the file via socket to the calling Web File Service.
10. The Web File Service sends the incoming file data without caching to the calling proxy Web File Service.
11. The (Proxy) Web File Service sends the receiving data without caching as respond of the redirection GET request to the browser.

Configuration

This chapter explains the possibilities to configure the Web Proxy Service of the web file management system.

Configuration parameter

The web proxy supports a main switch to activate or deactivate the proxy mechanism. The main switch is accessible via the “Configuration parameter” mask (Manager->System Configuration->Other Parameters).

```
EDB-FMS-PROXY     ON/OFF
```

The default of this configuration parameter is “OFF”, so you have to insert this configuration parameter to activate the proxy mechanism.

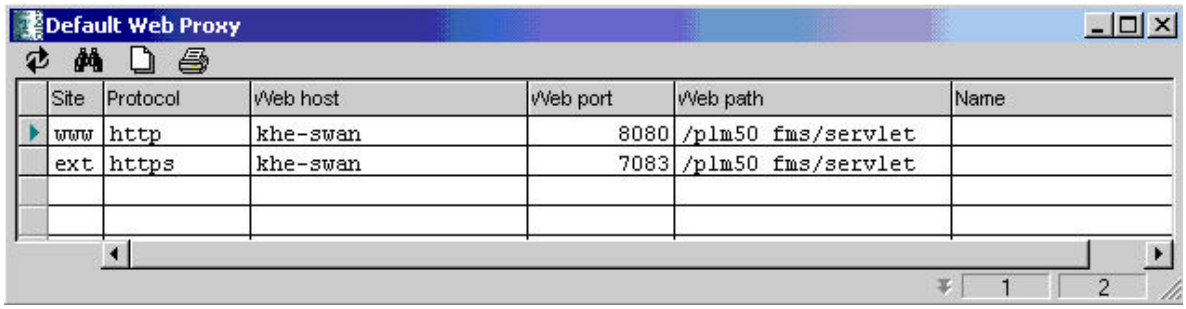
Define a Web Proxy Service

In the case of a high-end security environment the Web File Service on the external Web Server works as a proxy to the internal Web File Service. The definition of a Web Proxy needs two steps.

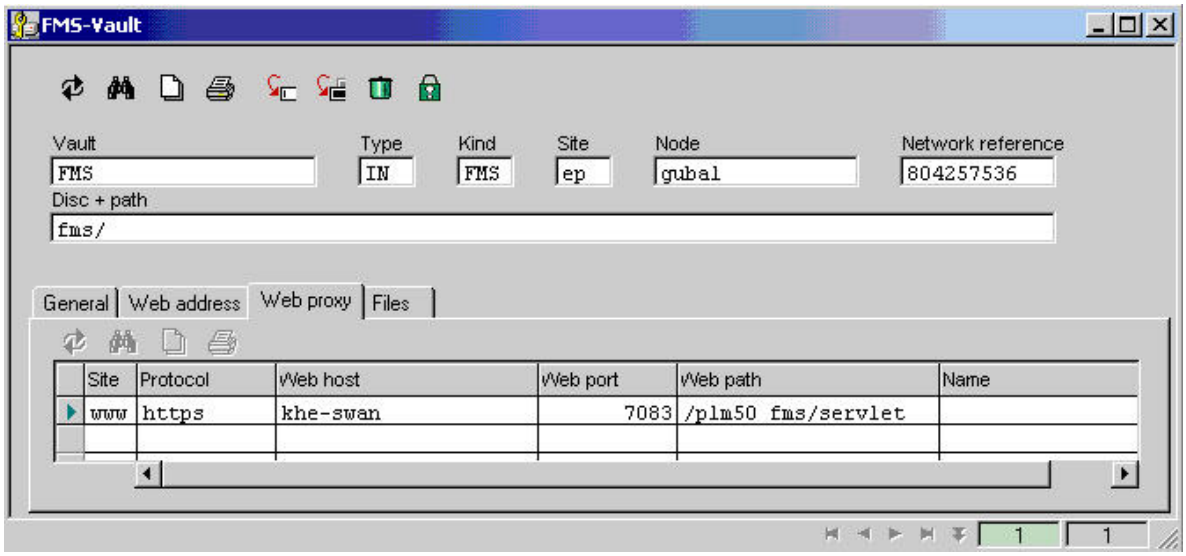
1. Define a default proxy for a site or a specialized proxy for a vault for the site
2. Set the site information in the configuration file of the Web Client

Definition of a proxy

You can define a set of default proxy entries for several sites, by adding the information of the external Web Server with the path to the (Proxy) Web File Service.



Additional you can define a set of proxy entries for several sites, by adding the information of the external Web Server with the path to the (Proxy) Web File Service. If a vault has no proxy information the default proxy information is used.



Site information of the Web Client

Each external Web Client should have a unique site information, so that the PLM Server could decide if a (Proxy) Web File Service is necessary. You can add this information in the webplm.properties file of the Web Client, which is located in the <webclient_root>/config directory.

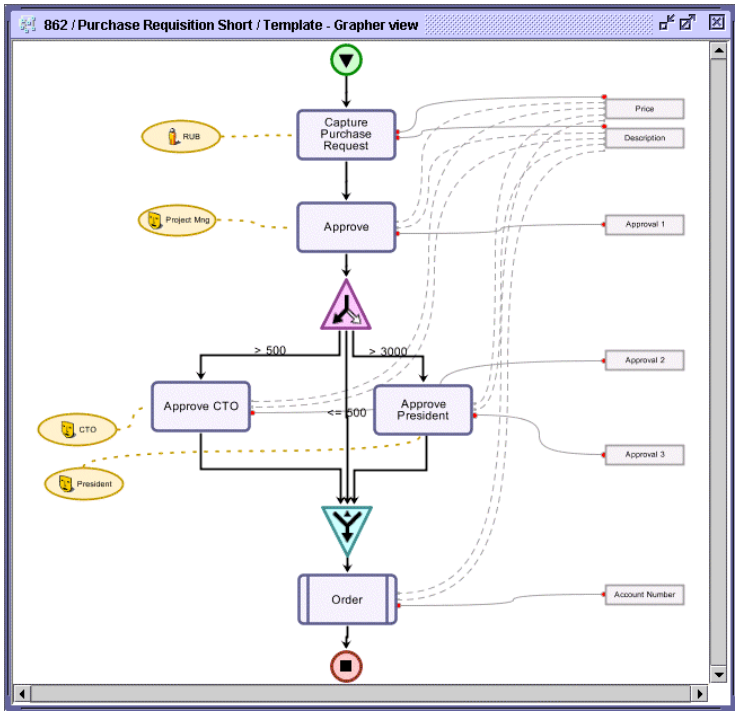
The following line sets the site “www” for this Web Cient:

```
axalant.env.EP_PROXY_SITE=www
```

Workflow Editor

Agile e6 includes a workflow solution that allows automating business processes.

