agile How Products Become Profits™ EIGNER PLM 5.1

Product Information for Agile e5 (Eigner PLM 5)

Services and Communication Types

Copyrights and Trademarks

The material in this document is for information only and is subject to change without notice. While reasonable efforts have been made in the preparation of this document to ensure its accuracy, Agile Software Corporation assumes no liability resulting from errors or omissions in this document or from the use of the information contained herein. Agile Software Corporation reserves the right to make changes in the product design without reservation and without notification to its users.

NOTICE OF RESTRICTED RIGHTS:
The Software is a “commercial item,” as that term is defined at 48 C.F.R. 2.101 (OCT 1995), consisting of “commercial computer software” and “commercial computer software documentation” as such terms are used in 48 C.F.R. 12.212 (SEPT 1995) and when provided to the U. S. Government, is provided (a) for acquisition by or on behalf of civilian agencies, consistent with the policy set forth in 48 C.F.R. 12.212; or (b) for acquisition by or on behalf of units of the Department of Defense, consistent with the policies set forth in 48 C.F.R. 227.7202-1 (JUN 1995) and 227.7202-4 (JUN 1995).

Part Number: TP1107-8.8B

June 22, 2004

(EP-00057675, 0)

CONTENTS

Chapter 1

Architecture Overview

This chapter gives an introduction to the overall architecture of Agile e5 (Eigner PLM 5), including its services and their communication types.

Architecture Overview

The Agile e5 system architecture is based on an object-relational database (Oracle or Microsoft SQL-Server). A central component is the object-oriented repository. It is unique in that it stores all meta data which define the application with respect to:

- Object model
- User interface
- Business logic

The business logic consists e.g. of

- Lifecycle definitions
- Workflow processes
- Consistency Checks
- Automation Scripts

The repository ensures the separation of system description and physical construction. Because all meta data is stored in the database, deployment and upgrade processes are simplified.

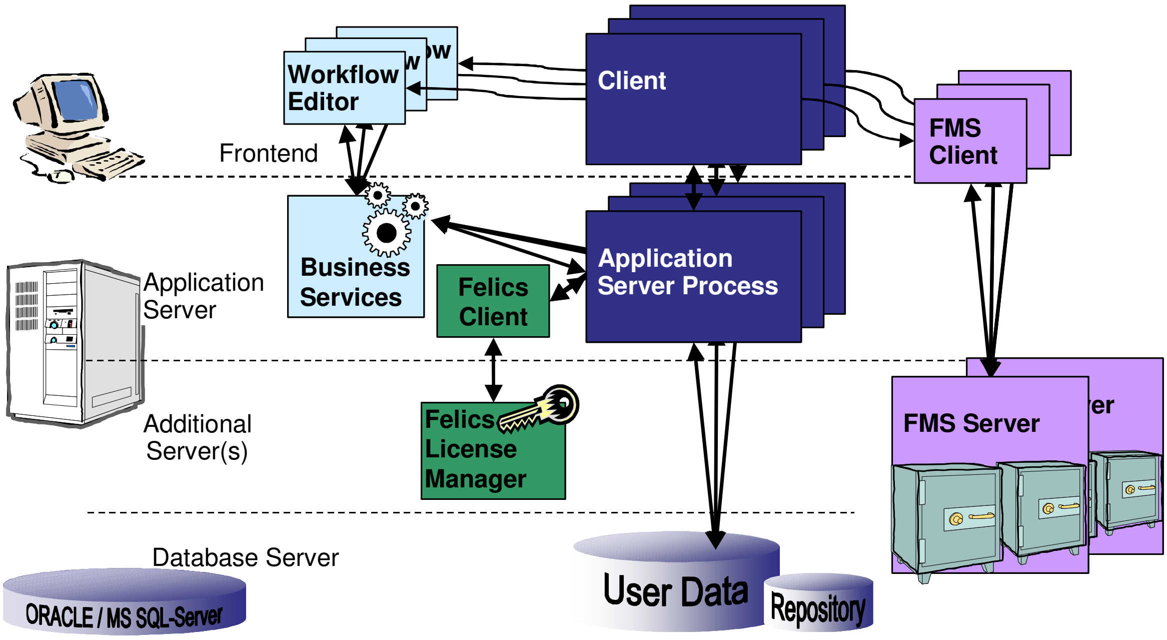
Various types of clients are available, serving the different needs of casual users and power users (see Chapter 2, page 3). With each client process that is launched, an application server process is started in parallel. After having received a license from the central license manager, the application server process interprets the repository and can thus dynamically reflect any changes applied to the meta data. Accessing any user data (stored in the object-relational database) is taken care of by the application server process[1]. This ensures that the clients do not require a database connection.

This separation of services is called a Three-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 License Client (responsible to communicate with the License Manager), the File Management Service "FMS" (responsible for managing physical files) and the Business Services (consisting of the Workflow Engine, responsible

[1] 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.

