Evaluation of open-source versus commercial healthcare interoperability tools

By

Supachai Parchariyanon, MD, BBA

A CAPSTONE PROJECT

Presented to Department of Medical Informatics & Clinical Epidemiology

Oregon Health and Science University

School of Medicine

In partial fulfillment of

Requirements of the degree of

Master of Biomedical Informatics

July 2011

School of Medicine

Oregon Health & Science University

CERTIFICATE OF APPROVAL

This is to certify that the Master's Capstone Project of

Supachai Parchariyanon, MD, BBA

Evaluation of Open-Source Versus Commercial Healthcare Interoperability Tools

Has been approved

Capstone Advisor

Table of Contents

Acknowledgements

I would like to thank my advisor, Dr. Judith Logan, for her assistance and enthusiasm. Her feedback and experiences helped into what is now this project. I would also like to thank Diane Doctor for her wonderful support of distance learning students and without whom this project would not have been possible. Thank you also to Dr. William Hersh and the other faculty at OHSU's Department of Medical Informatics & Clinical Epidemiology for providing such an excellent environment in which to learn. Thank you to Dr. Nawanan Theera-Ampornpunt who introduced me to Medical Informatics. Lastly, thank you to Assoc.Prof. Artit Ungkanont, deputy dean of information technology, faculty of medicine, Ramathibodi Hospital, Mahidol University, Thailand for his vision to create healthcare IT academic workforce and provide full scholarship for me to pursue informatics degree.

Abstract

Interoperability is one of challenges in health informatics and there are many tools on the market to help informaticians break down silos of information and integrate them for the benefit of patient care. This paper aims to evaluate tools for healthcare interoperability, both open-source (Mirth Connect) and commercial (JAVA CAPS – Oracle Corporation, CA) to point out the benefits and risks including recommendations on how to get started if one would like to adopt such tools. There are 2 phases of the evaluation. The first phase focuses on data gathering and analysis and the second phase emphasizes an experimental study of the tools' efficiency. The result is that Mirth Connect is easier to learn and use while Java CAPS has more enterprise features and functions. The author recommends Mirth Connect as a solution to implement nationwide in developing countries.

Introduction

Interoperability is a property referring to the ability of diverse systems and organizations to work together (inter-operate).[1] When we apply the term to healthcare as "healthcare IT interoperability", it refers to the ability of health systems and organizations to work together. Health systems can include patient registration systems, clinical management systems such as electronic health records (EHR) and ancillary systems such as radiology information systems (RIS) or laboratory information systems (LIS).

There are 2 kinds of interoperability, syntactic interoperability and semantic interoperability. In short, syntax is structure and semantic is meaning.[2] If we have two health systems and they are able to communicate and exchange data, then they have syntactic interoperability. This is as equivalent to having spelling and grammar rules. The Health Level Seven (HL7) Version 2.x messaging standard, and Digital Imaging and Communications in Medicine (DICOM) are examples of standards for syntactic interoperability.

Semantic interoperability is the ability to automatically interpret the information exchanged meaningfully and accurately to produce useful results, as defined by the end users of both systems.[3] The Health Level Seven International (HL7) Version 3.x messaging standard is an example of a standard for semantic interoperability.

When we can transfer data from one system to another, we start to see benefit from health

information systems. Information exchange is a key component in any health information system and that is where standards come into picture. According to HL7, there are 6 categories of standards[4] as shown in Table 1.

Category Name	**Description**	**Example of standard**
Messaging and data interchange Standards	For clinical and administrative data interchange purposes	Health Level Seven (HL7), Digital Imaging and Communications in Medicine (DICOM)
Terminology Standards	To understand the same clinical terminology in different settings	Systematized Nomenclature of Medicine (SNOMED), Logical Observation Identifiers Names and Codes (LOINC), International Statistical Classification of Diseases and Related Health Problems (ICD)
Document Standards	For clinical document interchange	SOAP (Subjective, Objective, Assessment, Plan), Clinical Document Architecture (CDA), Continuity of Care Record (CCR)
Conceptual Standards	To define the framework	HL7 Reference Information Model (RIM)
Application Standards	To define how clinical applications interact to one another	Single Sign-on
Architecture Standards	To define the way infrastructure such as database communicate to one another	Public Health Information networks of the U.S. Centers for Diseases Control and Prevention (CDC) and the U.S. National Disease Electronic Surveillance System

Table 1 : Categories of standards[4] (California Healthcare Foundation)

A number of tools are available to help organizations comply with healthcare standards. For this project, I selected Mirth Connect as an open-source tool and Java CAPS as a commercial-grade tool to demonstrate healthcare standards and evaluate interoperability. Before going forward, I would like to give an overview of both tools and their standard use in this study.