The definition of a workflow process consists of the activities, the resources responsible for the execution of the activities, and the routing. Workflow processes are graphically defined with the Workflow Editor, as depicted in the following figure:

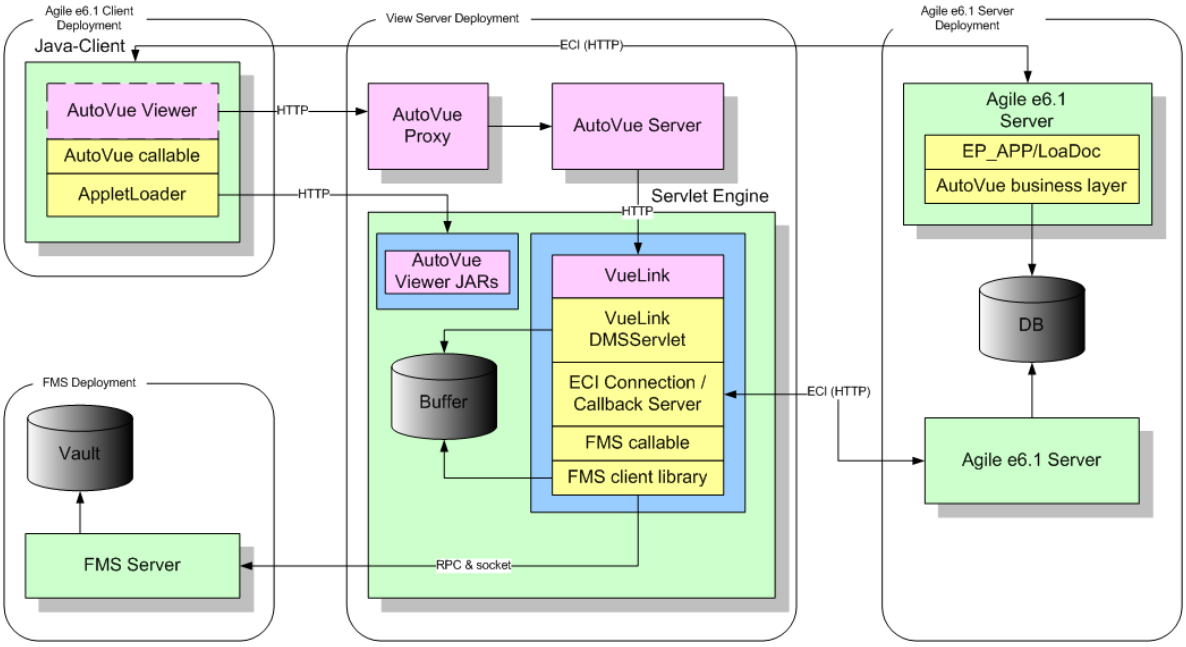


The Workflow Editor communicates with the Business Services, which include the Workflow Engine and the Permission Manager. All data related to a workflow process are stored in the database, thus ensuring the integrity of the system.

The Workflow Editor can be launched from the Windows and the Java Client. To start the Workflow Editor from Windows, an extra Java process is started.

View Server (AutoVue)

The following picture shows the communication between the involved components.



Chapter 3

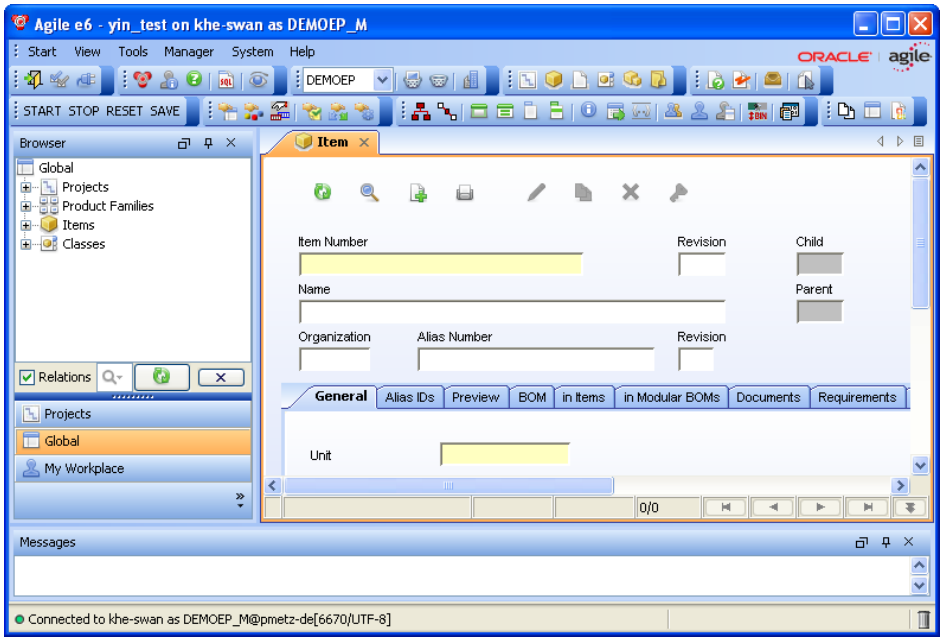
Clients

With Agile e6, three types of clients are available, serving the different needs of casual users and power users.

- Java Client
- Web Client
- Windows Client

Java Client

The Java Client is the standard client for Agile e6. It enables access to all functions of Agile e6 (depending on the client platform). The Java Client is suitable for Customizers (except for the mask generation – this can only be done in the Windows Client) and offers only limited support for Administrators.



In combination with Sun's Java Web Start technology (see http://java.sun.com/products/javawebstart/), the deployment of the Java Client can be reduced dramatically.

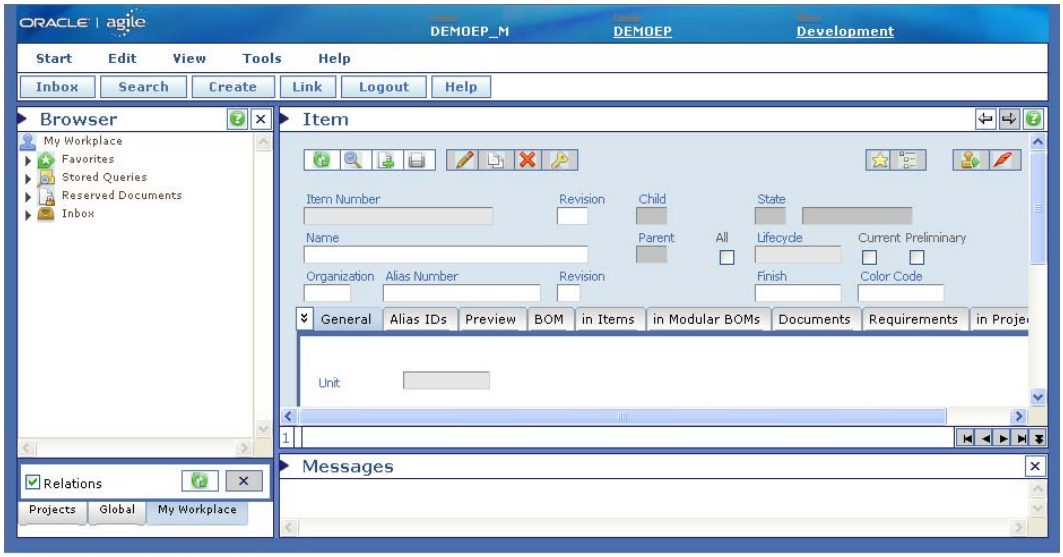
The Java Client does fully support M-CAD, E-CAD, EDA integrations (where the corresponding system itself is available on corresponding front-ends) and most other types of integrations (e.g. SAP Link).

Microsoft Office files checked in via the Office Suite, can be accessed from the Java and Windows Client. A bi-directional exchange of file properties as well as sophisticated features, e.g. to process links within the files, will only be available with the Java and Windows Client.

The user interface of the Java Client is dynamically defined by the metadata in the repository.

Web Client

The Web Client can be used by casual users on all front-ends. It requires either Microsoft Internet Explorer or Netscape/Mozilla. It enables access to most functions of Agile e6 (which are not Windows-specific). The Web Client is not suitable for Customizers and Administrators.



The Web Client uses HTML, respectively DHTML (Dynamic HTML, a combination of HTML and JavaScript). Because the Web Client is a Web Presentation Service (in this document we refer to it as the Web Client) on the web server, it does not require a local installation that can be accessed by a web browser. It is therefore the ideal client for zero-deployment costs.

The most important limitations compared to the Java Client include:

- Slightly different concept of the integrated Explorer window (similar functionality can be achieved, but the customization is partially separate).
- No support for significant fields in lists (that means all columns in a list will scroll horizontally).
- No support for drag&drop from/to the desktop.
- Modal dialogs may not be supported completely (however the cases that are not supported have been reduced dramatically, allowing the Java Client to be used in most scenarios).