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 3, page 6).
These separate services can reside on the same server where the application server processes are executed (this is mandatory for the License Client and 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 e5 actually features a distributed architecture.

The following picture gives an overview of the main services which are used in the architecture of Agile e5:



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

The following chapters explain in more detail about the main services and about the communication between these services. The goal is especially to provide an understanding of the communication types that affect the configuration of a firewall. Generally spoken, all communication is based on TCP/IP. Only unprivileged ports (above 1024) are used. Once a TCP/IP connection is established, it will not dynamically change the port.

Lines of communication that are local to the Client, local to 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

Available Clients

With Agile e5, multiple types of clients are available, serving the different needs of casual users and power users. This chapter characterizes the following clients:

- ❑ Windows Client
- ❑ Unix Client
- ❑ Java Client
- ❑ Web Client

Windows Client

The Windows Client is the standard client for Agile e5. It enables access to all functions of Agile e5. Therefore all types of users may use the Windows Client. Administrators and Customizers must use the Windows Client.

Thanks to its tight integration with the underlying Windows operating system, users have the benefit of drag&drop operations between Agile e5 and other Windows applications (e.g. Windows Explorer, Microsoft Excel).

The Windows Client fully supports available M-CAD, E-CAD, EDA integrations and most other types of integrations (e.g. SAP Link).
The Office Suite is exclusively available with the Windows Client, allowing users to manage files created with Microsoft Office authoring tools (mainly Word, Excel, Powerpoint). Files check-in via the Office Suite can be access from all other clients too (if a suitable application is available to display the corresponding file type), however the 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 Windows Client.

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

Unix Client

The Unix Client is used by power users on Unix front-ends. It enables access to all functions of Agile e5 (which are not Windows-specific of course). The Unix Client is not suitable for Customizers and offers only limited support for Administrators.

The most important limitations compared to the Windows Client include:

- No integrated Explorer window
- No support for significant fields in lists (that means all columns in a list will scroll horizontally)
- Basic support for drag&drop within the client, but no drag&drop from/to the desktop

The Office Suite is obviously not supported, however the Unix Client does fully support M-CAD, E-CAD, EDA integrations (where the corresponding system itself is available on Unix) and most other types of integrations (e.g. SAP Link).

The user interface of the Unix Client is dynamically defined by the meta data in the repository.

Java Client

The Java Client is optionally available in Agile e5. It will replace the platform-dependant Unix Client in a later release of Agile, but can be tested and used with restricted support in Agile e5 already.

The Java Client can be used by power users on all front-ends[2]. It enables access to all functions of Agile e5 (which are not Windows-specific). The Java Client is not suitable for Customizers 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 most important limitations compared to the Windows Client include:

- Different concept of the integrated Explorer window (this optional windows can be attached to most form and grants a structured overview of all objects loaded in the form).
- No support for significant fields in lists (that means all columns in a list will scroll horizontally).
- No support for drag&drop within the client or from/to the desktop.
- Modal dialogs may not be supported completely (however the cases that are not supported have been dramatically reduced, allowing the Java Client to be used in most scenarios).

The Office Suite is not supported, however the Java Client does fully support M-CAD, E-CAD, EDA integrations (where the corresponding system itself is available on corresponding front-end) and most other types of integrations (e.g. SAP Link).

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

Web Client

The Web Client can be used by casual users on all front-ends[3]. It enables access to most functions of Agile e5 (which are not Windows-specific). The Web Client is not suitable for Customizers and Administrators.

[2] The Java Client requires a Java Runtime Environment

[3] The Web Client requires a Microsoft Internet Explorer or Netscape/Mozilla. For more details, please check the up-to-date list of supported platforms in the Agile support area.

The Web Client uses HTML respectively DHTML[4] only requires no local installation[5]. It is therefore the ideal client for zero-deployment costs.

The most important limitations compared to the Windows 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 dramatically reduced, allowing the Java Client to be used in most scenarios).

The Web Client does neither support Office Suite integrations nor M-CAD, E-CAD, EDA integrations. However the Web Client can be used with integrations that are processes on by server, e.g. the SAP Link.

The user interface of the Web Client is dynamically defined by the meta data in the repository, except for some specific components, including the integrated Explorer windows, the Search Panel and the Wizard.

[4] Dynamic HTML, a combination of HTML and JavaScript.

[5] To be precise: actually there is no Web Client. There only is a Web Presentation Service on the web server, which can be access by a web browser. For reasons of simplicity, this document will however talk about the "Web Client".

Chapter 3

Dedicated Services

This chapter provides an introduction to additional services:

- ❑ File Management Services
- ❑ Workflow

File Management Service

Files are stored on any server in the network under control of the File Management Service (FMS). A FMS client communicates with the corresponding FMS server to check-in and check-out files.