Mirth Connect

Since its launch in 2006, the Mirth Connect interface engine has become the most widely downloaded open source software for healthcare data integration[6]. Started as the Mirth Project sponsored by Mirth Corporation, the contributor and user community now stands at over 8,000 registered users worldwide. As described in Mirth Corporation website (http://www.mirthcorp.com) "Mirth Connect is an open source standards-based healthcare integration engine. Mirth Connect facilitates the routing, filtering, and transformation of messages between health information systems over a variety of protocols (like LLP, Database, and FTP) with support for numerous standards (such as HL7, XML, and DICOM)."

There are 3 components in Mirth Connect. Mirth Connect Server performs message filtering, transformation, and transmission and also serves as the back-end of Mirth Connect; Mirth Connect Administrator connects to Mirth Connect Server and serves as a graphical user interface (GUI) tool to monitor interface activity and browse the message store; Mirth Connect Server Manager serves as a GUI tool to manage the Mirth Connect service, displays log files, and contains other configuration settings for the Mirth Connect Server.

Java CAPS

Java CAPS is a standards-based extensible software suite developed by Sun Microsystems[7] which was acquired by Oracle Corporation on January 27, 2010. As a result, Java CAPS is now an Oracle Product.

As described by Czapski[8], "Java CAPS stands for Java Composite Application Platform Suite, is a toolbox that supports many Enterprise Application Integration (EAI) styles, including database sharing, backbone messaging infrastructure both event-driven and service-oriented architectures and message transformation and routing can be implemented with support for numerous standards (such as HL7, TCP/IP, etc.)"

There are 4 principal components of Java CAPS. The Logical Host is in charge of hosting the applications deployed in it; this is where an instance of the Sun Enterprise Service Bus runs. The Repository is a version control system for all projects, and enables users to access and modify files when needed. The Enterprise Designer is a tool to create business processes, collaborations and connectivity maps to manage its inputs/outputs and the message flow based on business process execution language (BPEL). Finally, the Enterprise Manager is a web portal to monitor information flow through BPEL diagrams generated by Enterprise Designer. It also includes server logs, activities details and business processes parameters. More information about Java CAPS solution and architecture can be found in Appendix A.

My first experience with Mirth Connect amazed me. It helped me transfer a file from one folder to another server within a few seconds and with an easy to use GUI. Users can do the same with little effort and training. My experience with Java CAPS, on the other hand, was challenging as I had to be trained before I could understand the application and be able to use its tools effectively and efficiently; this is not a “plug and play” application.

The ISO/IEC 9126 international standard

ISO/IEC 9126 International Standard for software product quality is a widely accepted reference for terminology regarding the multi-faceted concept of software product quality[9]. There are 6 characteristics or dimensions namely, functionality, reliability, usability, efficiency, maintainablilty and portability. As outlined in the standard[10], functionality is a set of functions and their specified properties; reliability is a capability of software to maintain level of performance under stated conditions for a stated period of time; usability is the effort needed for use, and on individual assessment of such use, by a stated or implied set of users; maintainability is the effort needed to make specified modifications, and portability is an ability of software to be transferred from one environment to another.

The aim of this capstone project is to evaluate Mirth Connect as an open-source tool for healthcare interoperability and compare it with Java CAPS as a commercial tool based on The ISO/IEC 9126 international standard; in addition, this project will point out risks, benefits and investment needed if one would like to adopt such tools as healthcare integration engines.

Materials and Methods

In order to demonstrate both Mirth Connect and Java CAPS based on ISO/IEC 9126 international standard, the study was conducted in 2 phases, a data gathering and analysis phase and an experimental phase. Extensive review of books, articles and journals related to both tools was done in data gathering and analysis phase in addition to a self-study of a series of free webinars provided by Mirth Corporation (http://www.mirthcorp.com/webinars/mirth-connect-webinar) and 5-day, extensive training from x-tension (http://www.x-tention.at), a certified and experienced Java CAPS implementor. The aim of this phase was to summarize 5 out of 6 dimensions of analysis: functionality, reliability, usability, maintainablilty and portability.

