

Vision JMSME

Piotr Anders, Paulina Kania, Łukasz Osipiuk, Paweł Ostrowski

June 15, 2003

Contents

1	Introduction	2
1.1	Purpose	2
1.2	Scope	3
1.3	Definitions	3
1.4	Attachments	4
1.5	Document Summary	4
2	Product Context	4
2.1	Benefits	4
2.2	Problem Introduction	4
2.3	Product Context	5
3	Stakeholders	5
3.1	Market Demographics	5
3.2	List of the stakeholders	6
3.3	List of identified users	6
3.4	Environment of target user	7
3.5	Stakeholders description	7
3.5.1	Project Coordinator	7
3.5.2	JMSME programmer	8
3.6	Users description	9
3.6.1	Programmer	9
3.6.2	Administrator	9
3.7	JMSME Application user	10
3.8	Alternatives and competitors	10
3.8.1	kJoram	10
3.8.2	iBus//Mobile from Softwired Inc.	10
3.8.3	WebSphere MQ Everyplace V2.0 from IBM	11
3.8.4	Jtom	11

4	Product description	12
4.1	Product perspective	12
4.2	Product functions	12
4.3	Assumptions and dependences	13
4.4	Costs	13
4.5	Licensing and Installation	13
5	Product features	13
5.1	Point-to-point reliable message transfer model	13
5.2	Publish/subscribe message transfer model	14
5.3	Synchronous/Asynchronous message consumption	14
6	Limitations	14
7	Quality ranges	15
7.1	Performance	15
7.2	Robustness	15
7.3	Fault Tolerance	15
7.4	Usability	15
8	Priorities	15
9	Other Requirements	16
9.1	Standards	16
9.2	System Requirements	16
9.3	Performance Requirements	17
10	Documentation Requirements	17
10.1	User Manual	17
10.2	Class Documentation	17
11	Changes History	17

1 Introduction

1.1 Purpose

The purpose of this document is to collect, analyze, and define high-level needs and features of the JMSME. It focuses on the capabilities needed by the stakeholders, and the target users, and why these needs exist. The details of how the JMSME fulfils these needs are detailed in the use-cases and supplementary specifications.

1.2 Scope

This Vision document is associated with the JMSME project, it discusses:

- product context (chapter 2).
- stakeholders and target users association with project properties (chapter 3).
- project products (chapter 4).
- project properties (chapter 5).
- project realization limits (chapter 6).
- project quality ranges (chapter 7).
- project priorities (chapter 8).
- other product requirements (chapters 9, 10).

1.3 Definitions

Client-side - JMS implementation classes running on J2ME device

J2EE - Java 2 Enterprise Edition

J2ME - Java 2 Micro Edition

J2ME CLDC - J2ME Connected Limited Device Configuration

JMS - Java Message Service

JMS provider - third party JMS implementation

JMSME - The name of the project, stands for Java Messaging Service Micro Edition

JVM - Java Virtual Machine

OpenSource - as defined on www.opensource.org

OS - Operating System

ptp - Point-to-Point messaging model

pub/sub - Publish/Subscribe messaging model

Server-side - JMS proxy running on J2EE middleware machine

1.4 Attachments

no attachments so far

1.5 Document Summary

Following chapters refer to:

- Product Context - Introduction of project goals and problems for which the project is the solution. Introduction of benefits the product will give.
- Persons related to the project - Stakeholders and target users description, analysis of their influence on the project and their needs.
- System Capabilities - System capabilities and needs they meet statement. Determination of goals priority. Notice about system limitations.

2 Product Context

2.1 Benefits

Realization of JMSME project will bring the following benefits:

- availability of free open-source jms-like api for J2ME platform
- possibility of building effective distributed systems in JMS technology, using devices running J2ME (palms, cell phones)
- easier development of J2ME network applications
- more powerful and attractive applications for J2ME
- popularization of JMS technology

2.2 Problem Introduction

Existing (very few) JMS implementations for J2ME come with some of the following drawbacks (for details see 'alternatives and competitors' section 3.8):

- they are not free
- they are not open source
- they use bandwidth consuming protocols (eg. based on xml), what makes them useless on devices with slow network connections (most of devices running J2ME)
- they are not intended for real use, are not practical

JMSME is intended to be free from all mentioned issues.