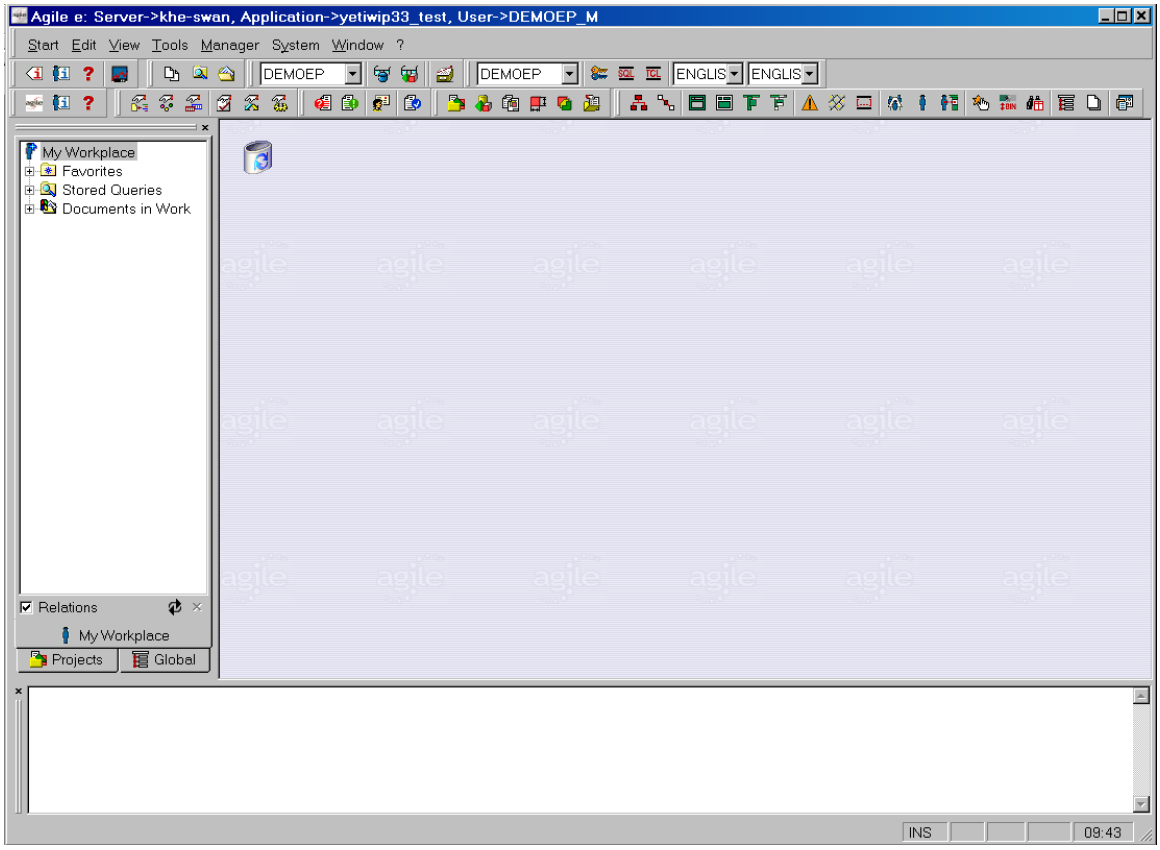
The Web Client does not support M-CAD, E-CAD, EDA integrations, and the Office Suite features. However, the Web Client can be used with integrations that are processed on the server, e.g. the SAP Link.

The user interface of the Web Client is dynamically defined by the metadata in the repository, except for some specific components, including the integrated Browser windows, the Search Panel

and the Wizards.

Windows Client

The Windows Client enables access to all functions of Agile e6. Administrators and Customizers can use the Windows Client.



The Windows Client allows users to drag&drop between Agile e6 and other Windows applications (e.g. Windows Explorer, Microsoft Excel, etc.).

The Windows Client fully supports available M-CAD, E-CAD, EDA integrations, and most other types of integrations (e.g. SAP Link).

Microsoft Office files checked in via the Office Suite, can be accessed from the Java and Windows Client. A bi-directional exchange of file properties as well as sophisticated features, e.g. to process links within the files, will only be available with the Java and Windows Client.

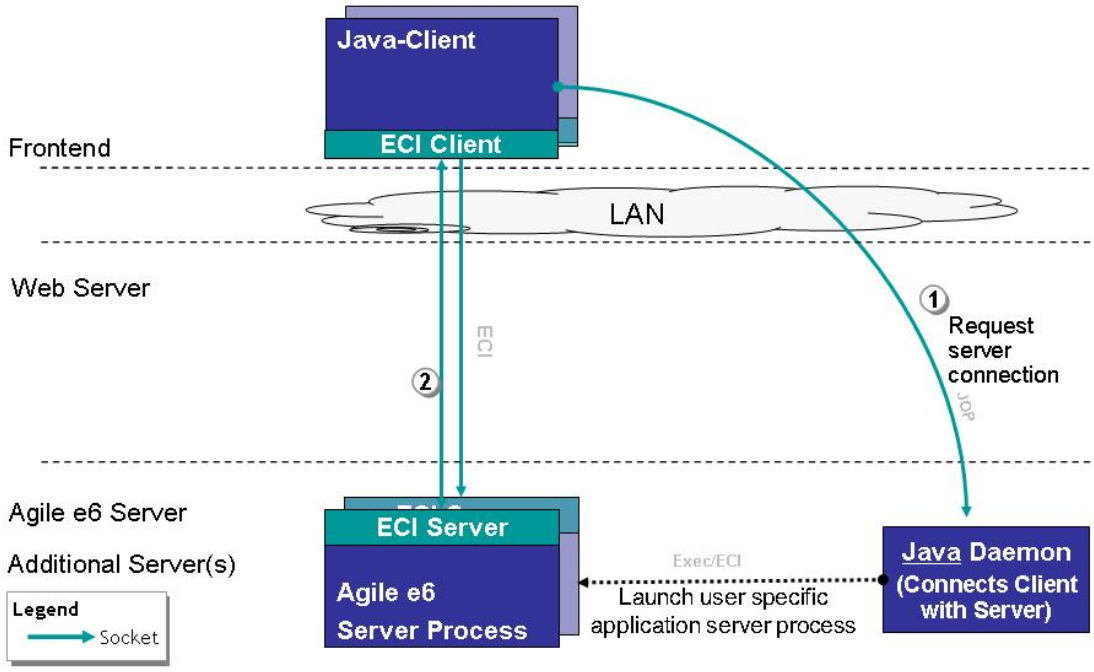
The user interface of the Windows Client is dynamically defined by the metadata in the repository.

Chapter 4

Java Client Communication

Lines of Communication when Launching the Java Client

The lines of communication when launching the Java Client are depicted in the following figure:



The following steps are executed when the user launches the Java Client:

- The Java Client connects to the Java Daemon, requesting to launch a user specific Application Server Process and to return the connection information for this process (communication line (1)).
- The Java Client connects to the given Application Server Process and starts communication (communication line (2)).

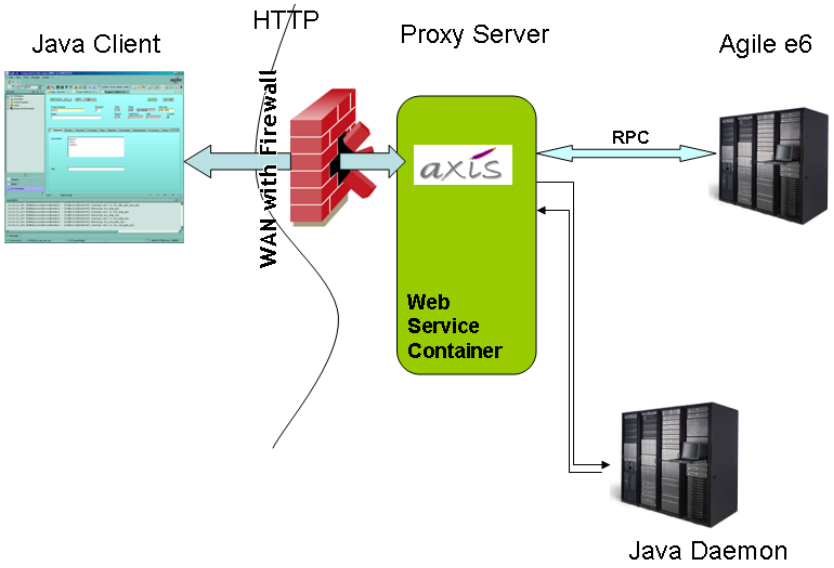
The communication line between the Java Client and the Application Server Process utilizes the ECI (External Communication Interface) protocol – the same that is used by e.g. an M-CAD system to communicate with an Agile e6 client on the front-end.