The standard provides a framework for organization to define a quality model for a software application but does not include attributes as they vary between different software products. I selected some of sub-characteristics of software from the standard that can be applied to healthcare integration engines in order to summarize these dimensions and analysis (Figure1).

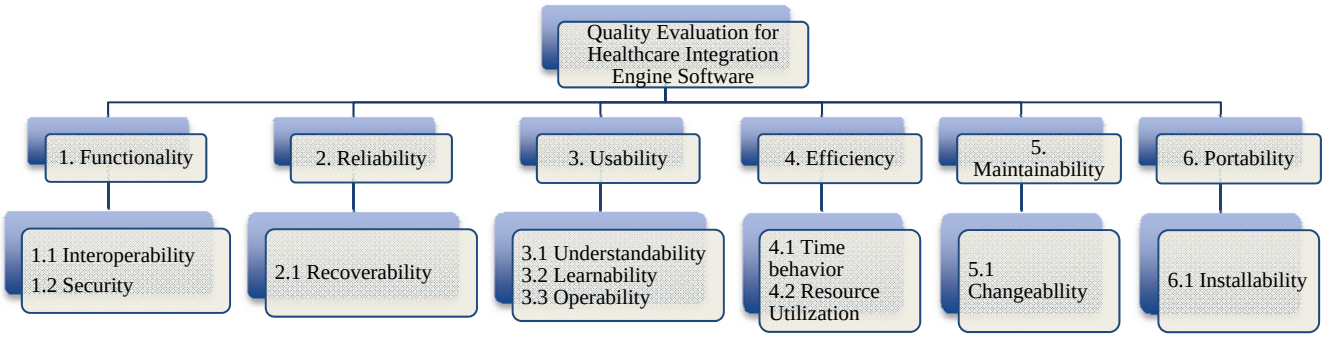


Figure 1 : Dimension and sub-characteristics for healthcare integration engine

For the experimental phase, focusing on the efficiency dimension, a simple experiment was performed, based on the current load of the system and expected growth rate in the future. I configured an environment using Virtual Machine, VMWare (http://www.vmware.com) with the Intel® Xeon® CPU, X5670 2.93GHz, Harddisk 1 TB, Memory 3.81 GB and Microsoft Windows Server 2003, R2 Enterprise edition, service pack 2, and installed Mirth Connect and Java CAPS on this virtual machine. I then generated text files containing HL7 messages as inbound messages to Mirth Connect and Java CAPS one system at a time and configured both systems to have the same outbound interface path in the same virtual machine.

The setting was a laboratory information system (LIS) which is the source system and the electronic medical record (EMR) which is the target system at the Faculty of Medicine, Ramathibodi hospital, Mahidol University, Thailand. Text files were generated in 2 rounds manually from the source system, the first round having 5,000 messages which is equal to 5,000 laboratory results and represents the average laboratory results volume per day at our institution; the second round had 20,000 messages to represent an estimated 4-fold rising of laboratory results volume in the next 4 years. After text files were generated, I recorded how long Mirth Connect and Java CAPS took to complete routing of messages from the source system to the target system and also monitored them via JConsole[11]which is out-of-the-box, Java monitoring and management console that allows administrators to monitor the usage of various resources at runtime.

As described by Chung[12], the used memory and committed memory are the amount of memory currently used and the amount of memory guaranteed to be available for use by the Java Virtual

Machine. My assumption is that less used memory and committed memory during runtime means more efficiency of the tools. This also helps determine scalability and flexibility if the concurrent load is higher in the future.

Results

Functionality. The results of the data gathering and analysis phase begins with the functionality dimension which is comprised of the attributes interoperability and security. Both Mirth Connect and Java CAPS have the same basic functionalities to cover healthcare data exchange. They both support Transmission Control Protocol/Internet Protocol (TCP/IP), File Transfer Protocol (FTP), HyperText Transfer Protocol (HTTP), File Directory, etc. One feature that Java CAPS can handle better than Mirth Connect is message validation. Mirth Connect only sends messages from source system to destination system as they are, without message validation, while Java CAPS has a feature to validate the format of message and drop poorly formatted messages out of the process.