2.3 Product Context

JMSME products are usefull for all programmers who want to write applications in J2ME using JMS technology, and for organizations interested in using reliable distributed systems in JMS technology making use of devices capable of running J2ME.

JMSME delivers JMS-like API based on a subset of JMS capabilities for asynchronous communication in JMS point-to-point or subscription model. Asynchronous communication is perfect for devices usually running J2ME (cell phones, palms) that lack permanent internet connection. Unlike the few other available JMS implementations for J2ME, JMSME is going to be open source and lightweight (use efficient network protocols, and take as little system resources as possible). This efficiency is going to make JMSME products really usable on devices with very limited system resources and low network bandwidth (palms, cell phones). Delivery of highly efficient, open source JMS implementation for J2ME should encourage developers to make use of our product.

3 Stakeholders

3.1 Market Demographics

Year by year mobile phones become more popular as a way of communication. These days the number of cell phone users exceeds one billion. The growth of this market is very dynamic. Reliable sources (e.g. International Telecommunication Union) say, it is very likely that in 7 years every man in Western Europe will own one or more cell phones. Therefore mobile terminals should work not only as phones - actually possibilities for using mobile devices are almost unlimited.

They may be used to:

- chat with friends
- receive information we are interested in (e.g from topic lists)
- access remote systems
- have some fun (e.g. network games)
- and many others.

In some cases it is very important to know that our message was succesfully delivered to point of destination, or to be sure that our message will be sent as soon as the network is available.

The Java Message Service is a Java API that allows applications to create, send, receive, and read messages. JMS enables easy and efficient production of software with asynchronous and reliable communication.

JMSME will provide software with api similar to JMS, but dedicated to cell phones and Palms. There are a few products like JMSME trying to satisfy programmers needs but they are:

- commercial (e.g. Softwired product)
- too big to use it on devices with little memory
- inefficient (e.g. kJoram)

JMSME will be an Open Source application and will contain the fundamental functionality of JMS API. The client application will be small enough to be used in cell phones and will have a set of additional modules extending functionality for Palmtops. Amount of non-significant data transferred between mobile device and server will be minimized, which means better network utilization. Advantages mentioned previously make JMSME ideal for everybody who wants to create network applications for mobile devices - JMSME will be flexible, efficient and easy to use.

3.2 List of the stakeholders

Name	Represents	Role
Project coordinator	JMSME project coordinator	• Coordinates JMSME project development. • Checks compatibility of project with assumptions included in documentation.
JMSME programmer	Anybody who wants to develop using JMSME.	• Writes sources of JMS. • Creates project documentation. • Introduces new ideas concerning the project.

3.3 List of identified users

Name	Description	Stakeholder
Programmer	Writes applications for mobile devices using J2ME and high-level network communication implemented in JMSME.	Represented by “JMSME programmer”.
Administrator	Administrates the JMSME proxy server.	Represented by “JMSME programmer”.

3.4 Environment of target user

User name	Environment
Programmer	Any platform with Java Virtual Machine. Application written in J2ME.
Administrator	Proxy server administration application on any platform with Java Virtual Machine.

3.5 Stakeholders description

3.5.1 Project Coordinator

Representative	Grzegorz Grudziński
Description	Coordinates the whole JMSME project.
Type	Lecturer on faculty of Computer Science at Warsaw University.
Responsibilities	• Describing system requirements. • Making decisions about the final shape of the product. • Coordinating JMSME project development. • Checking compatibility of the project with the specifications included in documentation.
Success criteria	• Creating working and stable product. • Introducing JMSME to wide public.

Involvement	• Coordinating JMSME project development. • Checking compatibility of the project with the assumptions included in documentation.
Deliverables	Receive all documents connected with JMSME project: • Vision • Business Usecase • Software Development Plan • System Architecture Document • Usecase
Comments / Issues	Not yet.

