SCHWEIZER BÖRSE SWISS EXCHANGE
BOURSE SUISSE BORSA SVIZZERA

Swiss Exchange

Elektronischer Repo Markt (ERM) - Java Technology Feasibility Study

I-ERM-JFS-100B/E, Draft Version 1.0B, 15 May 98

This document describes the Java technology study for the ERM project. The goal is to examine the advantages and risks of using an architecture based on Java middleware, and position them to the technologies in place (Microsoft NT server, DEC USP).

Identification

Title:	Elektronischer Repo Markt (ERM) - Java Technology Feasibility Study
Version:	Draft Version 1.0B, 15 May 98
Classification:	For Internal Use Only
Intended Audience:	ERM project management, GBI responsible.
Keywords:	
Reference:	I-ERM-JFS-100B/E
Filename:	\\bzh000\s\projekte\117_erm\jfs\jfs100be.doc/REB
Synopsis:	This document describes the Java technology study for the ERM project. The goal is to examine the advantages and risks of using an architecture based on Java middleware, and position them to the technologies in place (Microsoft NT server, DEC USP).

Author(s):	....	Blaise Rey-Mermet
Reviewer(s):	....	TRN, FGO, MUE, WEH
Approval:	....	SPI
Distribution:		

Revision History

Version, Date	*Change Description*
Draft Version 1.0B, 15 May 98	Draft Version - Strategy proposal, no content.
Draft Version 1.0, 8 Apr 98	text update of non-functional requirements and architecture overview.
Draft Version 0.0A, 8 Apr 98	New document

Table of Contents

1. Introduction

1.1 Purpose of the Document

This document describes the Java technology study that is performed during the project set-up phase. The document examines the advantages and risks of using an architecture based on Java technologies and middleware, and compared them to the technologies in place (Microsoft NT server, DEC USP). The goal is to assess the feasibility of using Java middleware and technologies to develop the ERM system.

1.2 Study Objectives and Strategies

The strategy followed to perform this evaluation consists of:

- Describing a possible technical solution. This means a description of the architecture, a selection of middleware products to realise it and a evaluation of potential advantages and risks.
- Finding a demonstration site. We should try to benefit from the experiences of projects with similar performance issues.
- Building a GUI prototype to evaluate the GUI capabilities.
- Building a gateway prototype for performance and load testing.

1.3 Changes Since Last Version

This is the first version of this document.

1.4 Structure of this Document

section 1	introduces the document and describes the purpose, scope, structure and relationship to other documents. It contains references and abbreviations.
section 2	provides a management summary, conclusions and recommendations.
section 3	provides an overview of the ERM technical architecture and introduce the main technical concepts.
section 6 and 7	describes a solution based on Java and a NT-based solution and how they fulfil the technical requirements.
Appendix A	describes the non-technical requirements specific to the company and development team. This is an extension of the product-specific non-technical requirements described in 🗎4 Non-Functional Requirements for ERM.

1.5 Relationship to Other Documents

This document lists technical requirements and is therefore strongly dependent on the business and non-functional requirements (see 🗎3 and 🗎4).

1.6 Definitions & Abbreviations

Term	***Meaning***
authentication	The verification of the identity of a person or process. In a communication system, authentication verifies that messages really come from their stated source, like the signature on a (paper) letter.
CORBA	Common Object Request Broker
DCOM	Distributed Component Object Model
EJB	Enterprise JavaBeans™
ES	Exchange System
HTTPS	Secure hypertext transfer protocol
IIOP	Internet Inter-Orb Protocol
ISI	Interoperability Service Interface
JDBC	Java™ Database Connection
JDK	Java™ Development Kit
JRE	Java™ Runtime Environment
JIT	Just-in-time compiler
JMAPI	Java Management API (JMAPI)
JMS	Java Messaging Services
JNI	Java™ Native Interface
JTS	Java™ Transaction Service API
MAPI	Member Application Programming Interface
NTB	Network Transaction Bus
RMI	Remote Method Invocation
SSL	Secure Sockets Layer (SSL). An encryption mechanism for remote Web server administration. http://java.sun.com/security/ssl/API_users_guide.html
TS	Trading System
USP	Universal Server Control Process, a request broker, providing fundamental services for distributed applications (Digital Equipment Corp.)