The following table contains the details about the relevant lines of communication:

Line of Communication	Type	Port or Range of Ports
(1)	Application specific protocol based on Sockets	Port 16077 by default (can be configured during installation)
(2)	Application specific protocol based on Sockets	One unprivileged port per Application Server Process from range as configured for Java Daemon (5000 to 5999 by default)

Firewall Friendliness

The new Firewall Friendliness allows the communication to Agile e6 application server through firewalls (via http). This is accomplished by using ECI via Web Service with SOAP communicating through http.



The Java client sends an HTTP requests to the proxy server. First, the proxy server connects with the Java daemon to get the address for the Agile e6 server. Then it connects itself with the Agile e6 server for further calls from the Java client.

Note The proxy server is built from a web service container (e.g. oc4j) and web services.

Note Using the new Firewall Friendliness can have an influence on the performance of the Java client.

Lines of Communication with HTTP Support

The connection between JavaClient and Agile e6 through HTTP occurs through a component called PLMAPI Proxy which provides the HTTP communication capability.

The JavaClient basically makes two connections:

3. With JavaDaemon

   To bootstrap the actual communication

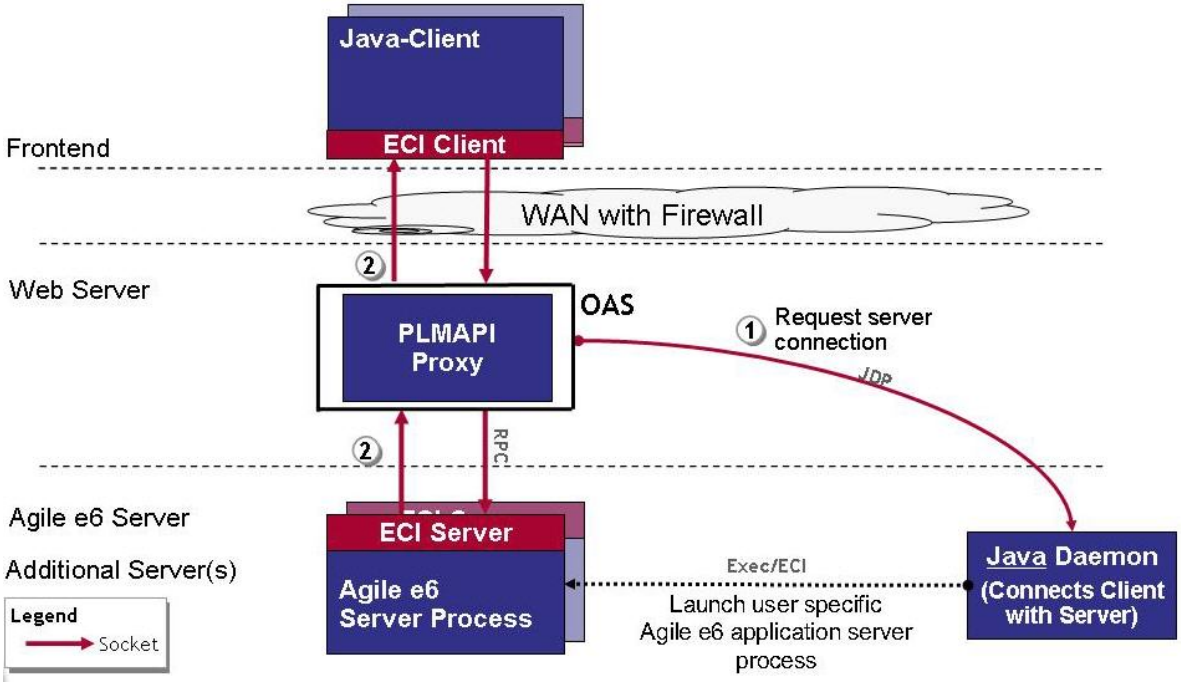
4. Connects to Agile e6

   To start the actually communication with Agile e6 server and then closes the connection with JavaDaemon.

The connections are socket based, and use a proprietery protocol. In the WAN scenario this is not possible as there is almost always a firewall which restricts the communication. But as a standard, the firewall does allow HTTP communication through standard port number 80 for HTTP, and 443 for HTTPS.

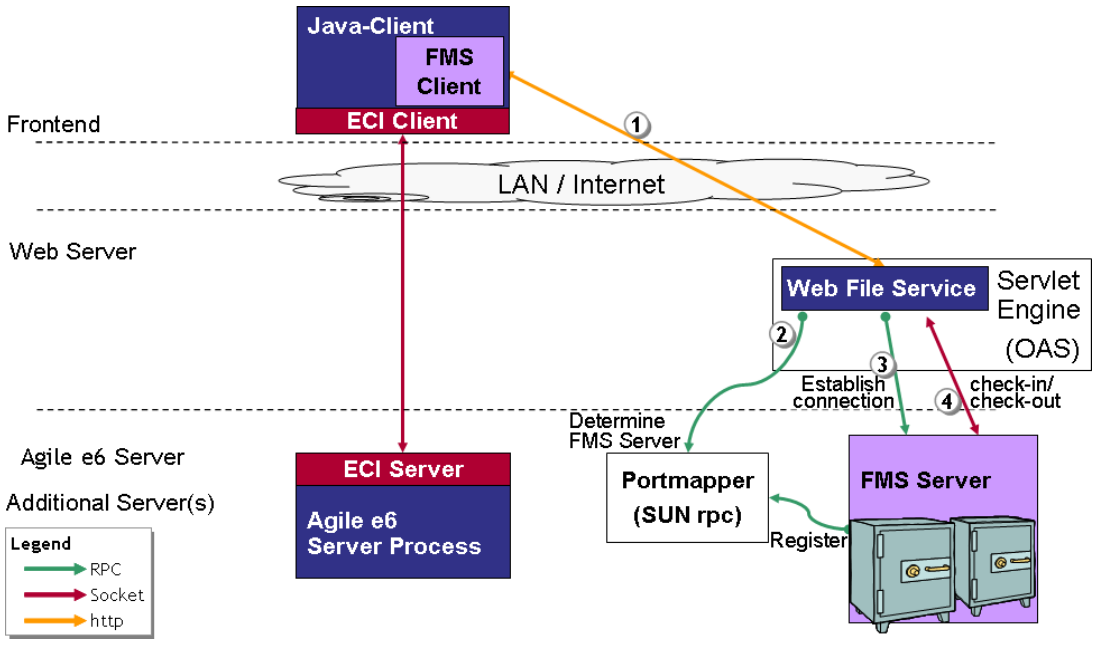
In case of a HTTP communication, there is a web server sitting between Java client and Agile server components (Agile e6 server and JavaDaemon) inside the firewall. The PLMAPI Proxy should be deployed in this webserver.

The Java client connects to the PLMAPI proxy and the PLMAPI bootstraps the connection for the JavaClient by first connecting to Java Daemon, and then to Agile e6 server by using the credentials sent from Java client. Once the PLMAPI has finished the connection bootstrapping, the JavaClient can communicate through PLMAPI Proxy with Agile E6 server.