3.5.2 JMSME programmer

Representative	Piotr Anders, Paulina Kania, Łukasz Osipiuk, Paweł Ostrowski
Description	Creates JMSME client and JMSME proxy server for mobile devices.
Type	Students of faculty of Computer Science at Warsaw University.
Responsibilities	• Making decision about final shape of a product. • Writing documents for JMSME project. • Writing source code of JMSME.
Success criteria	• Creating working and stable product. • Introducing JMSME to wide public.
Involvement	Creating product as similar to JMS as possible.

Deliverables	Creates and receives all documents connected with JMSME project: • Vision • Business Usecase • Software Development Plan • System Architecture Document • Usecase
Comments / Issues	Not yet.

3.6 Users description

3.6.1 Programmer

Representative	Piotr Anders, Paulina Kania, Łukasz Osipiuk, Paweł Ostrowski
Description	Writes applications for mobile devices using J2ME and high-level network communication implemented in JMSME.
Type	Students of faculty of Computer Science at Warsaw University.
Success criteria	• Possibility of using OpenSource JMS for mobile devices. • Easier and more comfortable creating stable application using JMSME product.

3.6.2 Administrator

Representative	Piotr Anders, Paulina Kania, Łukasz Osipiuk, Paweł Ostrowski
Description	Administrator of JMSME proxy server.
Type	Students of faculty of Computer Science at Warsaw University.

Success criteria	• Efficient and stable JMSME proxy server. • Wide possibilities of server configuration.
------------------	---

3.7 JMSME Application user

We will not consider user of JMSME based application a provider of requirements for JMSME project. The kind of application that can be written using JMSME can vary very much. That is almost impossible to guess what such application requires from a low level library (such as JMS), except the obvious matters.

3.8 Alternatives and competitors

3.8.1 kJoram

kJoram is light version of Joram - open source implementation of JMS messaging specification.

Advantages

1. It is Open Source, so it is free.
2. It is said to be already written.

Disadvantages

Data transfer in kJoram is implemented using XML so:

1. An XML parser is needed on the mobile device. Most cell phones have insufficient memory capacity for that.
2. Using XML means transferring very large amount of non-significant data. Network connections in mobile devices are very slow, so we can not afford their bandwidth be wasted.
3. It is still unavailable.

3.8.2 iBus//Mobile from Softwired Inc.

iBus//Mobile is commercial JMS implementation for J2ME platform.

Advantages

1. It contains almost fully featured Wireless JMS client library and JMS server gateway.
2. It is reliable.
3. It is efficient.
4. It supports many protocols: SMS, WAP, UDP, HTTP, HTTPS, GPRS, WiFi, GSM, MMS and many more

Disadvantages

1. It is commercial, non-free product.
2. Source code is unavailable.

3.8.3 WebSphere MQ Everyplace V2.0 from IBM

Message Oriented Middleware from IBM which implements JMS functionality.

Advantages

1. It is reliable.
2. It supports many different platforms.
3. It is full MOM system.

Disadvantages

1. It is commercial, non-free product.
2. Source code is unavailable.
3. It is one of the slowest in the market

3.8.4 Jtom

Jtom (java to mobile) is a message-oriented middleware which enables you to develop Java applications for small sets (PDAs, mobile phones etc.) and to access information of remote systems. With Jtom you can access application servers, SAP or other legacy systems from handheld online and modify data.

Advantages

1. It is reliable.
2. It is available.

Disadvantages

1. It is commercial, non-free product.
2. Source code is unavailable.
3. It is not exactly message sending system. Although it may be used in a similar way.
4. It is based on XML protocol. Therefore it can be inefficient.

4 Product description

4.1 Product perspective

JMSME will be designed to use one or more third party JMS implementations eg:

- Joram.
- OpenJMS

4.2 Product functions

- Implementation of reliable point-to-point message delivery protocol.
- Implementation of reliable subscribing of messages (specified by topic).
- Delivery of API similar to existing JMS API for J2EE platform.
The benefits:
 - User who knows J2EE JMS API will have to spend less time on learning how to use JMSME.
 - It will be easy to port programs written for J2EE which use JMS to JMSME.
- Efficient nonacademic implementation suitable for use with existing hardware (cell phones, Palms)
- Possibility of customization.