1.7 References

reference & document title	***applicable version and reference***
🗎1 ERM Systemkonzept	I-DEV-ERM-100D/D
🗎2 ERM Business Plan	I-ERM-JFS-100B/E
🗎3 Functional Requirements for ERM	I-ERM-FRQ-100A/E
🗎4 Non-Functional Requirements for ERM	I-

1.8 Outstanding Issues

a. Discussion of security APIs an products to provide authentication, authorisation and integrity.

b. For the EJB solution, select an EJB implementations (Weblogic Tengah, Gemstone/J, Oracle Application Server, BEA Iceberg).

c. Better describe ISI solution

d. Add conclusion

2. Management Summary and Recommendations

Java has been used a lot in GUI Applications so far and is now entering into the server area. For the server side, it is a very difficult matter to find expertise in this area and even if the technology is ready the resource know how can be a show stopper for such an implementation approach.

The main benefit in using this language is the smooth transition form one operating system to another. It also gives SWX a chance to start the transition phase from one paradigm to another paradigm.

Conclusions *[text]*

Recommendations *[text]*

Summary table: the following table is an overview of the discussed solutions and a general rating of the corresponding risk level.

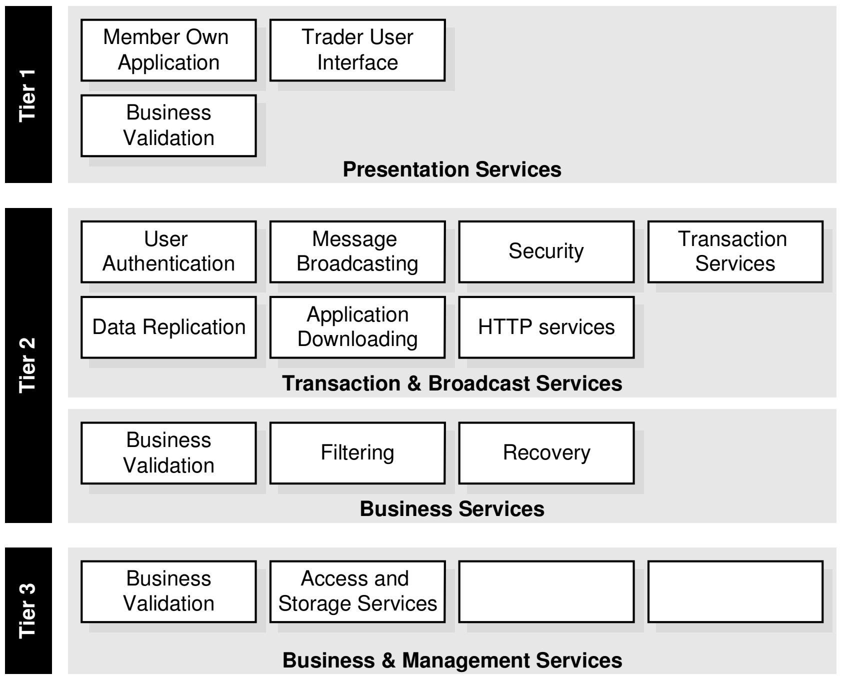
Solution	**Risk level (1:low, 5:high)**	**Cost to adapt to future techs (1:low, 5:high)**	**Short Comments**
1) Current USP and OpenVMS server solutîon	2	5	This solution use a Java-based client, connected to a server based on C, USP and VMS technologies. The main risk is to miss the delivery deadlines due to the large amount of middleware code to write.
2) Java Server with ISI Integration Framework	3	1.5	This is an transition solution, using a proven distributed open architecture, but not the new generic Java APIs (EJB). The migration to the EJB solution is possible.
3) Java Enterprise Server platform (EJB) solution	5	1.5	This is the most advanced solution, providing interoperability and portability, based on open, generic specifications (EJB). The availability of mature products implementing the EJB specifications is not granted.
4) Microsoft Windows NT server solution	No rating	No rating	The main risk is the absence of proven, large banking server solutions using NT-based technologies (scaleability).

