

Java Application Client Container White Paper



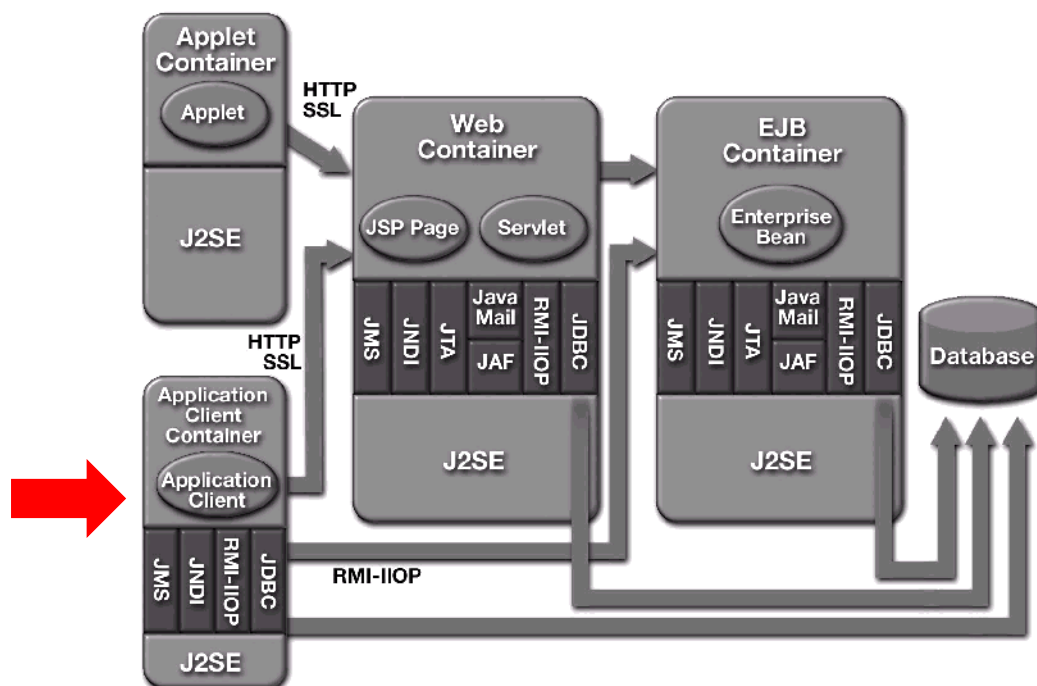
Version 1.2
August 18, 2002

Content

1.	Introduction	3
2.	Challenges of Enterprise Client Applications Development	3
2.1	Programming Productivity	4
2.2	Type of Client	4
2.3	Maintaining Security	4
2.4	Managing Change	4
3.	Requirements and Constraints	5
3.1	Operating Environment	5
4.	The World of the EJB Application Client Containers	6
4.1	Types of Clients	6
4.2	EJB-Client Container: The official definition	7
4.3	Strengths and weaknesses	7
4.4	EJB Client Container: Ergon's point of view	8
5.	Ergon's Java Application Client Container	9
5.1	Platform independence	9
5.2	Architecture	10
5.3	Java Virtual Machine	10
5.4	Communication	10
5.5	JFC/Swing Extensions	11
5.6	Automatic software update	11

1. Introduction

This paper describes an approach to design clients in a multitier enterprise application with the Java™ 2 Platform, using Enterprise Java Bean Containers for the client application. The paper does not contain detailed information on how to implement such applications, but rather focuses on general principles of the running environment of such distributed applications across tiers and choosing among design options within the client tier.



2. Challenges of Enterprise Client Applications Development

Distributed applications are the packages in which an organization delivers information and transactions as a commodity to its customers. In the competitive environment of the information economy, time to market is key to the value of custom applications to the enterprise.

To leverage Internet economics, it's imperative not only to project enterprise systems into various client channels, but to do so repeatedly and in a timely manner, with frequent updates to both information and services. The principal challenge is therefore one of keeping up with the Internet's hyper-competitive pace while maintaining and leveraging the value of existing business systems. In this environment, timeliness is absolutely critical in gaining and maintaining a competitive edge.

2.1 Programming Productivity

The ability to develop and deploy applications is key to success in the information economy. Applications need to go quickly from prototype to production, and to continue evolving even after they are deployed.

Productivity is thus vital to responsive application development. Providing application development teams with standard means to access the services required by multitier applications, and standard ways to support a variety of clients, can contribute to both responsiveness and flexibility.

2.2 Type of Client

Another complicating factor in application development time is the choice of clients. While many applications can be distributed to Web browser clients through static or dynamically generated HTML, others may need to support a specific type of client, or to support several types of clients simultaneously.