- User will be able to select the version of JMSME depending on his needs and hardware he uses. Eg. when writing application for cell phone he will use smaller version (with less functionality) and when writing for Palm the version which support most of JMS functionality.
- Administration interface for configuration and tuning the transport layer (proxy translating HTTP/UDP into JMS API calls)
 - Makes system maintenance easier.

4.3 Assumptions and dependences

4.4 Costs

It's going to be made without any costs at all :)

4.5 Licensing and Installation

JMSME will be distributed under the terms of GNU Lesser General Public Licence (LGPL).
Licence can be read at: <http://www.gnu.org/copyleft/lesser.txt>

5 Product features

5.1 Point-to-point reliable message transfer model

- JMSME system will provide API and middleware which will allow programmer (user of JMSME) sending and receiving messages over network. The ptp messages are sent and received from buffers (queues) provided by the middleware software (part of JMSME project). Each such queue is really “an interface” to queue provided by one of supported third party JMS providers.
- Reliability of communication. This means, that if you send a message using JMSME and get the confirmation you can be sure that no matter what happens the message will reach queue in third party JMS system.
On the other hand, when you receive the message, the JMSME system will take care that the message will be kept in the JMS provider queue until it is acknowledged.
- ★ There will be also an option of less reliable sending of messages, which does not guarantee arrival of the message in case of system or network failure, but
 - enhances performance of communication, minimizing additional time spent on writing messages to hard disk or database.
 - minimizes amount of additionally transfered data

- minimizes resources consumed by proxy server and library on client device
- ★ JMSME will provide API that will allow adding property fields to messages. Properties allow user to have a JMSME system select or filter messages on his behalf using application-specific criteria.
- ★ JMSME will provide client-side transactions. User will be able to group a series of operations together into an atomic unit of work called a transaction. If any of those operations fails, the transaction can be rolled back and the operations can be attempted again from the beginning. If all the operations succeed, the transaction can be committed.

5.2 Publish/subscribe message transfer model

- This is another messaging model provided by the native JMS API, which will be implemented in JMSME. In this model (pub/sub) client addresses messages to a topic. Publishers and subscribers may dynamically publish or subscribe to the content hierarchy. The system distributes messages incoming from a topic's multiple publishers to its multiple subscribers. Topics retain messages only as long as it takes to distribute them to current subscribers.
- ★ JMSME will enable user to create durable subscribers. The difference between normal and durable subscriber is that system will send messages to normal subscriber only when he is online (connected to provider). The durable subscriber will get also messages which were published when he was offline as soon as he connects.

5.3 Synchronous/Asynchronous message consumption

With JMSME you can receive messages in two ways:

- Synchronously. A subscriber(pub/sub) or a receiver(ptp) explicitly fetches the message from topic or queue.
- ★ Asynchronously. A client can register a message listener with a consumer. A message listener is similar to an event listener. Whenever message arrives a user specified action is taken.

Items marked with ★ will be available in extended version of JMSME.

6 Limitations

1. Implementation of JMSME must be written in J2ME.
2. The interface of JMSME must be a subset of JMS 1.1 API.

3. Size of client-side for mobile phones must be smaller then 30 KB.
4. Amount of non-significant data transferring between client and proxy application must be minimized.

7 Quality ranges

7.1 Performance

The client-side part of system utilizes up to 30 percent of CLDC platform memory and processor resources. The system also fully takes advantage of a network bandwidth of CLDC platform it works with. It means, the amount of additional data sent with the message is minimal.

7.2 Robustness

Assuming lack of errors in JVM and OS environment the server-side part of the system is able to work continuously and without supervision for indefinite time period. The client-side library will not allow the application using it to cause crash of J2ME system.

7.3 Fault Tolerance

In case of any fault:

server-side tries to overcome the fault and if it is impossible safely shuts down generating information

client-side stops execution and generates an exception or in other way informs parent application about the error

There can appear a communication problem or the application may be closed before all data was sent. In this case data is queued and sent as soon as possible.

7.4 Usability

The system can be used as a part of any application incorporating JMS 1.1 connections on J2ME CLDC platform. The only requirement is the usage of an JMS provider compatible with the system.