Lines of Communication in Case of File Access (using FMS)

The communication between the Java Client, the Agile e6 application server process, the Web File Service and the FMS Server is depicted in the following figure:



The following steps are executed when the user performs a check-in or a check-out operation in the Java Client:

- The embedded FMS Client connects to a Web File Service, which is hosted by a servlet engine (communication line (1)). The address of the Web File Service is taken from the configuration of the corresponding vault in the Agile e6 database. For a check-out operation, the FMS Client sends the corresponding request parameter to the Web File Server. For a check-in operation, the FMS Client sends the corresponding file to the Web File Server.
- With the information about the FMS Server, the Web File Service connects to the Portmapper to determine the port of the corresponding FMS Server (communication line (2)).
- Now the Web File Service can request the FMS Server to create an individual thread and return the connection parameter (communication line (3)).
- The file data is transmitted between the Web File Service and the FMS Server thread (communication line (4)). The Web File Service itself returns the call to the embedded FMS Client.

Files are not stored temporarily, but transferred directly between the front-end and the FMS Server.

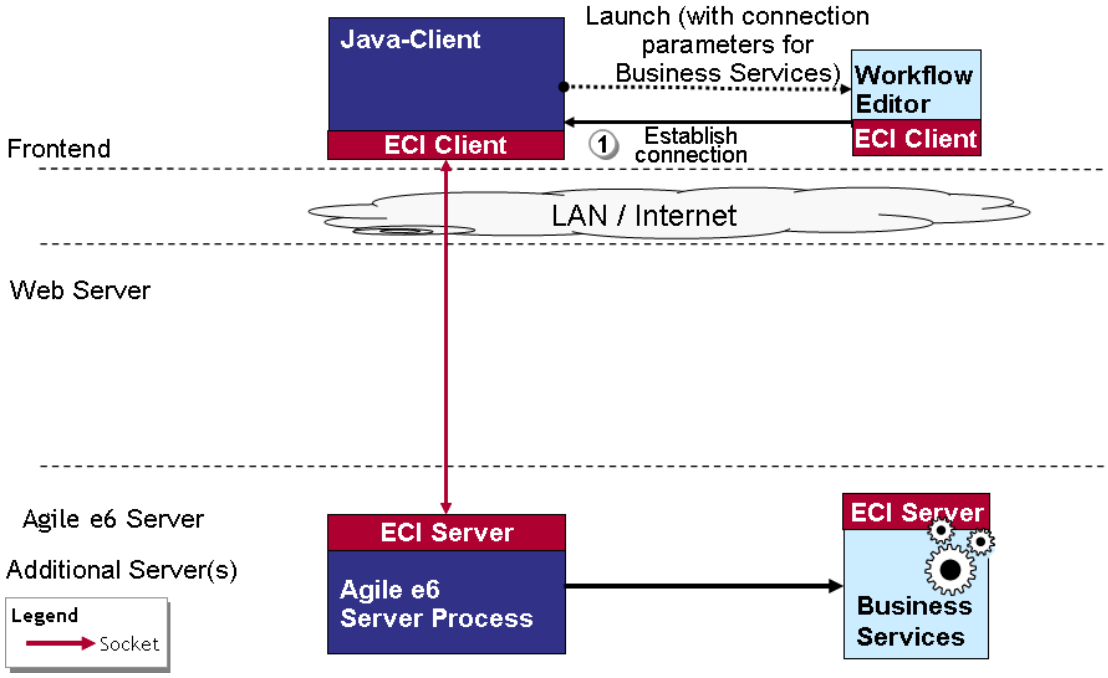
The following table contains the details about the relevant lines of communication:

Line of Communication	Type	Port or Range of Ports
(1)	http (or https)	Port 8088 by default (can be configured during installation)
(2)	RPC	Port 111 (used by Sun RPC)
(3)	RPC	One unprivileged port for each file FMS Server (typically only one), assigned by Portmapper – see (2)

Line of Communication	Type	Port or Range of Ports
(4)	Application specific protocol based on Sockets	One port for each concurrent file upload or download in range from 52517 to 53516

Lines of Communication when Using the Workflow Editor

The communication between the Java Client, the Workflow Editor and the Agile e6 application server process is depicted in the following figure:



The following steps are executed when the user launches the Workflow Editor from the Java Client:

- The Java Client launches the Workflow Editor. Parameters with information about the Business Services are passed.
- The Workflow Editor connects to the given Business Services through the Java client and Agile e6.1 Server.

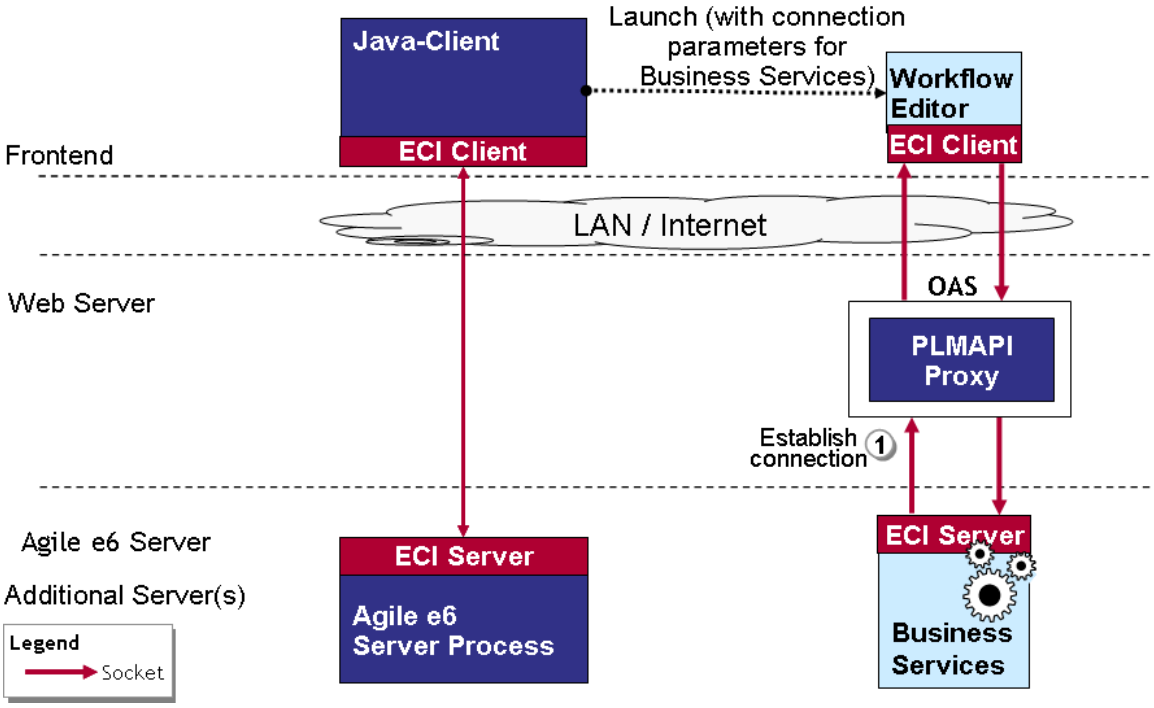
The following table contains the details about the relevant lines of communication:

Line of Communication	Type	Port or Range of Ports
(1)	Application specific protocol based on Sockets	Port 19997 by default (can be configured during installation)

With HTTP Support

The Workflow Editor is a component in the Java client that connects to the business service through an ECI protocol.

Once the Java client is communicating in a WAN scenario, the firewall restrictions also apply for the Workflow Editor, thus the Workflow also needs to communicate through the PLMAPI proxy as described for the Java client in "Lines of Communication with HTTP support".