These technologies present a diversity of programming models, some based on well-defined standards, others on more ad-hoc standards, and others still on proprietary architectures. Without a model for client applications in an J2EE environment, it can be difficult to reach application requirements effectively and productively.

2.3 Maintaining Security

Traditionally, IT departments have been able to maintain a relatively high level of control over the environment of both servers and clients. When information assets are projected into less-protected environments, it becomes increasingly important to maintain tight security over the most sensitive assets, while allowing seemingly unencumbered access to others.

One of the difficulties in integrating disparate systems is providing a unified security model. Single signon across internal application and beans is important to creating a positive user experience with the applications. Security needs to be compatible with existing mechanisms. In cases where customers need to access secure information, the mechanisms need to maintain high security (and user confidence) while remaining as unobtrusive and transparent as possible.

2.4 Managing Change

How much time and effort do you expend on remodelling and re- building new applications to keep up? The fact is, the constant rework does drastically impact the possibility to attract the customers.

The ability for applications to evolve easily to anticipated or unexpected changes is key to achieving the goals. Systems that require any restructuring or massive redeployment for changes will impede growth and diminish the developer's expected performance.

The systems need to be designed to handle multiple versions of client applications with ease. They need mechanisms for efficient management of system resources and services for software distribution and deployment.

They need to have access to features such as automatic incremental software update without any effort on the part of the application developer. Applications should be able to run on different clients operating systems and hardware.

3. Requirements and Constraints

For every application, there are requirements and expectations that the client must meet, constrained by the environment in which the client needs to operate. Some of the considerations guiding our choice of client are: Is the client intended to be used over the Internet, or within a company intranet? What host platforms must the client work on? For each specific enterprise application, some constraints are more important than others. There are also a number of choices rather than constraints the developer needs to keep in mind.

3.1 Operating Environment

Whether the client will be deployed inside a company intranet or in the Internet determines many aspects of the client. Applications designed for the Internet typically take a lowest-common-denominator approach to the client. In other words, the client must work acceptably over the slowest link and assume only a minimal set of platform capabilities.

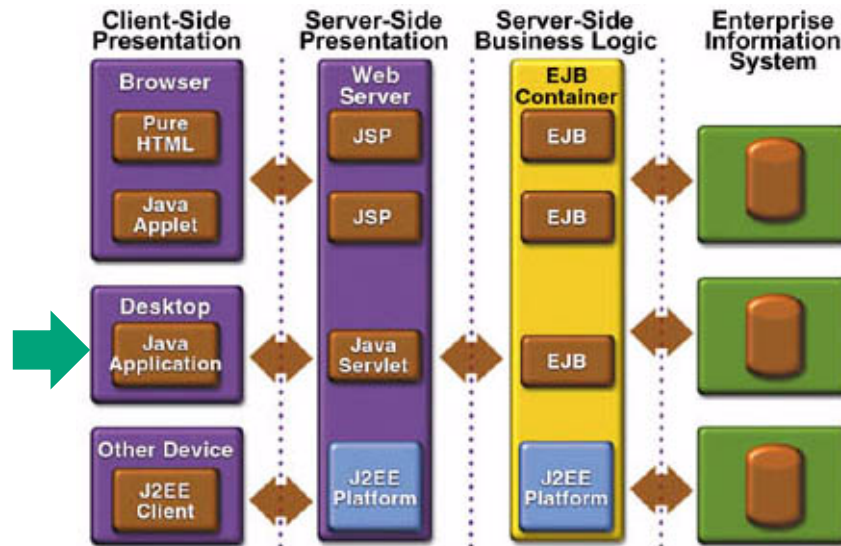
Highly interactive applications place greater demands on network bandwidth and well as latency. Low bandwidth requires settling for a less interactive interface or, it may be necessary to move portions of the presentation responsibilities to the client; this, coupled with some form of caching of the data, could yield acceptable response times. Clients that can take over presentation responsibilities, are better adapted to work in low bandwidth situations.

Clients that work within the intranet usually do not have to worry about fire-walls between the client and the server. However, clients that need to connect over the Internet must be designed to be able to talk to servers that are often behind firewalls. The presence of a firewall limits the possible choices for protocols that the client can use.

Most firewalls are configured to allow HTTP and HTTPS protocols to pass across. Firewalls configured to allow IIOP communications are not widespread. And even when a corporation allows IIOP to pass through, the details of configuring firewalls for this purpose may not be widely understood.

Another aspect is confidentiality. While confidentiality may not be a concern for information within the corporate intranet, confidentiality must be ensured if the communication occurs over the Internet. In this case, its necessary to use a protocol that can ensure confidentiality, such as HTTPS.