In term of interoperability, Java CAPS empowers users with flexibility to manipulate the messages via Enterprise Designer[13] while Mirth Connect out-of-the-box features limit that the messages received must be in a correct syntax to be processed.

In terms of security, both have user authentication by username and password but Java CAPS has a role-based user management feature where an administrator can assign the right to access some features based on the role of the users.

Reliability. For the reliability dimension, I focused on recoverability. For Java CAPS, as it breaks down all message channels into services, if one service is not available, the other services are not affected e.g. if laboratory results data is sent from a LIS to an electronic medical record

and a separate clinical research repository system via Java CAPS, there are 3 services running in this scenario, one service is for the LIS to Java CAPS and another 2 services are running for Java CAPS to the electronic medical record and to the clinical research repository system. If the clinical research repository system is not functioning, laboratory results data will still be available to the electronic medical record. Mirth Connect uses the same concept but in a different way, e.g., Mirth Connect specifies one service from the source to destination as one channel, so if we use the same scenario as above, there are only 2 services, the LIS to the electronic medical record and the LIS to the clinical research repository system. It is meaningful that if the LIS is down, the messages queued in Java CAPS are still able to proceed since the other 2 services are available; on the contrary, the messages queued in Mirth Connect cannot be processed as the 2 services which connect to the LIS are also down.

Usability. For the usability dimension, Mirth Connect's functionality can be managed via GUI tools which are the Dashboard (a bird-eye-view of all channels and message statuses), Channel management (a specific view for each channel, to enable and disable channels), and Message tasks (tools for sending, importing, removing and reprocessing messages.) Java CAPS's functionality can be managed via GUI tools which are the Enterprise Designer (a tool to create business processes and connectivity maps), the Enterprise Manager (a tool to monitor information flow) and Watch Beyond (a tool to monitor messages and services statuses). I defined usability as an ability to understand, learn and operate the tools, and for Mirth Connect, I can understand its concept very easily, was able to learn how to use it from webinars, and required minimal support from Mirth Corporation's support forum[14]. Java CAPS, on the other

hand, is a more complicated application, such that I needed a book and more extensive training to understand its features and functions and could only operate it with third party support.

Maintainability. For the maintainability dimension, I focused on the attribute changeability. Both Mirth Connect and Java CAPS allow users to do some programming to better validate and

Dimension	Mirth Connect	Java CAPS
Functionality		
Interoperability	Supports HL7 Version 2 & Version 3, National Council for Prescription Drug Programs (NCPDP), American National Standards Institute ASC X12/EDI, XML, DICOM, Delimited text	Same as Mirth Connect
Security	Username and password authentication	Role-based username and password authentication
Reliability		
Recoverability	Low	High
Usability		
Understandability	Easy	Complicate
Learnability	High	Low
Operability	Easy	Complicate
Maintainablilty		
Changeability	Easy with GUI but has no library to support developer, if one needs to program something which is not a standard feature, one needs to do it from scratch	Easy with GUI and has its own library to import to help developer to program it faster
Portability		
Installability	Yes	Yes with option to manage multiple domains

TABLE 2 : Dimensions Comparison between Mirth Connect and Java CAPS

manipulate messages; interestingly, for Mirth Connect, users need to program from scratch while Java CAPS has its own library that users can import to allow faster development.

Portability. For the portability dimension, I focused on the attribute installability. Mirth Connect has an installer which helps you install on any server or machine. Java CAPS needs to be installed on specific servers as recommended by Oracle[15]

I have summarized and compared the dimensions of Mirth Connect and Java CAPS in Table 2.

Experimental Phase

Time behavior. Mirth Connect took 2 minutes and 6 seconds and 5 minutes and 28 seconds to complete a transfer of 5,000 messages and 20,000 messages respectively. Java CAPS took 33 minutes and 26 seconds and 6 hours, 15 minutes and 15 seconds to complete a transfer of 5,000 messages and 20,000 messages respectively. The result of the experimental phase of evaluation which shows the efficiency dimension of Mirth Connect and Java CAPS is in Table 3. In addition, screen captures of these processes can be found in Appendix B.