The File Management Service manages documents (referred to as "Files") in Electronic Vaults, thus implementing the "check-in" and "check-out" functionality provided by the Document Management System in Agile e5. The File Management Service consists of a FMS Client and a FMS Server. The FMS Client is installed – together with the Agile e5 Windows Client or the Agile e5 Unix Client – on each front-end[6]. The FMS Server is installed on one or more server computers. Each FMS Server can manage one or more Electronic Vaults.

- If the FMS Server is installed on a Windows platform, this server must be NTFS based. FAT does not work.
- The Electronic Vaults managed by a FMS Server have to be created on local hard discs of the computer where the corresponding FMS Server is running.
- The FMS Client communicates with the Agile e5 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 e5 application server process.

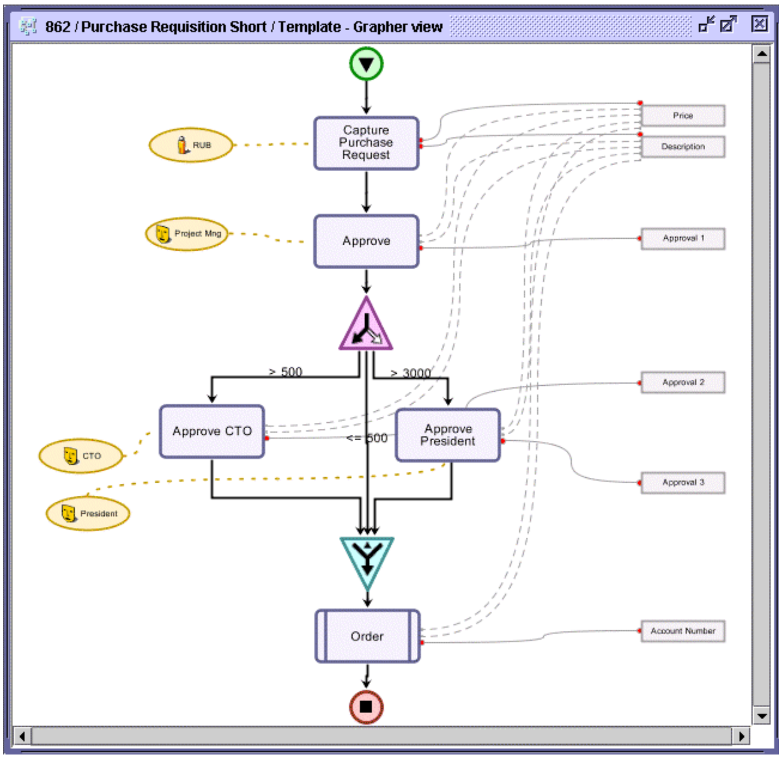
In a very simple configuration, there is only one FMS Server with a single Electronic Vault (e.g. when Agile e5 is installed on a single workgroup server or on a laptop).
In large installations of Agile e5, there may be FMS Servers running on several computers, each FMS Server managing multiple Electronic Vaults.
Installations with remote users that connect to an Agile e5 Server through a Wide Area Network (WAN), e.g. an external office that is connected to the headquarter, would usually be limited when accessing files by a small bandwidth. To improve performance in such a configurations, a FMS Server and one or more Electronic Vaults can also be installed at the remote location (even though the Agile e5 server and the database are running at the headquarter). Files can be checked-in and checked-out into these Electronic Vaults by any FMS Client that is able to communicate with this FMS Server.

[6] 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, not requiring any installation on the front-end.

Workflow

Agile e5 includes a workflow solution that allows to automate 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 Client, Unix Client and the Java Client.