8 Priorities

Project goals divide into following categories according to their priority:

Critical Required for the project to succeed:

- basic JMS functionality
- client-side working implementation in J2ME CLDC environment
- server-side working implementation in J2EE environment
- compatibility with a single OpenSource JMS provider

Important The project will focus on:

- reliability
- low memory and processor requirements of the client-side
- high efficiency

Additional Desirable, but not necessary:

- full JMS functionality
- compatibility with different OpenSource JMS providers
- security

9 Other Requirements

9.1 Standards

- The client-side is compatible with J2ME standard.
- The server-side is compatible with J2EE standard.
- Communication between sides is based on TCP/IP or UDP/IP protocol.
- The system incorporates part of JMS standard.

9.2 System Requirements

- The client-side works on JVM compatible with J2ME standard.
- The server-side works on JVM compatible with J2EE standard.
- The system requires network connection between client and server-side to work. This connection does not have to be permanent.
- The server-side requires network connection to a compatible JMS provider to work.

9.3 Performance Requirements

- The system is reliable. It means that, if a message goes through the system, it is delivered or the sender is informed about a failure.
- The system fully takes advantage of a network bandwidth of CLDC platform it works with. It means, the amount of additional data send with the message may be ignored.

10 Documentation Requirements

10.1 User Manual

It is necessary to support user with a manual describing:

- installation and configuration of server-side
- creating connection between server and client-side

10.2 Class Documentation

Client-side must be accompanied with Class Documentation fully describing those classes, knowledge about which may be necessary to create application based on the system.

11 Changes History

\$Log: vision.tex,v \$

Revision 1.21 2002/11/11 13:12:58 lucash

Commands-pliczek z makrami, polowka pierwszego usecasa;)

Revision 1.20 2002/11/10 11:04:44 lucash

Wersja wyslana Grzesiowi

Revision 1.19 2002/11/06 11:11:21 piotrek

Dodane pare konkurencyjnych produktow.

Revision 1.18 2002/11/06 07:42:53 lucash

Nalozenie wiekszosci Grzesiowych poprawek (oprocz konkurencji)

Revision 1.17 2002/11/05 12:47:33 ofca

drobne poprawki

Revision 1.16 2002/11/01 15:23:15 pasza

Kilka kosmetycznych poprawek w punktach 1 i 2

Revision 1.15 2002/10/23 13:05:15 piotrek
Troche poprawek jakosciowych.

Revision 1.14 2002/10/23 05:40:31 lucash
spellcheck

Revision 1.13 2002/10/22 14:47:06 ofca
jako tako zrobione 3

Revision 1.12 2002/10/22 11:39:13 lucash
dodalem troche do 4

Revision 1.11 2002/10/21 15:57:21 lucash
drobne poprawki

Revision 1.10 2002/10/21 15:29:57 pasza
Posprzątałem po sobie (zrobiłem bałagan), dopisałem definicje o
których pisał Piotrek

Revision 1.9 2002/10/21 14:36:10 pasza
Jeszcze drobne uaktualnienia, poprawione kilka kiksów

Revision 1.8 2002/10/21 13:09:50 pasza

Dopisałem pkt. 2, poprawiłem (?) parę błędów jęz. u siebie i w punkcie 3
(1 conditional, zdania zaczęte od So, Or itd.), reszta jeszcze do sprawdz

Revision 1.7 2002/10/21 12:24:48 pasza
*** empty log message ***

Revision 1.6 2002/10/21 12:00:44 piotrek
punkty 7-10

Revision 1.5 2002/10/20 22:50:27 ofca
jako tako 3.1 i zaczete 6

Revision 1.4 2002/10/20 22:12:40 lucash
Dodatki w 4 rozdziale i 2 punkty w 5 rozdziale.

Revision 1.3 2002/10/20 19:58:08 lucash
robocza wersja pkt 4.

Revision 1.2 2002/10/20 19:12:15 pasza
*** empty log message ***

Revision 1.1 2002/10/17 14:47:34 lucash
Added vision template.

Revision 1.1.1.1 2001/10/30 17:49:15 robmar
zajecia, etc.