Resource Utilization. Mirth Connect took 2,630 Kbytes as used memory and 5,056 Kbytes as committed memory to complete a transfer of 5,000 messages and took 2,834 Kbytes as used memory and 5,056 Kbytes as committed memory to complete a transfer of 20,000 messages. Java CAPS took 415,092 Kbytes as used memory and 1,015,808 Kbytes as committed memory to complete a transfer of 5,000 messages and took 496,697 Kbytes as used memory and 667,120 Kbytes as committed memory to complete a transfer of 20,000 messages.

Measure	Mirth Connect	Java CAPS
Time Behavior		
To complete 5,000 messages transfer from LIS to EMR	2 minutes and 6 seconds	33 minutes and 26 seconds
To complete 20,000 messages transfer from LIS to EMR	5 minutes and 28 seconds	6 hours, 15 minutes and 15 seconds
Resource Utilization		
5,000 messages : Used memory	2,630 Kbytes	415,092 Kbytes
5,000 messages : Committed memory	5,056 Kbytes	1,015,808 Kbytes
20,000 messages : Used memory	2,834 Kbytes	496,697 Kbytes
20,000 messages : Committed memory	5,056 Kbytes	667,120 Kbytes

Table 3 : Efficiency Comparison between Mirth Connect and Java CAPS

For the initial investment, Mirth Connect is free, even though you might need a training to help you gain familiarity with its functionality faster. On the other hand, Java CAPS license will cost you $US 43,000 and with an annual fee of $US 8,500.

Discussion

From the results, we can see that Mirth Connect and Java CAPS share common functionalities and interoperability but there are 3 points worth discussing here.

1. Mirth Connect has a very limited monitoring tool. Concerning a big hospital with over 20,000 messages a day, a monitoring tool which includes email alerts would be very helpful to the administrator of the system; Java CAPS has such a feature while Mirth Connect does not.
2. Java CAPS efficiency, resource utilization and scalability need to be well analyzed prior to implementation. The reasons why Java CAPS took a longer time to process a message compared to Mirth Connect is because of its architecture. It needs to communicate with other modules before sending out a message while Mirth Connect is just a store-and-forward application. Therefore, Java CAPS needs bigger and better hardware and it comes with ability to manipulate messages better.
3. Java CAPS cost and long term investment. The biggest part of the story is the investment cost, while Mirth Connect you might be able to set up, run and maintain by your own team, for Java CAPS you would need an expert to help you from start. Apart from the initial investment for the software, also consider the cost of service which is usually not less than the cost of the software itself.

This study has some limitations. First, it was not intended to evaluate a comprehensive list of studies that measured quality model of a software or software evaluation framework, particularly

because such a list would be prohibitively long. Since the goal was never to comprehensively describe the evaluation framework used by all studies on software evaluation, but instead to describe the real experience so that insights could be gained that would lead to well-informed conduct of future studies, I feel that this study has accomplished its goal.

Another limitation is time limitation, as Java CAPS can take years to understand all features and functions, this study may not covered everything in Java CAPS but the fundamental knowledge and real experience were described in this study.

There are many possible future studies concerning healthcare integration engine include but not limited to adoption of healthcare integration engine, cost and benefit study, and outcome study of healthcare integration engine implementation.

Conclusion

I would recommend Mirth Connect as an healthcare interoperability tool of choice for Thailand for a countrywide implementation concerning the cost of investment and I encourage to set up the global communities for Mirth Connect to share problems and solutions, tips and techniques for implementation and community itself is able to help one another by developing a better monitoring tool to make Mirth Connect even better.

Reference

(1) Wikipedia. Interoperability. 2010; Available at: http://en.wikipedia.org/wiki/interoperability. Accessed Nov 18, 2010.

(2) Mead CN. Data Interchange Standards in Healthcare IT—Computable Semantic Interoperability: Now Possible but Still Difficult, Do We Really Need a Better Mousetrap? J Healthc Inf Manag;20(1):71.

(3) HL7 E-LEARNING COURSE, HL7 Argentina. Introduction to Healthcare Interoperability. 2010.

(4) California Healthcare Foundation, Clinical Data Standards Explained 2004 Available at: http://www.iha.org/pdfs_documents/calinx/FactSheetClinicalDataStandardsExplained.pdf. Accessed Sep 2, 2011.

(5) Wikipedia. Semantic Interoperability. 2010; Available at: http://en.wikipedia.org/wiki/Semantic_interoperability. Accessed Nov 18, 2010.