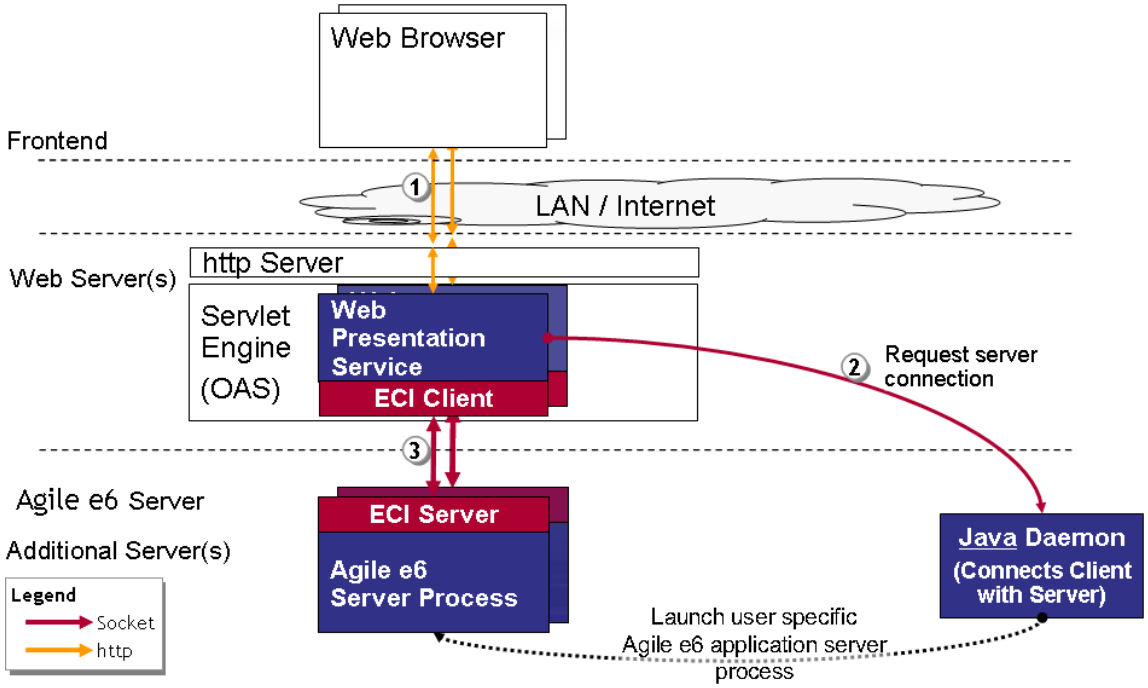


Chapter 5

Web Client Communication

Lines of Communication when Launching the Web Client

The lines of communication when starting the Web Client (respectively when accessing the corresponding URL in a web browser) are depicted in the following figure:



The following steps are executed when the user launches the Web Client:

- The web browser sends a request to the Web Presentation Service (communication line (1)) that is hosted by a servlet engine.
  Remark: it is possible to configure the web browser in such a way, that it bypasses the http server and communications directly with the servlet engine. This configuration may be configured if the web browser is not used for other (http) services but the Agile e6 Web Client.
- The Web Presentation Service connects the Java Daemon, requesting to launch a user specific Application Server Process and to return the connection information for this process (communication line (2)).
- The Web Presentation Service connects to the given Application Server Process via ECI (communication line (3)).

The Web Presentation Service will process requests from the web browser by translating these into ECI calls to the Application Server Process. The Web Presentation Service is also responsible for

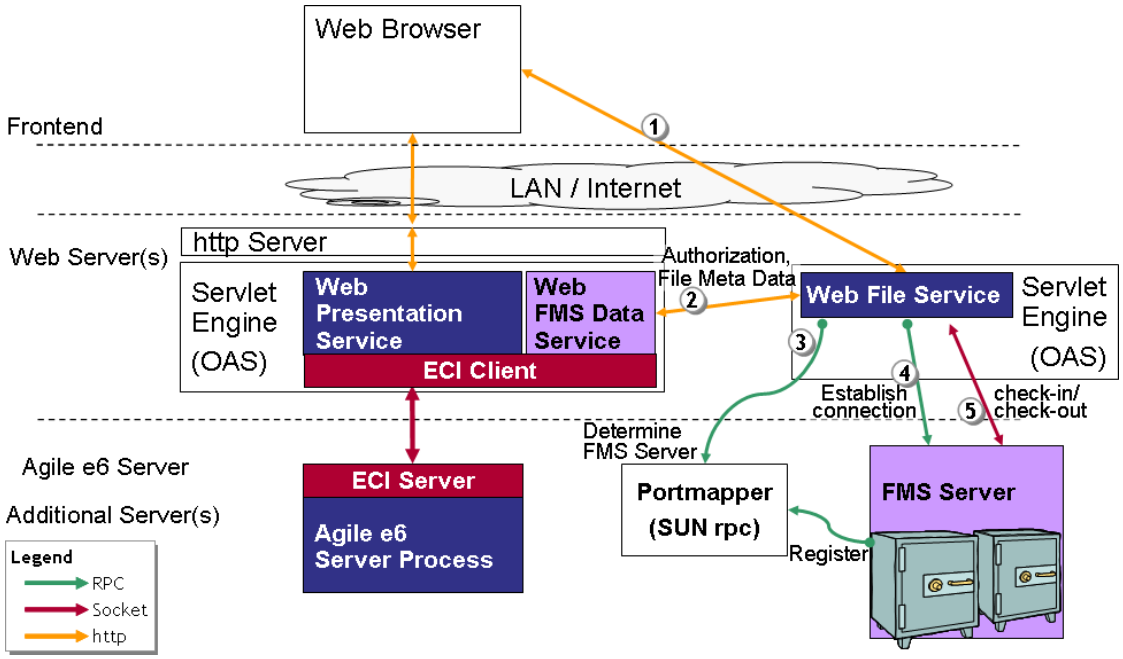
the rendering of lists and forms into DHTML (dynamic HTML - HTML enriched with JavaScript). The JavaScript in the HTML pages realizes consistency checks and interactive controls within the Web Client.

The following table contains the details about the relevant lines of communication:

Line of Communication	Type	Port or Range of Ports
(1)	http (or https)	Port 8088 by default (can be configured during installation)
(2)	Application specific protocol based on Sockets	Port 16077 by default (can be configured during installation)
(3)	Application specific protocol based on Sockets	One unprivileged port per Application Server Process from range as configured for Java Daemon (5000 to 5999 by default)

Lines of Communication in case of File Access (using FMS)

The communication between the Web Client (respectively the web browser), the Agile e6 application server process, the web services and the FMS Server is depicted in the following figure:



The following steps are executed when the user performs a check-in or a check-out operation in the Web Client (respectively the web browser):

- The web browser sends a request to the Web Presentation Service. This request is recognized as a file check-in or check-out operation and as a result the web-browser is re-directed to the Web File Service (communication line (1)). The address of the Web File Service is taken from

the configuration of the corresponding vault in the Agile e6 database. For a check-out operation, the web browser sends the corresponding request parameter to the Web File Server. For a check-in operation, the web browser sends the corresponding file to the Web File Server.
Remark: The Web Presentation Service and the Web File Service can be hosted by the same servlet engine. For performance reasons it is recommended to use two separate servlet engines.
The web browser can be configured that it bypasses the http server and communications directly with the servlet engine. This configuration may be configured if the web browser is not used for other (http) services but the Agile e6 Web Client.

- To avoid a misuse of the Web File Service, the re-directed request contains an authorization ticket and the address of the Web FMS Data Service. This service is hosted by the same servlet engine as the Web Presentation Service and they share a single ECI connection to the Application Server Process. The Web File Service provides the authorization ticket and in return receives required file meta data (communication line (2)).
- With the RPC-number of the FMS Server, the Web File Service connects to the Portmapper to determine the port of the corresponding FMS Server (communication line (3)).
- Now the Web File Service can request the FMS Server to spawn an individual thread and return the connection parameter (communication line (4)).
- The file data are transmitted between the Web File Service and the FMS Server thread (communication line (5)). The Web File Service itself returns the call to the web browser.

Files are not temporarily stored on the Web Server, but transferred directly between the web browser and the FMS Server. This avoids possible security issues (files that resided temporarily on the web server are accessible to all users) and increases performance especially in the case of large files, because only a minimum of file write/read operations are performed.