Chapter 4

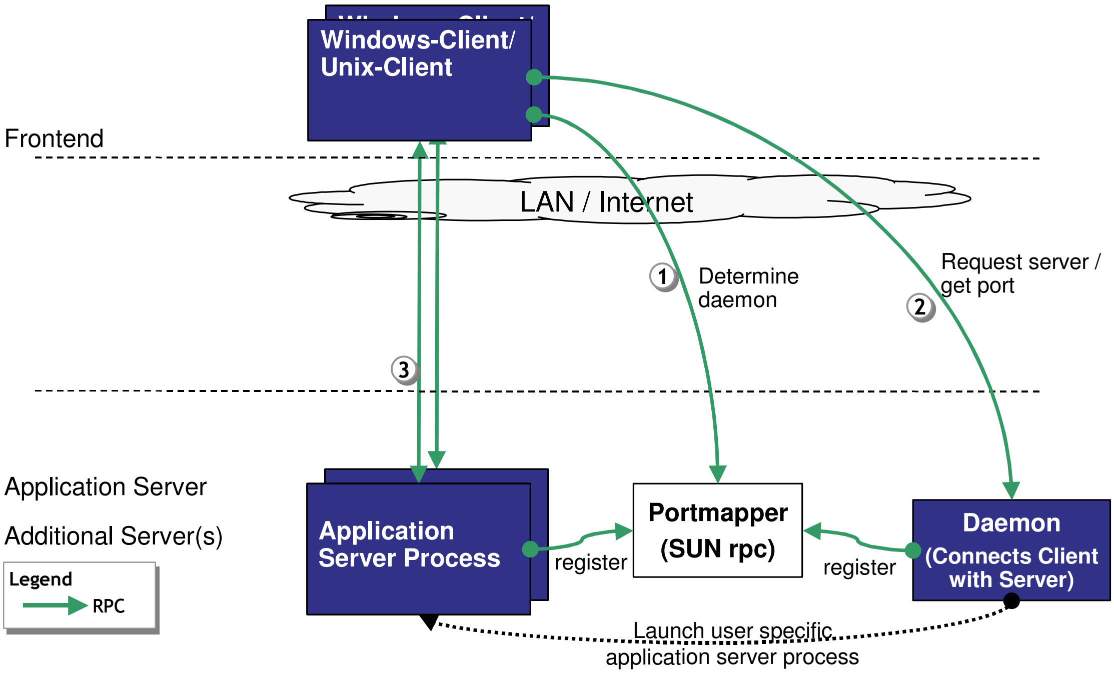
Windows Client Communication

This chapter describes the lines of communication for the Windows Client:

- ❑ Communication between Windows Client (respectively Unix Client) and the Application Server Process
- ❑ Communication with FMS Server

Lines of Communication when launching the Windows Client

The steps when launching the Windows Client are identical with launching the Unix Client. The following figure depicts the lines of communication:



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

- The Windows Client/Unix Client connects to the Portmapper to get the address of the required Daemon (communication line (1) in the figure above). The Daemon had registered at the Portmapper initially and is therefore known to the Portmapper.
- The Windows Client/Unix 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) in the figure above).
- The Windows Client/Unix Client connects to the given Application Server Process and starts communication (communication line (3) in the figure above).

The communication line between the Windows Client/Unix Client and the Application Server Process uses a proprietary protocol which is very efficient in that it only transfers a minimum of data.

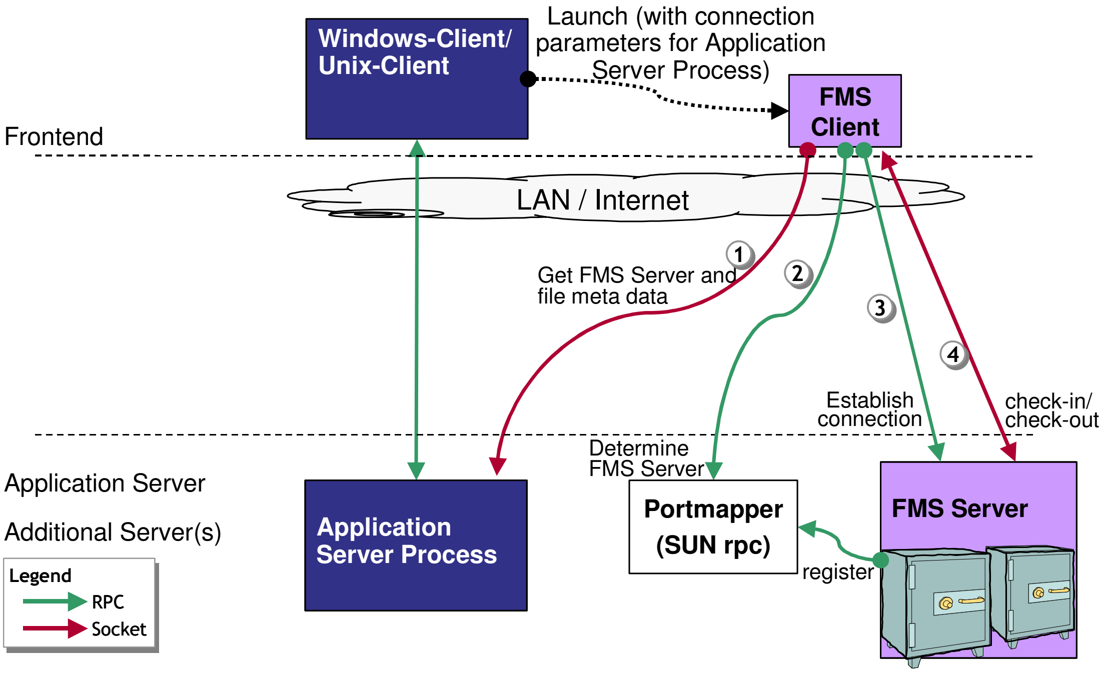
The following table contains the details about the relevant lines of communication:

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.