(6) The Mirth Story. 2011; Available at http://www.mirthcorp.com/company/the-mirth-story. Accessed Aug 21, 2011

(7) Java CAPS. 2011; Available at: http://en.wikipedia.org/wiki/Java_Caps. Accessed Aug 21, 2011

(8) Java CAPS Basic. Implementing common EAI Patterns; Czapski M, Krueger S, Walker A, Prentice Hall; 2008

(9) International Organization for Standardization. ISO/IEC 9126-1: Software engineering - product quality - part 1: Quality model, 2001.

(10) Correia, J.P.; Kanellopoulos, Y.; Visser, J.; , "A survey-based study of the mapping of system properties to ISO/IEC 9126 maintainability characteristics," *Software Maintenance, 2009. ICSM 2009. IEEE International Conference on* , vol., no., pp.61-70, 20-26 Sept. 2009

(11) Chung M.; Monitoring and Managing Java SE 6 Platform Applications; Available at http://java.sun.com/developer/technicalArticles/J2SE/monitoring. Accessed on Aug 21, 2011

(12) Chung M.; Using JConsole to Monitor Applications; Available at http://java.sun.com/developer/technicalArticles/J2SE/jconsole.html. Accessed on Aug 21, 2011

(13) Java CAPS Basic. Implementing common EAI Patterns; Czapski M, Krueger S, Walker A, Prentice Hall; 2008., pp.203

(14) Mirth Connect Forums; Available at http://www.mirthcorp.com/community/forums/forumdisplay.php?f=3. Accessed on Aug 21, 2011

(15) Planning for Java CAPS installation; Available at http://download.oracle.com/docs/cd/E19509-01/820-3741/jcapsinstall_intro/index.html. Accessed on Aug 21, 2011

Appendix A : Java CAPS Solution and Architecture

Java CAPS Architecture

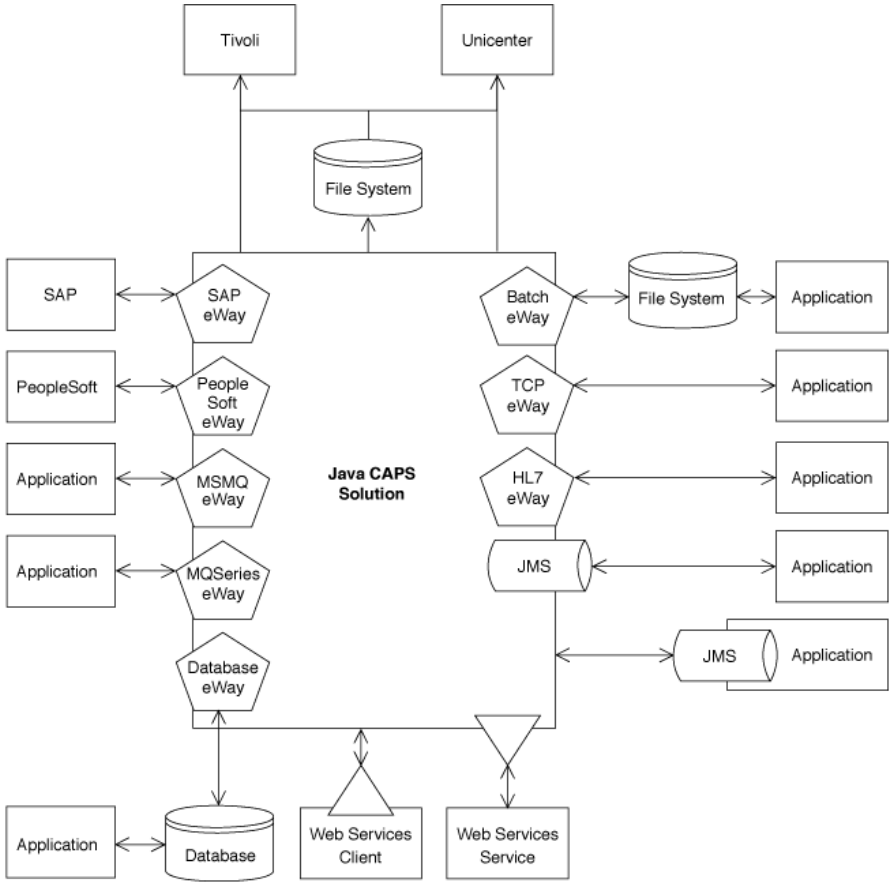


Figure 1 : Java CAPS Solution Context

Source : Java CAPS Basic; Implementing common EAI Patterns; Czapski M, Krueger S, Walker A, Prentice Hall; 2008