4. The World of the EJB Application Client Containers



4.1 Types of Clients

Choices such as which tier or tiers of the application that clients interact with and the protocols used for this communication are dependent on several factors. Different tiers expose different levels of detail and complexity:

- The Web tier presents a simplified functional facade to the application based on its presentation functionality for the client.
- The EJB tier presents an interface to access and manipulate the business objects according to the business rules set in the application, leaving presentation issues to the client connecting to it.
- The enterprise information system tier presents a raw view of the data, delegating the responsibility of enforcing the business rules, as well as presenting the data to the client.

The protocols used for communication have different strengths and limitations:

- The Web tier typically uses HTTP-based protocols. HTML over HTTP is suitable for handling the presentation needs of the client, while XML over HTTP or similar protocols are better suited for interchange of data to be presented by the client. Both can be used over firewalls.
- The EJB tier uses the RMI-IIOP protocol. As a full-scale distributed computing protocol, it gives the client direct access to the services of the EJB tier. A client can use business objects hosted by the EJB server and can participate in transactions.

Many aspects of the client are determined by the tier of the enterprise application the client connects to. The clients can be classified into three broad categories based on the tiers that they interact with:

- Web clients connect to the Web tier. They execute on a desktop, a browser or a browser plug-in. The presentation logic as well as the business logic of the application can run on the server or the client. Web clients use HTTP or HTTPS as the transport protocol.
- EJB clients connect to the EJB tier. These are typically GUI programs that execute on a desktop computer. EJB clients have access to all of the facilities of the J2EE EJB tier. The presentation logic of the application runs on the EJBclient, while the business logic runs on the server. EJB clients interact with the J2EE EJB tier using the RMI-IIOP protocol. A variety of middle-tier services are available to an application client.
- Enterprise information system clients interface directly to an enterprise information system resource. Typically these programs serve administrative and management functions for the backend system. Both presentation and business logic are contained in the client.

4.2 EJB-Client Container: The official definition

A client container is usually a library that is distributed along with the client. It is specific to the J2EE EJB container, and is often provided by the same vendor.

The container manages details of RMI-IIOP communication. It also handles security, transaction, and deployment issues. EJB application clients are packaged in JAR files that include a deployment descriptor similar to other J2EE application components. The deployment descriptor describes the enterprise beans and external resources referenced by the application.

The client application must be authenticated to access the J2EE middle tier. Techniques for authentication are provided by the client container, and are not under the control of the application client. The container can integrate with the platform's authentication system, authenticate when the application is started, or use some other lazy authentication policy. The container takes responsibility for gathering authentication data from the user by presenting a login window on the screen.

4.3 Strengths and weaknesses

EJB clients have a number of strengths, including:

- **Provide a more flexible user interface**
Application clients can be made to look and feel more like native applications on the client machine. Since these clients implement their own user interface, they can provide a richer, more natural interface to the application tasks.
- **Distribute the workload**
An application client can share some of the computational expense by doing the task on the client desktop, and thereby reducing load on the server. In particular, the work of generating the user interface is performed by each client. This is useful for applications with specific graphical display capabilities.
- **Handle complex data models**
Sometimes the data associated with an application is sufficiently complex and the manipulation interface rich enough, that a Web-based interface to manage the in-

teraction is not enough. In such cases, you want direct access to the underlying object model on the client and to manipulate it directly.

- **Are transaction-capable**
Since EJB clients communicate using RMI-IIOP, they are capable of participating in client-demarcated transactions through JTA APIs.
- **Provide integrated security**
Application client containers can integrate with the security of the underlying operating system where the client is executed, thereby providing a more transparent and manageable security infrastructure.

The disadvantages of EJB clients are that they:

- **Require explicit deployment**
EJB clients need to be distributed and installed on the client desktops. In an intranet, where desktops can be standardized, this is less of an issue. However, on the Internet, distribution becomes a serious consideration. Furthermore, upgrades and fixes to the client need to be distributed as well, and the server has to deal with multiple versions of the client programs.
- **Require firewall reconfiguration**
The RMI-IIOP protocol does not usually go through firewalls without additional set-up on the firewall host. This makes use of EJB clients on the Internet very limited.
- **Tend to be more complex**
Since the application client needs to manage its own user interface, its implementation is more complex. Furthermore, it communicates with the J2EE server at a much lower level than browser-based Web clients and needs to handle the complexity introduced as a result.

4.4 EJB Client Container: Ergon's point of view

Ergon's Java Application Client Container solves most of the disadvantages of the "official" EJB-Client in supporting:

- Web based deployment and automatic updates
- HTTP over SSL for passing firewalls
- Environment for Java Beans based on a common framework
- Connecting to application servers not supporting IIOP

5. Ergon's Java Application Client Container

Ergon's Java application client container offers the flexibility needed to run powerful Java applications on the Internet. By installing the container, problems with different Browser versions, different Virtual Machines, or underlying operating systems can be avoided. Its strong built in security allows to develop information sensitive applications.