Table 1

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

The communication between the Windows Client, the FMS client, the Agile e5 application server process and the FMS Server is identical for the Unix Client. The following figure depicts the lines of communication:



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

- Windows Client/Unix 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) in the figure above).
- 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) in the figure above).
- Now the FMS Client can request the FMS Server to spawn an individual thread and return the connection parameter (communication line (3) in the figure above).
- The file data are directly transmitted between the FMS Client and the FMS Server thread (communication line (4) in the figure above).
- The FMS Client will terminate after a given timeout period.

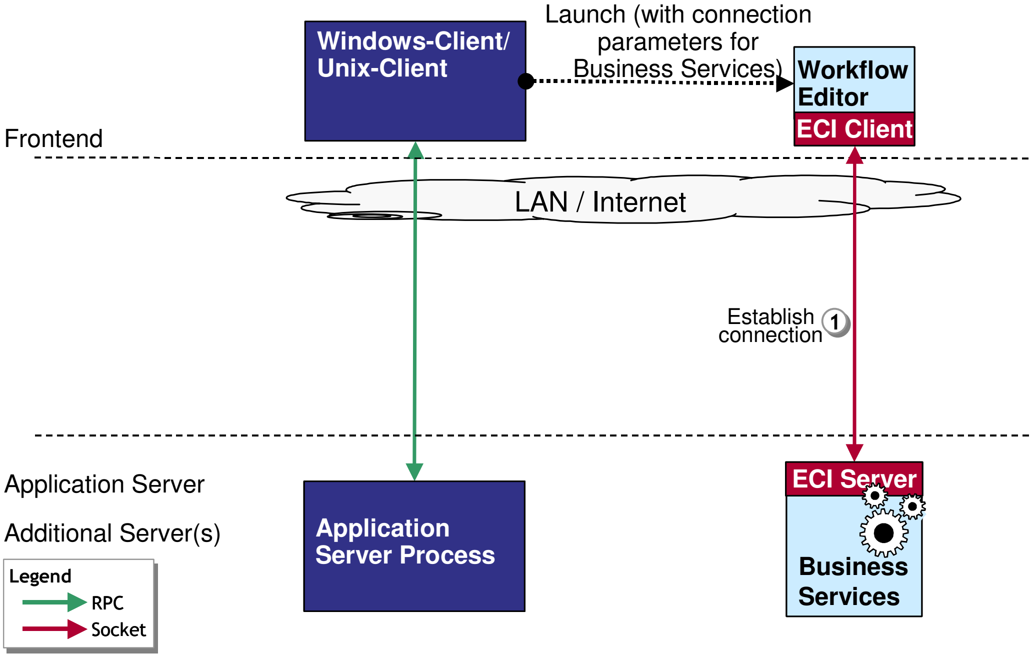
The following table contains the details about the relevant lines of communication:

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

Table 2

Lines of Communication when using the Workflow Editor

The communication between the Windows Client, the Workflow Editor and the Agile e5 application server process is identical for the Unix Client. The following figure depicts the lines of communication:



The following steps are executed when the user launches the Workflow Editor from the Windows Client or the Unix Client:

- Windows Client/Unix Client launches the Workflow Editor. Parameters with information about the Business Services are passed.
- The Workflow Editor connects to the given Business Services (communication line (1) in the figure above).

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).

Table 3

Chapter 5

Unix Client Communication

Lines of Communication are identical for the Windows Client and the Unix Client. Please refer to Chapter 4, page 8.