3. System Description and Architecture

3.1 Simplified architecture overview

3-tiered architecture

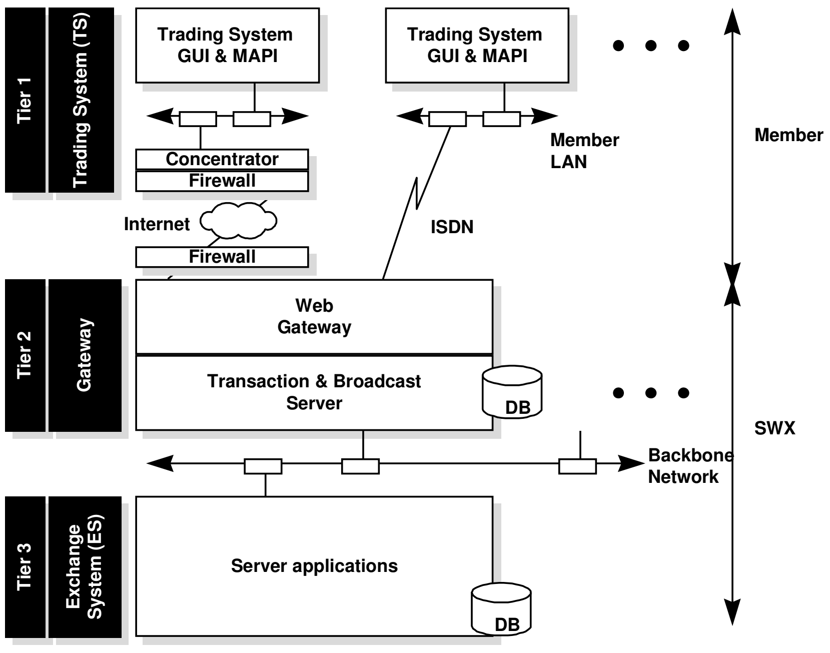
3-tiered architectures represent today's general paradigm for Web distributed architectures. A 3-tiered architectures enable the presentation, business logic and data elements of application to be cleanly separated and run on different machines connected by a network. The next figure shows a 3-tiered Internet distributed architecture.



ERM 3-tiered architecture (logical view)

ERM architecture

We can describe the ERM architecture as a 3-tiered system. The background tiers (trading system, interfaces to SEGA and SECOM etc) are omitted for simplification purposes.



Distributed architecture for the ERM system(physical view)

The ERM architecture is described in more details in 🗎1.

The Trading Systems (TS) (GUI & MAPI) main goal is to interface with the traders, I.e. entering, modifying, and display of orders and trades. The trading systems are connected through the local network of the member to the WEB Gateway, either through the Internet or a dedicated SWX ISDN line.

The Concentrator collects and distributes the services to the TS clients.

The WEB Gateway provides distribution to the members trough Internet and ISDN (Web Server) as well *transaction and broadcast services*. The Gateway provides some business functionality related to broadcasting and transaction services, such as system recovery.

The Exchange System (ES) provides normal handling of orders and trades (matching). The Exchange System provides also services such as the transmission of trades to SEGA, the control and supervision of the exchange, and archiving services.

3.2 Transaction model overview

The goal of this chapter is to introduce the main concepts related to the transaction model, to serve as background information for the performance and load requirements.

Definition of terms:

Messages

All information crossing the ES – TS interface is in the form of a "message".

A message is a functionally indivisible unit of data. That is, the information within a message is all functionally related to the same event. For example, any trades resulting from matching upon entry of one order entered to the system are reported in one message with a description of each trade.

Transactions

A transaction across the interface is a sequence of messages which together perform a given action and report the results of that action. Like a computer transaction, the transaction either happens or does not happen – it cannot half-happen – and all constituent messages of the transaction are self consistent and relate to a single state within the exchange.

A few examples of transactions are: enter an order, delete an order from the exchange order book, indicate a statement of interest to all trading members in dealing a particular security, etc..

Example: Enter Order transaction (extracted from 🗎4):
i Member A enters an order and gets, after validation, a reply from the Exchange System.
ii The incoming order results in no changes to the public orderbook ie, it was an order of type Fill Or Kill which was killed or an order of type Accept which found no qualifying counter orders.
iii In all other cases, there are changes to the public orderbook caused by the incoming order. The changes are broadcast to all members.
then: (a) The security is not trading or is trading and no matching occurs. No trades are sent to the members.
(b) The security is trading and a partial matching occurs. The trades are broadcast to all members.
(c) The security is trading and a full matching (including Fill Or Kills which fill and Accepts which finds qualifying counter orders) occurs. Member A gets a message containing the order result. The trades are broadcast to all members.

Transaction Size A transaction generates an average of 60 messages of 100 BYTE/message

4. Current USP and VMS server solution

4.1 General Description

This solution is to connect a Java-based client to a gateway implemented with the technologies in place at SWX. The main integration component is the Universal Server Control Process (USP), a request broker, providing fundamental services for distributed applications (Digital Equipment Corp.)