By bringing the presentation logic in form of small Java beans back to the client, applications perform fast even over slow Internet connections. An application can be extended with plugins, implemented by several Java beans.

The integrated upgrade possibilities allows to upgrade the application and its plugins on the fly class by class from a trusted server. The underlying technology has been proven within a growing number of business projects like internet banking, discount brokerage and messaging applications.

Once the Java application client container is installed, the user no longer has to care about getting new functions or technology updates. He will automatically be prompted if he wants to accept a new feature or an new version. The different communication protocols supported allows to adapt a bean to many application server. HTTP over SSL can be used over firewalls without additional configuration of the firewall. To ensure the integrity of the data, the communication can be encrypted using 128-bit SSL.

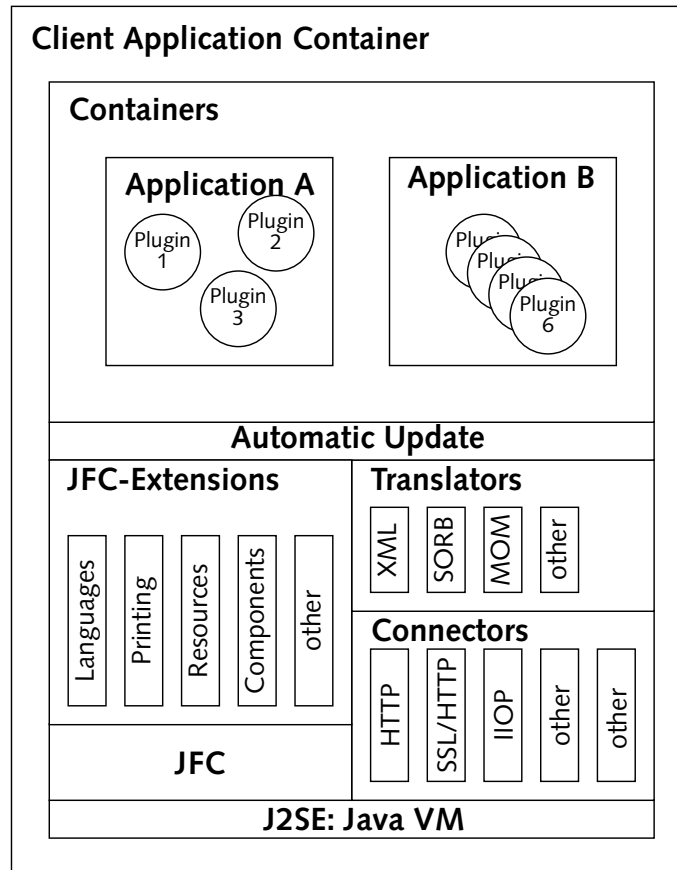
5.1 Platform independence

Running EJB-Clients as an applet in a browser needs a lot of maintenance effort: Different browsers, different versions and different Java environments absorb a lot of development capacity.

The Java application client container gives a well defined environment for running any type of applications. The architecture is designed to run the environment on most of the platforms supporting a virtual machine (VM). Therefore, only one application needs to be built and stored on your application server.

For mainstream clients like PC and Mac, native packages can be built to make the installation on the client as simple as possible.

5.2 Architecture



5.3 Java Virtual Machine

The standard Sun VM is used to run the client. However, it has been stripped down to the minimum needed to run Java applications fast and kept the code to the minimum.

5.4 Communication

Communication to other application is implemented using Connectors and Translators. Components which support CORBA-IIOP, XML, TIB-RV can be plugged into the container for additional communication protocols.

- Connectors enable the communication to other tiers. Different protocols (HTTP, HTTP over SSL, IIOP, RMI, JMS etc.) are supported and can be extended by loading own connectors into the container.
- Translators allow to handle different underlying communication data formats like XML, SORB, MOM and any other by loading the corresponding components.

SORB (Simple Object Request Broker) is used as the a communication Layer to hide the underlying communication mechanism. It implements a simple object model, which allows to send and receive self describing messages. It implements the basic datatypes

integer, string, float and the extended data type associative array, byte arrays, files and binary data.

5.5 JFC/Swing Extensions

JFC (Swing) is used to build the user interface components. Additional support for printing and international languages is added in the extensions to the JFC. The international language extension allows the user to dynamically change the language at runtime.

5.6 Automatic software update

To support clients used by retail customers, upgrades to new features or bug fixes are a major challenge. The Java application client container implements a simple and reliable update mechanism. As soon as the customer has installed the application container from a CD or from the internet, it updates and downloads additional applications and its plugins from the corresponding trusted application server automatically when it starts.