Chapter 6

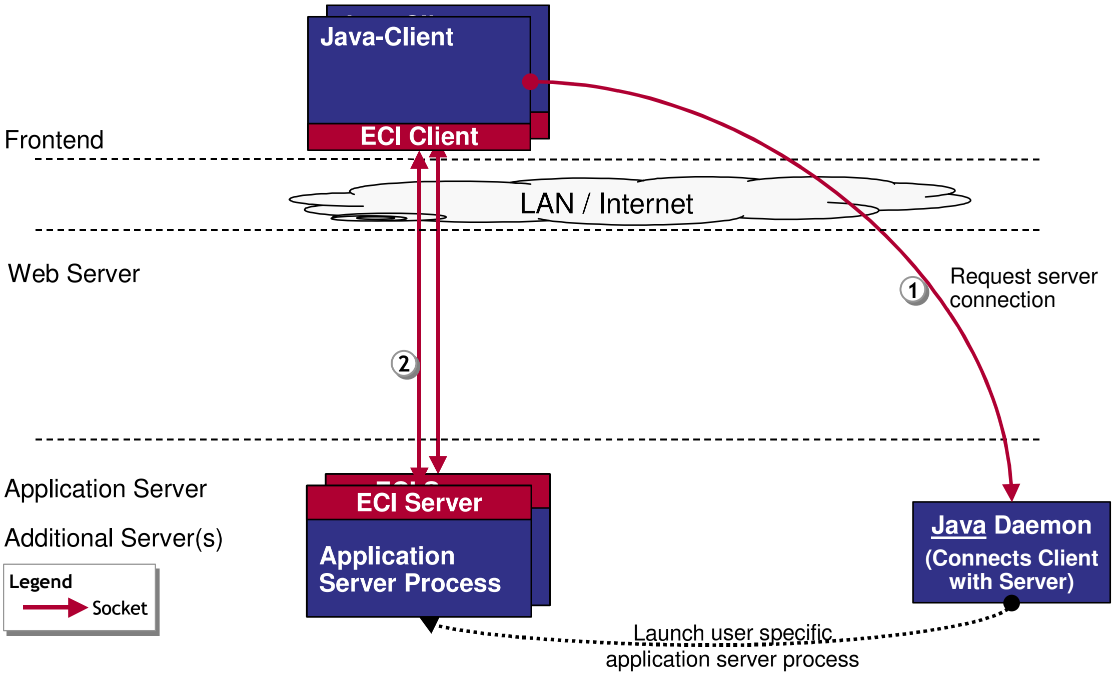
Java Client Communication

This chapter describes the lines of communication for the Java Client:

- ❑ Communication between Java Client and the Application Server Process
- ❑ Communication with FMS Server

Lines of Communication when launching the Java Client

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


Java-Client
ECI Client
Frontend
LAN / Internet
Web Server
1
Request server connection
2
Application Server
Additional Server(s)
ECI Server
Application Server Process
Java Daemon (Connects Client with Server)
Legend
Socket
Launch user specific application server process

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

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

The communication line between the Java Client and the Application Server Process utilizes the ECI[7] protocol – the same that is used by e.g. a M-CAD system to communicate to an Agile e5 client on the front-end.

[7] External Communication Interface

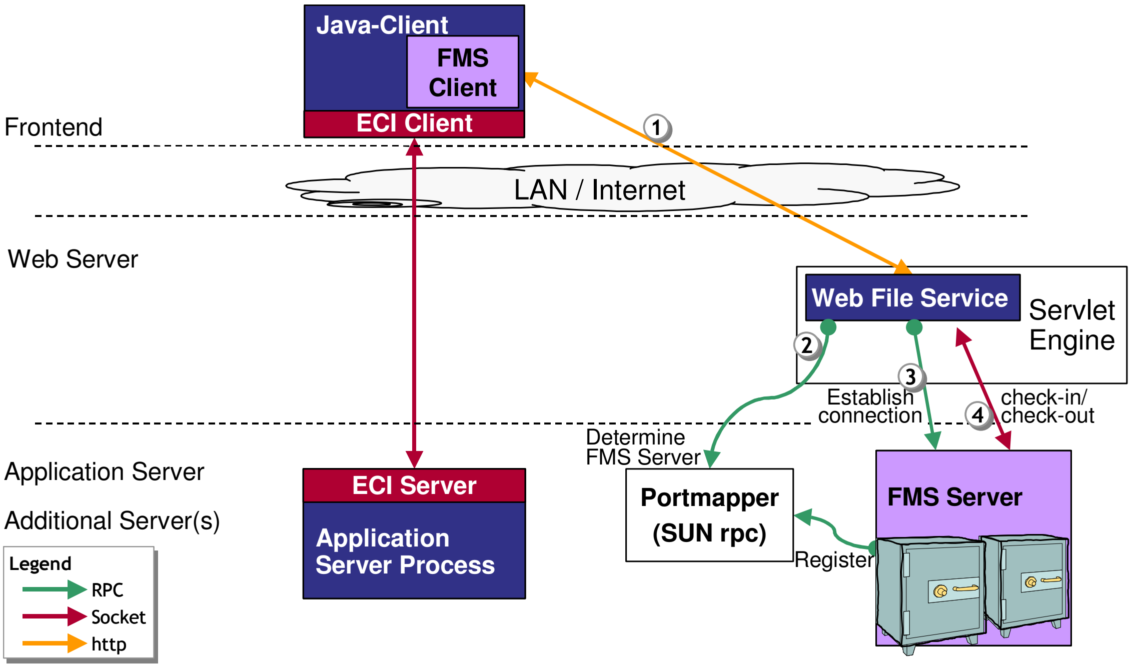
The following table contains the details about the relevant lines of communication:

Line of Communication	Type	Port or Range of Ports
(1)	Application specific protocol based on Sockets	Port 16067 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)

Table 4

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

The communication between the Java Client, the Agile e5 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) in the figure above). The address of the Web File Service is taken from the configuration of the corresponding vault in the Agile e5 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) in the figure above).
- Now the Web File Service can request the FMS Server to spawn an individual thread and return the connection parameter (communication line (3) in the figure above).
- The file data are transmitted between the Web File Service and the FMS Server thread (communication line (4) in the figure above). 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.