Reference: USP description is available from Digital Equipment Corp.

The description of this solution is out of the scope of this document. A list of risks and benefits is provided for positioning purposes.

4.2 Benefits

This lists the most important business themes for this specific project:

- **No learning curve.** The architecture is based on available skills.
- **Mature**, proven technologies.

4.3 Risks

- **Long development time**. All new services interfaces has to be implemented in-house.
- **No possible migration.** There is no migration path from a USP based architecture to an open IIOP or EJB-based solution. The system has to be redesigned and rewritten to inter-operate with future systems.
- Low-tech: The development of in-house expertise in the field of Internet technologies will be delayed for many years. The capacity of the company to adapt to new technological challenges is compromised.

  The engineers willing to gain new skills may leave.
- Requires the presence of proprietary technologies (USP bridge) at the customer WEB server side.

5. Java Server with ISI Integration Framework

5.1 General Description

The Interoperability Service Interface (ISI) is a CORBA 2.0-based multi-platform object request broker implementation designed for large-scale enterprise computing. ISI has been used large Internet-banking application, where the scaleability of standard ORBs was not sufficient. It is developed by AdNovum AG.

ISI is an open integration framework, supporting the integration of many ORBs (such as the IIOP-ORB or the TUXEDO-ORB) and many language-mapping, including C, C++ and Java.

Reference: a more complete description can be found in *http://www.adnovum.ch/ISI/isi.html*

Service	**benefit**	**..provided by**
Component integration	ease of use, scaleability	Interoperability Service Interface (ISI)
Distributed Object Interface	interoperability through heterogeneous systems	IIOP
Database Access Services		*[text]*
Management Services	network manageability	*[text]*
Security Services		based on RADIUS authentication server
Messaging and Broadcasting Services		*[text]*
Transaction Services		TUXEDO, OMG Object Transaction Service
Web Services		ISIWEB

Server-side Distributed Object Interface: Internet Inter-ORB Protocol (IIOP)

Technical description

The Internet Inter-ORB Protocol (IIOP) provides interoperability by specifying how objects exchange messages and addresses in a distributed system. The IIOP is basically TCP/IP with some CORBA-defined message exchanges that serves as a common backbone protocol.

Implementation Issues

- A **NTB-IIOP bridge** has to be implemented on the interface between the Exchange System (ES) and the gateway. The bridge is responsible to translate the services of the host (Exchange System) bus, in this case the Network Transaction Bus (NTB) to IIOP. This development is simplified by a a generic host access API.

Benefits

- In the case of mixed environment such as Java-to-CORBA objects, IIOP is preferred for **portability** and **interoperability** across platforms and languages. IIOP allows Java to act as CORBA objects, providing easy integration between CORBA and Java objects.

- Proven generic solution to implement a **NTB-IIOP bridge** exits**.** Adnovum has experience in the integration of such host interfaces. The ISI generic host access API reduces this development effort.

Database Access Services: *[Product]*

Technical description	*[text]*

Management Services : *[Product]*

Technical description	*[text]*

Security Services:

Technical description

ISI offers security service with authentication and encryption support :

- a strong Smartcard solution based on IDEA, without any export restrictions.
- integration of existing security technology
- optimised manageability of security services
- A RADIUS authentication server is required to for user authentication

Messaging and Broadcasting Services: *[Product]*

Technical description	*[text]*

Transaction Service: TUXEDO, OMG Object Transaction Service

Technical description

The ISI framework offers a CORBA wrapper around TUXEDO designed as a so-called ISI-TUXEDO-ORB. The TUXEDO-ORB uses TUXEDO transport services for communication purposes, enhancing the CORBA-compliant ORB with TUXEDO features like load-balancing and transaction management. Support for transactions is provided by an implementation of the OMG Object Transaction Service Interface.

See also http://www.beasys.com/products/tuxedo (BEA's Tuxedo)

Risks

- Availability of TUXEDO on OpenVMS

Web services: *[Product]*

Technical description	*[text]*

5.2 Benefits

- **Interoperability with open client-side technologies**. This solution allows the customer to integrate any IIOP-compliant client, including DCOM.
- **High scaleability, high performance**: This is the most scaleable architecture to date. ISI has proved a proved a key middleware component in large-scale banking environment s (600 servers).
- **Migration**: IIOP is a significant step toward an open distributed system. It assures a smooth migration to a full EJB compliant architecture as a future step.
- **Mature, proven solution**. The Java language and specification are mature and proven technologies. The ISI integration framework is in use with high-performance, large Internet banking applications.
- **Availability**: ISI is available on multi-platform including Sun Solaris 2.5.x, Windows NT 4.0 and HP-UX, however Solaris is the only proposed platform for high-performance large server systems.