The following table contains the details about the relevant lines of communication:

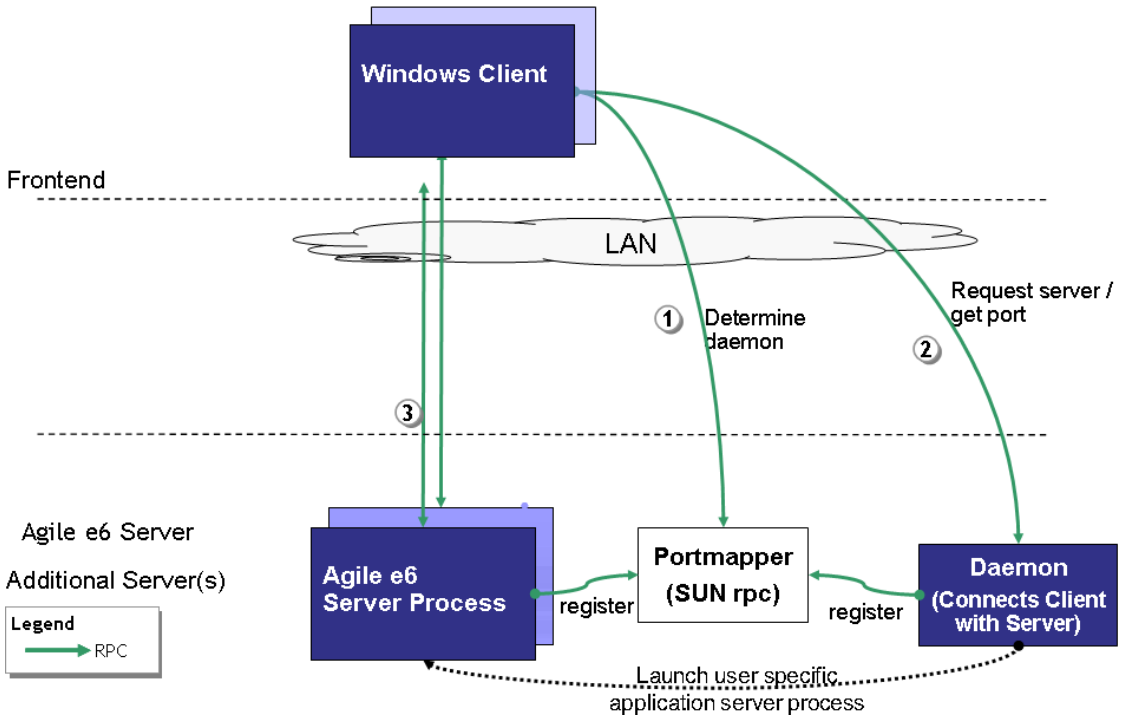
Line of Communication	Type	Port or Range of Ports
(1)	http (or https)	Port 8088 by default (can be configured during installation)
(2)	http (or https)	Port 8088 by default (can be configured during installation)
(3)	RPC	Port 111 (used by Sun RPC)
(4)	RPC	One unprivileged port for each file FMS Server (typically only one), assigned by Portmapper – see (3)
(5)	Application specific protocol based on Sockets	One port for each concurrent file upload or download in range from 52517 to 53516.

Chapter 6

Windows Client Communication

Lines of Communication when Launching the Windows Client

The following figure depicts the lines of communication between the Windows Client and the Application Server Process:



The following steps are executed when the user launches the Windows Client:

- The Windows Client connects to the Portmapper to get the address of the required Daemon (communication line (1)). The Daemon had registered at the Portmapper initially and is therefore known to the Portmapper.
- The Windows Client connects the Daemon, requesting to launch a user specific Application Server Process and to return the connection information for this process (communication line (2)).
- The Windows Client connects to the given Application Server Process and starts communication (communication line (3)).

The communication line between the Windows Client and the Application Server Process uses a

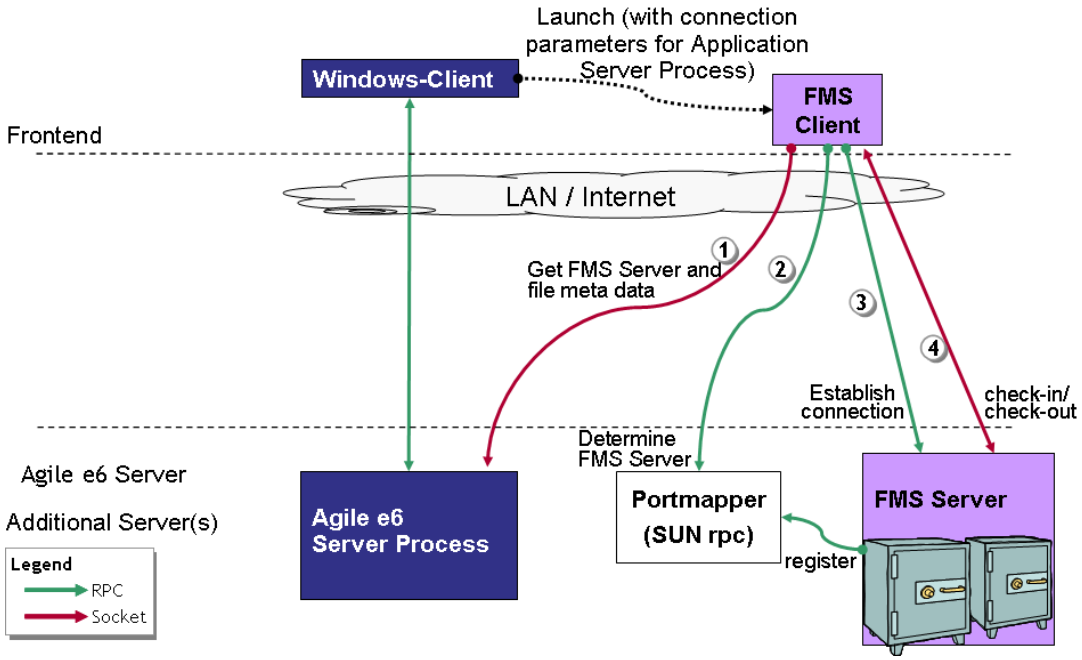
proprietary protocol that is very efficient in that it only transfers a minimum of data.

The following table contains the details about the relevant lines of communication:

Line of Communication	Type	Port or Range of Ports
(1)	RPC	Port 111 (used by Sun RPC)
(2)	RPC	A single unprivileged port, assigned by Portmapper – see (2)
(3)	RPC	One unprivileged port for each Application Server Process.

Lines of Communication in Case of File Access (using FMS)

The following figure depicts the lines of communication for the File Access using FMS:



The following steps are executed when the user performs a check-in or a check-out operation in the Windows Client:

- Windows Client launches the FMS Client. Parameters about the current Application Server Process are passed.
- FMS Client connects to the given Application Server Process to get the corresponding FMS Server data and file meta data (communication line (1)).
- With the information about the FMS Server, the FMS Client connects to the Portmapper to determine the port of the corresponding FMS Server (communication line (2)).
- Now the FMS Client can request the FMS Server to spawn an individual thread and return the

connection parameter (communication line (3)).