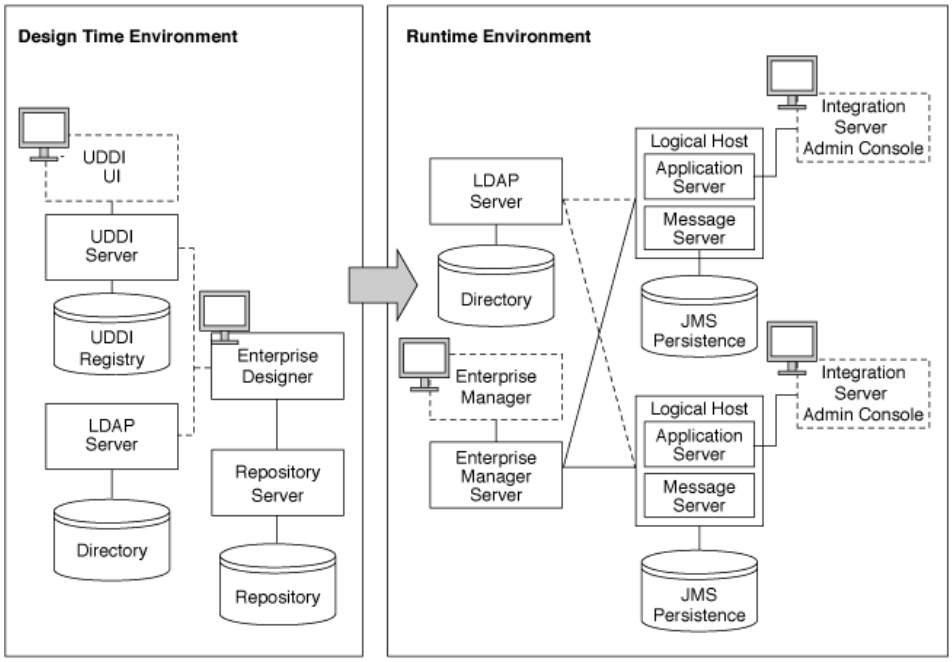
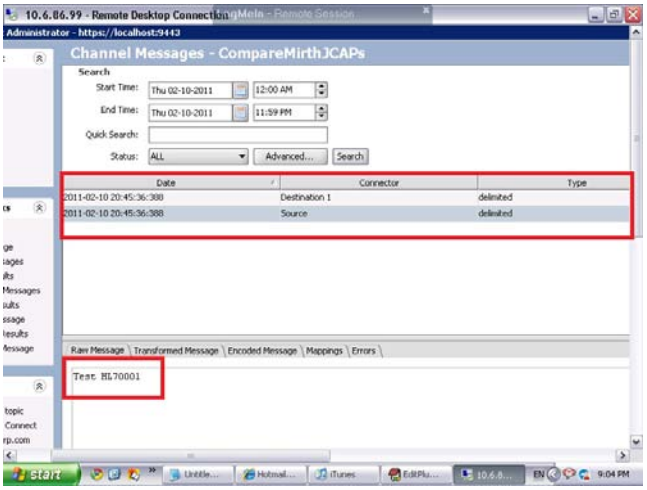


Figure 2 : Java CAPS Architecture

Source : Java CAPS Basic; Implementing common EAI Patterns; Czapski M, Krueger S, Walker A, Prentice Hall; 2008