  An OpenVMS implementation is in propriety of BEA Systems. BEA has no declared goals to maintain and support this version, in particular the CORBA 2.0 port is missing.
- **Faster product release**: Low development costs: the amount of proprietary development is small, allowing a fast product release.

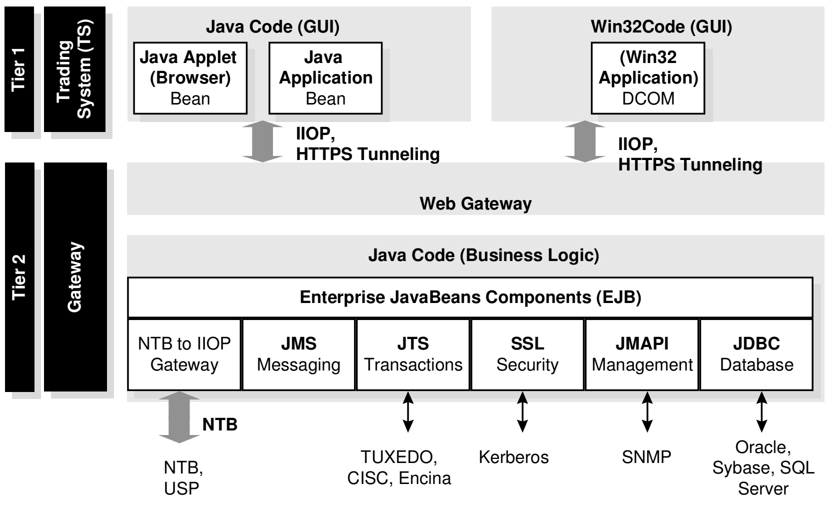
5.3 Risks

- This is **not a generic approach**. The interfaces to services are written using IIOP and IDL interfaces. These interfaces may be rewritten in a future release using higher-level Java APIs in order to get better portability and code reuse (longer development time for next product releases).
- **Complexity**: If not avoidable, the development of CORBA interfaces in IDL is complex. This is however well supported by tools such as design tool generating IDL code from the design models (for example Rational Rose).

6. Java Enterprise Server solution

6.1 General Description

The Java Enterprise Server platform is a standard set of application programming interfaces (APIs) which access a core set of underlying enterprise-class infrastructure services, regardless of actual implementations. Because it is a new set of specifications from Sun Microsystems, the availability of commercial products implementing these APIs is diverse, depending on the API and the platform. The Java Enterprise Server platform also provides a component model for application encapsulation and reuse. The model includes JavaBeans, the component model for Java on the client, and Enterprise JavaBeans, on the server.



Sun Entreprise Java Beans
Source: Sun Microsystems

Service	benefit	..provided by
Component integration	ease of use, scalability	Java Beans, Enterprise Java Beans
Distributed Object Interface	interoperability through heterogeneous systems	Java Remote Method Invocation (RMI) on IIOP
Database Access Services		Java Database Connectivity (JDBC)
Management Services	network manageability	SNMP, Java Management API (JMAPI)
Security Services		SSL, Kerberos, Java security model (Java language)
Messaging and Broadcasting Services		Java Messages Service API (JMS)
Transaction Services		Java Transaction Service API (JTS)
Web Services		Java Server API

source: Java Adoption White Paper, [Sun98a]

Client UI Framework: Java™ Development Kit (JDK) Version 1.2

Technical description

Not described here: Java Interfaces based on JDK has been the subject of a prototype implementation.

Risks

- **Portability** across platform is related to the complexity of the interface. Binary compatibility is provided by the Java Virtual Machine (JVM). Consistency with specific UI is difficult to achieve for complex interfaces.
- **Integration with Microsoft Office** tools is no simple or not flexible.

Benefits

- **Performance**: JDK 1.2 and the previous maintenance release (1.1.6) are targeting performance improvement as one of the strategic success factor. The JDK 1.2 performance improvement features are in particular:
  - JIT compiler,
  - faster memory allocation,
  - improved garbage collection,
  - JNI conversion,
  - class compression in memory, and
  - serialisation optimisation
- **Interoperability** with CORBA, (JavaIDL and IIOP)
- **Portability** across browser. Applets can run within the context of a browser or as standalone application.
- **High-level of functionality** on the client-side. High level of integration with Internet mechanisms.
- Easy distribution: **Dynamic Class Download** (applets). Typical systems that provide dynamic download are Java based, since binary incompatibilities are not an issue in Java.