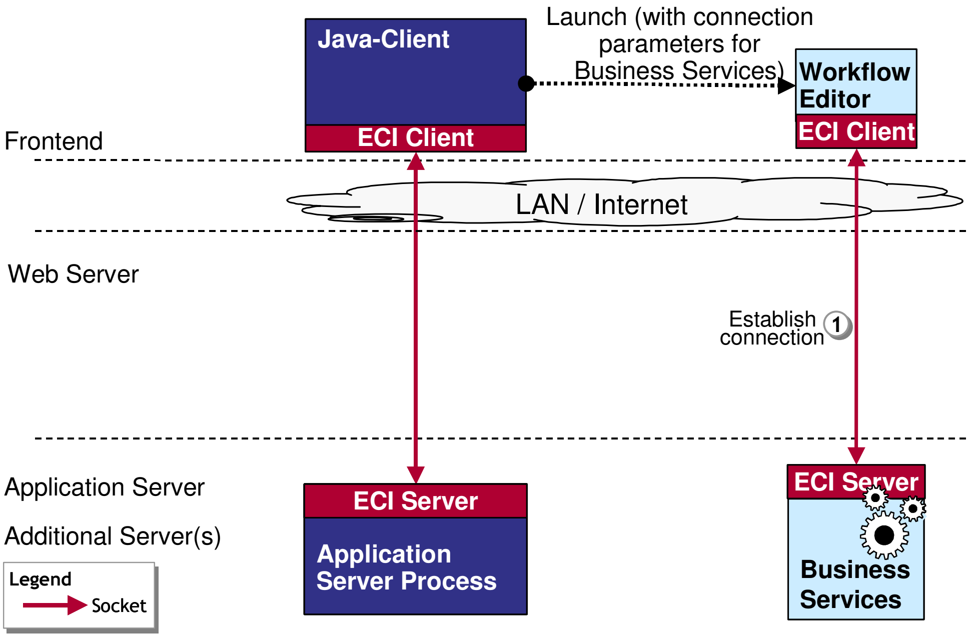
The following table contains the details about the relevant lines of communication:

Line of Communication	Type	Port or Range of Ports
(1)	http (or https)	Port 8080 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)
(4)	Application specific protocol based on Sockets	One port for each concurrent file upload or download in range from 52517 to 53516

Table 5

Lines of Communication when using the Workflow Editor

The communication between the Java Client, the Workflow Editor and the Agile e5 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:

- Java Client launches the Workflow Editor. Parameters with information about the Business Services are passed.
- The Workflow Editor connects to the given Business Services (communication line (1) in the figure above).

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)

Table 6

Chapter 7

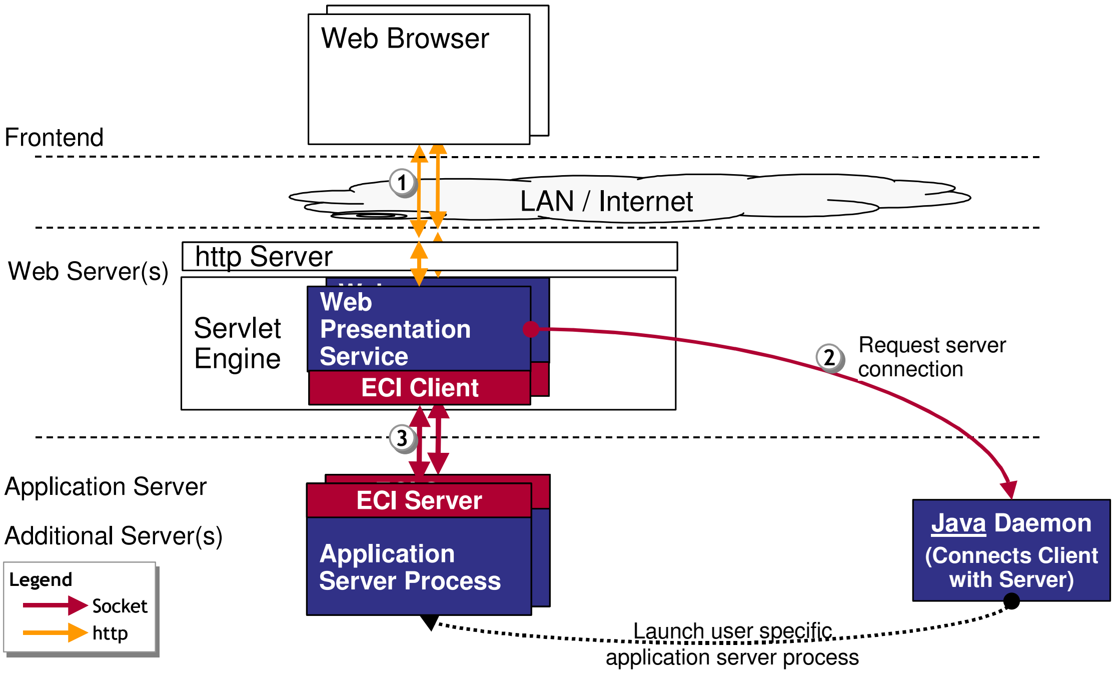
Web Client Communication

This chapter describes the lines of communication for the Web Client:

- ❑ Communication between Web Client and the Application Server Process
- ❑ Communication with FMS Server

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) in the figure above) which 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 e5 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) in the figure above).