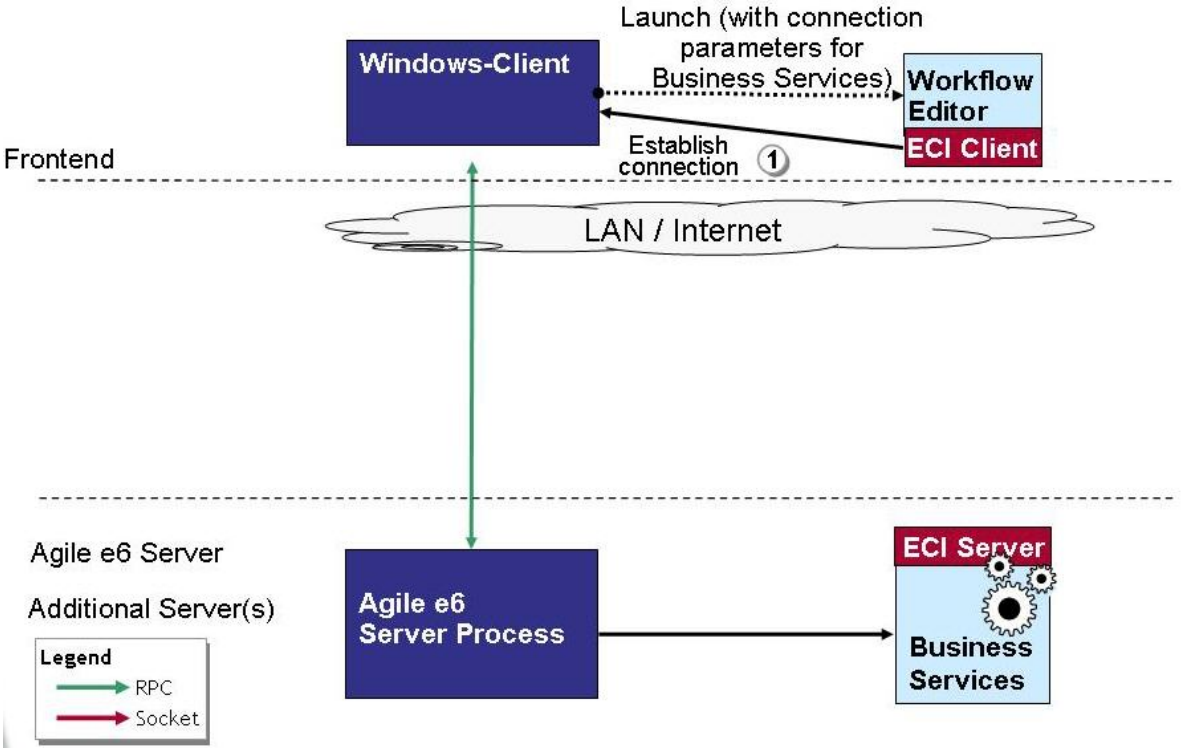
- The file data is directly transmitted between the FMS Client and the FMS Server thread (communication line (4)).
- The FMS Client will terminate after a given timeout period.

The following table contains the details about the relevant lines of communication:

Line of Communication	Type	Port or Range of Ports
(1)	Application specific protocol based on Sockets	One port for each concurrent file upload or download in range from 51516 to 52515
(2)	RPC	Port 111 (used by Sun RPC)
(3)	RPC	One unprivileged port for each file FMS Server (typically only one), assigned by Portmapper – see (2)
(4)	Application specific protocol based on Sockets	One port for each concurrent file upload or download in range from 52517 to 53516

Lines of Communication When Using the Workflow Editor

The following figure depicts the lines of communication when using the Workflow editor:



The following steps are executed when the user launches the Workflow Editor from the Windows

Client:

- Windows Client launches the Workflow Editor.
- The Workflow Editor connects to the given Business Services (communication line (1)).

The following table contains the details about the relevant lines of communication:

Line of Communication	Type	Port or Range of Ports
(1)	Application specific protocol based on Sockets	Port 19997 by default (can be configured during installation).

Chapter 7

Additional Aspects

Business Services

It is recommended to install the Business Services on the Application Server, not on a separate computer. This simplifies the communication between the Application Server Processes and the Business Services.

Access to Administration Interface for Business Services

The Business Services are managed (started and stopped) via a HTML-based administration interface.

By default port 12808 is used for this connection – this can be configured. Via your firewall configuration, access to this port should be strictly limited to prevent misuse of this administration interface.

Access to Administration Service

For creating and managing environments (e.g. test environment, development environment, in-production environment), the Administration Service is provided. An HTML-based administration interface is available.

Note	Further information regarding the respective ports can be found in the Administration Manual.

The Administration Service is no longer required once the environments have been created and the in-production environment is deployed.

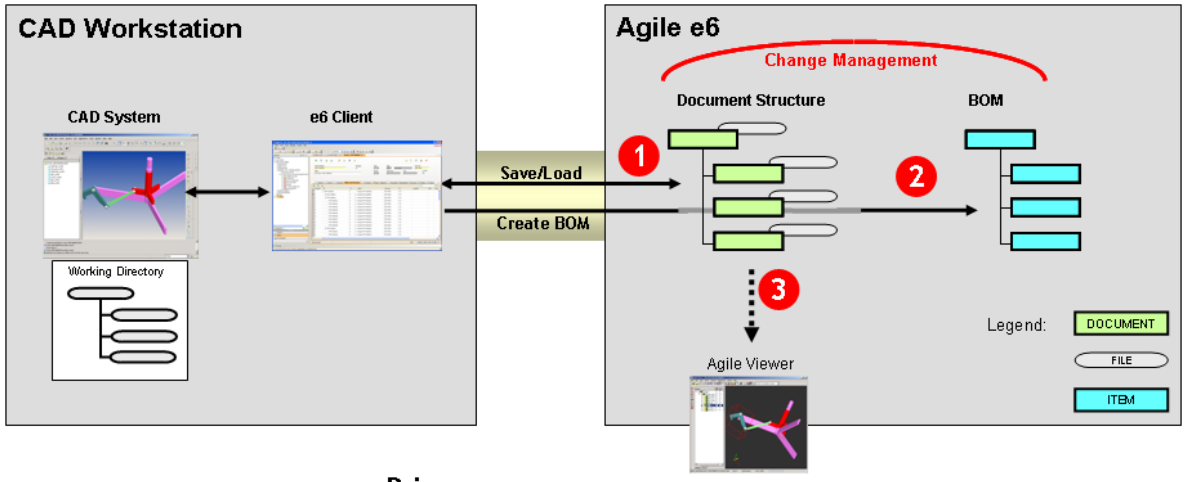
Configuration for Server-side Mailing

Via a userexit it is possible to send out e-Mails from the Application Server Process. Likewise the Business Services send out Workflow-related e-Mails.

The Application Server therefore needs to have access on port 25 (standard port used for SMPT) of the Mail Server.

Chapter 8

Integrations



Primary use cases:

1. **Save and Load CAD files**
2. **Create BOM**
3. **View CAD files (optional)**

Agile e6 CAD Integrations provides data and process integration between CAD applications and Agile e6. They allow CAD designers and engineers to capture and control the data representing a primary source of the product record. The integrations work from within the familiar CAD system user interface, providing commands which allow the user to interact with PLM to manage the CAD data.

CAD Systems

The following CAD Systems are supported with Agile e6:

- Solid Edge (CCH)
- Solid Works (CCM)
- Catia V 5(ECC)
- ProEngineer (ECP)
- Unigraphics (ECU)

The following additional integrations are available from partners:

- Autocad
- Inventor

Use Cases

Save and Load CAD Files

CAD designs are created within the designer's CAD workstation environment, with files in a working directory (which may be local, or network attached). The designer saves into Agile, which creates a document structure that mimics the structure of the CAD assembly. The native CAD files are attached to this document structure, which is used as the basis for loading and re-saving the CAD designs. Since Agile is a centralized repository, all CAD designers in the enterprise have access to these files, subject to the control of Agile access rights. Individual designers can set checkout reservations in Agile when they load files into their CAD session. Additional files such as viewables (PDF, TIFF, etc.) can be attached to the document structure.

Create BOM

When a design or design change has been completed, the designer may use the Create BOM function to create or update the Agile Item BOM, representing the true Product Structure. This function is used when there is a high correlation between the Document Structure and Product Structure, to avoid tedious manual entry of the BOM. Create BOM will create or update the BOM either for all sub-structures, or only one level (flat). To avoid unnecessary manual entries in the BOM, it is possible to merge standard parts and auxiliary parts. This function is used when an item represented in the design structure already exists. Creating and updating the BOM can be performed either in the background, or interactively, which enables modification of item attribute information in Agile e6.

View CAD Files (optional)

One reason for managing the native CAD files within Agile is that the Agile viewer (AutoVue) can be used to view and markup the files. This works across the web, and without having the native CAD system. Advanced functionality such as digital mockup, 3D comparison, interference checking, and real-time collaboration make this an important tool to support the overall design process. Support for AutoVue is not provided as standard with Agile e6 but can be configured on a project-specific basis.