Client-side Component integration: JavaBeans

Technical description

JavaBeans are Sun component architecture for the Java platform. This is the mechanism to integrate the different APIs on heterogeneous platforms. The component architecture promotes reuse and enables developers to write one and run them anywhere. Java Beans are Sun's answer to Microsoft's ActiveX.

Implementation Issues

Portability: Generally, a Java Bean is dependent on the Web Browser, it doesn't care about the platform because the browser hides the details of the platform. To integrate a Java Bean, the browser must support JDK 1.1 which currently, Netscape Communicator 4.0 and Internet Explorer 4.0 tout they do or will do.

Risks

- **Maturity**: Java Beans are very new technology thus immature.

Benefits

- **Simplicity**: Java Beans are designed to be easy to implement. To connect up some beans in an application, little software development is involved.

Server-Side Component integration: Enterprise JavaBeans (EJB)

Technical description

EJB, is an adaptation of Java's component model for databases and transaction and application servers. Each of these can function as a container for EJB components, which are intended to be portable between servers and operating systems. It compares to the Microsoft Transaction Server (MTS) for the ability to insulate business developers from the complexity of distributed transaction systems. EJB is an extension of the Java Beans component for the server, the main difference between them is the handling of events.

Discussed version: Enterprise JavaBeans 1.0 specification, March 1998
References: [Sun98a]

Implementation Issues

There are different implementations of the EJB model to date. This are the products mentioned by Sun:

WebLogic Tengah

Tengah is a Java application server, and is based on an open, extensible, standards-based platform for deploying, and managing distributed Java applications. References: *http://www.weblogic.com/*

Benefits

- Is a complete implementation of EJB. All APIs support Java.
- Good **tool support**: Tengah tools supports development and monitoring of server application. For example a graphical management console shows a real-time view of the distributed application environment.
- Available: the Tengha Web Server 3.0 is released.

Risks

- WebLogic is not a key provider of middleware solutions.

Gemstone/J by GemStone Systems, Inc.

This container requires the Visigenics ORB as integration technology. Andersen Consulting and Gemstone are developing an information system for the Banque Generale du Luxembourg using EJBs Gemstone/J and can serve as reference site. Reference: *http://www.gemstone.com/*

Benefits

- Is also a complete implementation of EJB. All APIs support Java.
- Available: GemStone/J 1.1 has been released in March 98.

BEA Iceberg

The Enterprise Object Middleware product (code-named "Iceberg") will support either CORBA-based business objects for BEA TUXEDO services and enable them to inter-operate.

Reference: *http://www.beasys.com/*

Risks

- Not available: Iceberg release is planned for mid 98.
- Is targeting TUXEDO customers wanting to migrate to objects.

ORACLE Application Server 4.0

Oracle Application Server 4.0 (OAS) us built around a CORBA 2.0 compliant ORB and the IIOP protocol. The OAS 4. 0 architecture is composed of components called *Cartridges*: Cartridges are manageable components that plug into the Oracle Application Server. A cartridge can have multiple instances to handle different requests. This allows sophisticated load-balancing for these cartridges and their instances.
References: *http://www.olab.com/*

Oracle Application Server implements the application cartridges and all system services as distributed objects integrated with IIOP.

Benefits

- Elegant, scaleable architecture.

Risks

- Not EJB Compliant
- Availability: Not available on OpenVMS. Supported platforms are Windows NT 4.0, Solaris 2.5.1 and HPUX 11. OAS 4.0 is only available in beta version at this time.

Distributed Object Interface: Internet Inter-ORB Protocol (IIOP)

Technical description

This is described in the previous solution.

Database Access Services: Java Database Connectivity (JDBC)

Technical description

The Java Data Base Connectivity (JDBC) API defines Java classes to represent database connections, SQL statements, result sets, database metadata, and RPCs, etc.

JDBC provides Java platform programmers with a uniform interface to a wide range of relational databases. It is a call-level, SQL-based interface, which provides database independence for Java applications. The JDBC API is implemented on top of the Java SQL Driver API, which in turn is supported by

a variety of pluggable database driver modules.
(Extract, see longer description in [Sun98a])
See also http://java.sun.com/products/jdbc

Implementation Issues

JDBC is included in JDK 1.1.