- The Web Presentation Service connects to the given Application Server Process via ECI[8] (communication line (3) in the figure above).

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:

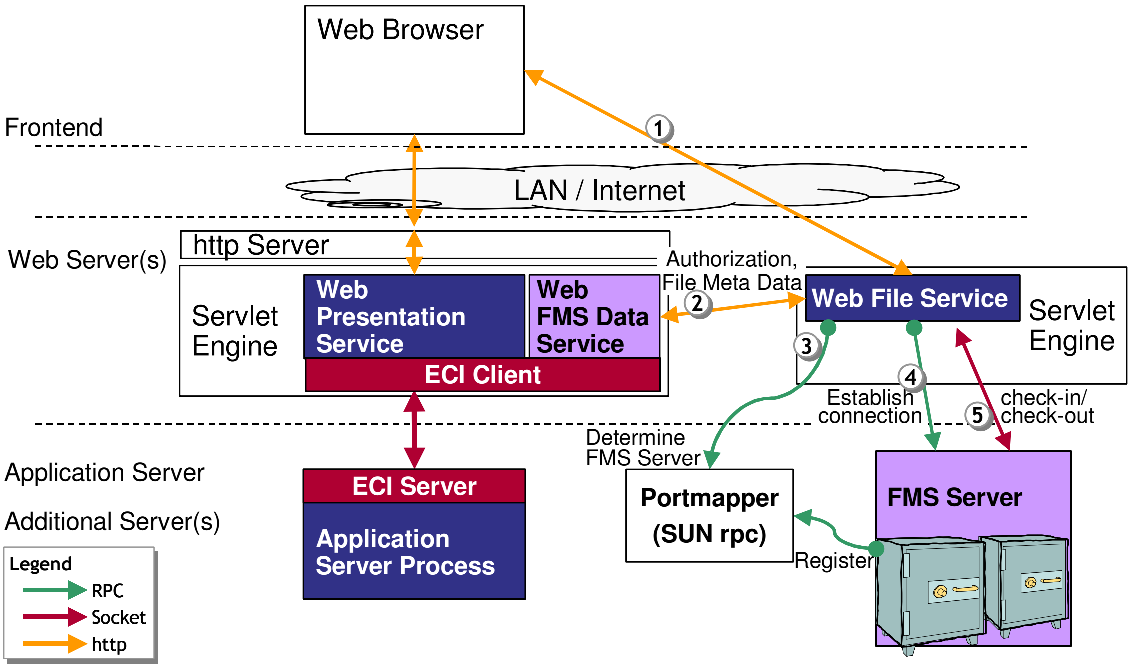
Line of Communication	Type	Port or Range of Ports
(1)	http (or https)	Port 8080 by default (can be configured during installation)
(2)	Application specific protocol based on Sockets	Port 16067 by default (can be configured during installation)
(3)	Application specific protocol based on Sockets	One unprivileged port per Application Server Process from range as configured for Java Daemon (5000 to 5999 by default)

Table 7

[8] External Communication Interface

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

The communication between the Web Client (respectively the web browser), the Agile e5 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) in the figure above). The address of the Web File Service is taken from the configuration of the corresponding vault in the Agile e5 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 may be preferable however to use two separate servlet engines.
  Likewise 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 e5 Web Client.
- To avoid a misuse of the Web File Service, the re-directed request contains an authorization ticked 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) in the figure above).

- 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) in the figure above).
- Now the Web File Service can request the FMS Server to spawn an individual thread and return the connection parameter (communication line (4) in the figure above).
- The file data are transmitted between the Web File Service and the FMS Server thread (communication line (5) in the figure above). The Web File Service itself returns the call to the web browser.

Different from earlier releases of the Web Client (namely the axalant2000 Web Client) 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 8080 by default (can be configured during installation)
(2)	http (or https)	Port 8080 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.

Table 8

Chapter 8

Additional Aspects

This chapter provides additional information regarding:

- ❑ Configuration for server-side mailing
- ❑ Business Services
- ❑ Access to Administration Service
- ❑ Access to administration interface for Web Presentation Services

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 (part of the JBoss server).
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, accessible via port 8017 (for http) respectively 8018 (for https). These default ports can be configured.
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.