There are several JDBC drivers available, including from Oracle.

Benefits

- Provides **database independence** for Java applications.
- **Simple**, Standard interface to integrate databases.

Management Services: Java Management API (JMAPI)

Technical description

The Java Management API (JMAPI) provides guidelines, Java classes, and specifications for developing seamlessly integrated system, network, and service management applications. These specification are not complete, and therefore not further described here.

References: see *http://java.sun.com/products/JavaManagement/* (for JMAPI)

Risks

- **Availability***:* not available yet, pre-release version.
- No supporting products known (?)

Security Services: SSL, HTTPS

Technical description

[text: the API to provide the required security features are not yet discussed.]

Implementation Issues

- Secure hypertext transfer protocol (HTTPS).
- **HTTP firewall tunneling:** Tunneling of HTTPS through corporate firewalls to Internet web servers of the client.

Messaging and Broadcasting Services: Java Messages Service API (JMS)

Technical description

Java Message Service (JMS) API will allow Java applications to access disparate messaging services and messaged oriented middleware (MOM) products. These specifications are not completed, so they will not be further described here.

Risks

- **Availability***:* this is a future product, the release dates are not known.

Transaction Service: Java Transaction Service API (JTS)

Technical description

Java Transaction Service (JTS), a Java platform API, is the Java binding to Object Transaction Service (OTS). OTS is OMG’s CORBA-based standard distributed transaction manager API. JTS is a low-level API intended for resource managers and TP monitor programmers.

Products such as BEA’s Jolt Gateway for Java provides Java-based access to Tuxedo and CICS transactional services, respectively, and are available today.

References:
http://www.visigenic.com/prod/tpb (Visigenic’s TPBroker)
http://www.software.ibm/ts/cics/platforms/clients (IBM CICS Clients)
http://www.beasys.com/products/tuxedo (BEA’s Tuxedo)

Benefits

- Simple, Standard interface to integrate different TP products.

6.2 Benefits

This lists the most important business themes for this specific project:

- **Protecting Future IT Investment** - Java is a portable environment that will help avoid costly future IT lock in. Because it is platform independent, Java breaks the previously fixed link between the application and the operating system.
- **Portability:** The Java platform replaces the dependency from the platform by a dependency to a set high abstraction level APIs, this allowing programs running on one server to be moved to a different server without rewriting the infrastructure code.
- **High interoperability**: More important, these Java programs can interoperate with the underlying infrastructure services on different platforms without code rewrite. The EJB architecture allows the easy integration of containers supporting new interfaces required by future developments.
- **Widespread industry support**: Java is next paradigm for distributed systems. The availability of tools and technologies supporting Java is growing rapidly. This concerns also distributed object protocols such as Java support for CORBA (the Common Object Request Broker Architecture). The key players including Oracle, Sun and Microsoft are providing products targeting the distributed transaction market.
- **Provide a high level of functionality at the front end.**
  The interface to the transaction server is provided by an unified object interface. This simplifies the integration of customer applications on the client side.
- **Simplicity**. All interfaces are supported by high-level standardised APIs written in Java (EJB). This has a large impact of the development time of the next product releases, and the future projects development times.

  The presence of an efficient Java garbage collector on the server-side simplifies the product development and maintenance.

6.3 Risks

- **Performance**: The inability of Java to support native features on all platforms brings up some performance issues. The Java portability is provided by the presence of a Java Virtual Machine (JVM), an code interpreter. Compared to native code, the performance handicap is claimed to be small, but should be quantified by a performance and load test.
- **Maturity:** Even if, Gartner Group research director David Smith cautions that vendor like IBM and Oracle are unlikely to deliver significant EJB support until late this year, there are many EJB implementations available to date. However, these products are new. The EJB specifications are technically mature but the product implementing them are not. As far as we know they are no large-size critical banking systems on the Internet using EJB to date.

  Few of the mentioned products are available on OpenVMS.
- The use of Enterprise Java Beans as middleware is dependent of the presence of **project-specific business logic written in Java**. This is a potential design goal for the new developments.

7. Microsoft Windows NT server solution

7.1 General Description

This platform is based on the WindowsNT™4.0 architecture and services to provide both the gateway- and transaction system function.

It is not in the scope of this evaluation to describe this platform. This chapter provides an brief overview and a list of risks and benefits for positioning purposes.

Service	**benefit**	**..provided by**
Component integration	ease of use	COM+, C++
Distributed Object Interface		DCOM
Database Access Services		ODBC
Management Services	network manageability	*[text]*
Security Services		*[text]*
Messaging and Broadcasting Services		*[text]*
Transaction Services		Microsoft Transaction Server 2.0 (MTS)
Web Services		IIS 4.0

7.2 Benefits

This lists the most important business themes for this specific project:

- **Availability**: DCOM is a Microsoft monopoly (at least on the PC world), this assures the availability on the Windows platform. Recently, third party vendors such as Digital Equipment Corporation begin to port DCOM to other operating systems, including multiple implementations of UNIX, although these products have little chances to mature.

  The Microsoft platform assures also an excellent tool coverage. There is a consistent set of development and support tools on both the client and server side.

7.3 Risks

- Impose the presence of a Microsoft DCOM ORB at the client-side.
- **Tunneling through Firewall.** For the application relying entirely on TCP/IP layers, a Firewall could pose serious problems. Some solutions to this problem have been implemented such as Tunneling in HTTP or SOCKS servers.
- **Maturity**: CORBA implementations exist on most platforms, and have been used successfully in some large projects long before DCOM appeared on the market. As far as we know they are no large critical banking systems on the Internet using DCOM to date.
- Performance of DCOM and Object Registry. This is a feature of the name service to register objects on each machine. While CORBA supports a

registry with ongoing persistence, DCOM utilises the NT's registry for finding objects/interfaces by their GUID.

- Dynamic Class Download. Typical systems that provide dynamic download are Java based, since binary incompatibilities are not an issue in Java.

Appendix A: Corporate- and development team technical requirements

The project and choice of technologies is obviously also shaped by constraints from the team size, the available time and the team goals. This chapter lists such requirements which are not related to the product, and therefore not part of 🗎3 Functional Requirements for ERM.

Team requirements:

Ref	Description
1	the project provides the team with technical, in-house experience in the field of WEB technologies. Therefore the technologies selected should be at least partly applicable to forthcoming products.
2	the project is to be realised with reduced resources and development time (see 🗎2 ERM Business Plan) This implies that underlying non-application specific middleware will be purchased and not developed in house in order to avoid complex time expensive implementations (use off-the-shelf technologies)
3	the costs of integrating the diverse technologies are minimised. This could be achieved in particular by the selection of a single provider (vendor) with integrated solutions.
4	minimised complexity: a state-of-the-art development environment is available
5	minimised complexity: the GUI Components are layered on a high-level abstraction layer (class library)
6	minimised complexity: an high-abstraction level interface is provided to access the middleware (for example the ORB interface). The middleware is transparent to the application.

Support and service requirements :

Ref	Description
7	the vendor services (training, consulting, specialists) are available in Switzerland and also in-house if necessary
8	the vendor has an efficient support channel.
9	the vendor is dedicated to the product and capable of providing long-term support. The vendor long-term strategy for the product is clear.
10	the vendor can offer consultants.
11	the vendor can provide one or more reference sites. SWX may contact this reference site and arrange on-site visits if thought necessary.
12	support channel has established goals and metrics for support ranging from: critical (show stopper) problems that are addresses within one hour (?) to minor problems and suggestions that are addressed by the next release.
13	Available pool of resources skilled in the use of the product for outsourcing.

The product commercial requirements are:

Ref	Description
14	product availability: the product is available now (main release)
15	the pricing and licensing policy is defined and within budget

8. References and links

Market Research and reports about Middleware

- [Gartner01]A.Perc - *Server Java's Identity Crisis: Just Another VAM Activity?* GartnerGroup ResearchNote 27 Feb 1998 SPA-03-6532
- [Ovum98] Rock-Evans, Rosemary - Middleware and the Internet
http://www.ovum.com/evaluate/mdi/mdi000.html
Available March 1998 - Ovum Ltd.
- [Seybold97] Thomas, Anne - Enterprise JavaBeans™ - Server Component Model for Java
Patricia Seybold Group © 1997 - December 1997

Technology white papers

- [Sun98a] *The Java TM Enterprise Server Platform - A Java Adoption White Paper for Developers*. Sun Microsystems, Inc. - November 25, 1997 http://www.sun.com:80/javasystems/whitepapers/ServerAPI.pdf
- [Berdan97] Berdan, Mihai - *DCOM White Paper* - http://objectworks-usa.com/papers/dcom.htm, ObjectWorks Consulting Inc. - July 1997