Appendix B : Screen Captures of the Experimental Phase

Mirth Connect took 2 minutes and 6 seconds to process 5,000 messages



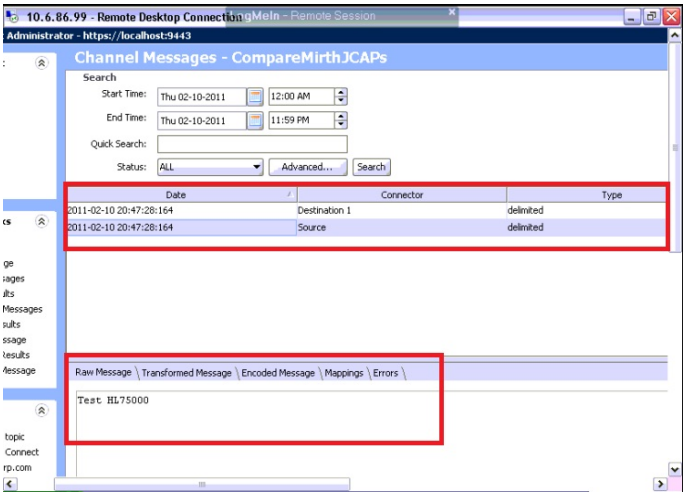
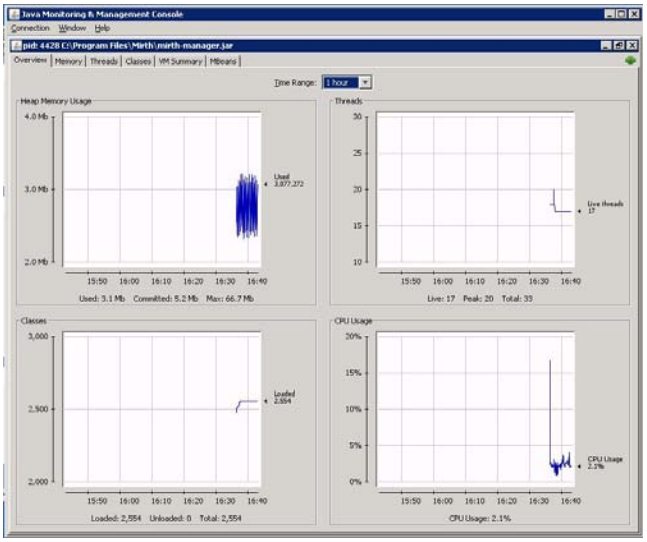


Figure 1&2 : Mirth Connect, Start time and End time



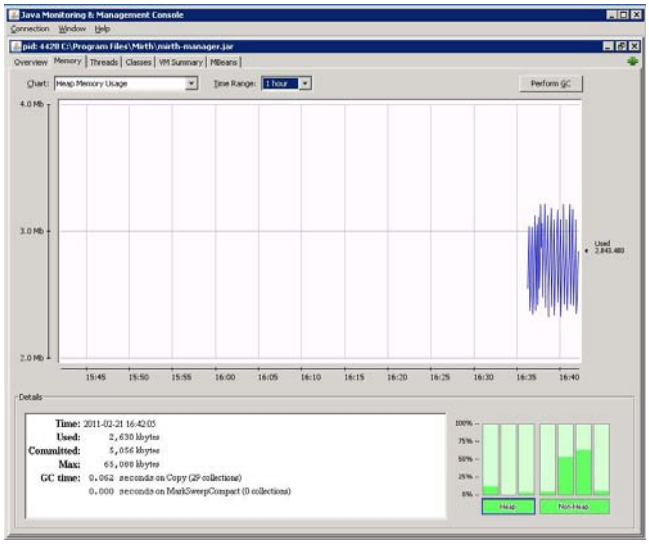
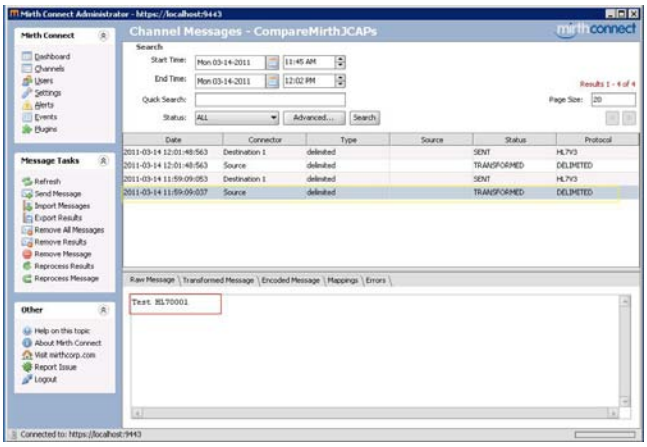


Figure 3&4 : JConsole of Mirth Connect

Mirth Connect took 5 minutes and 28 seconds to process 20,000 messages



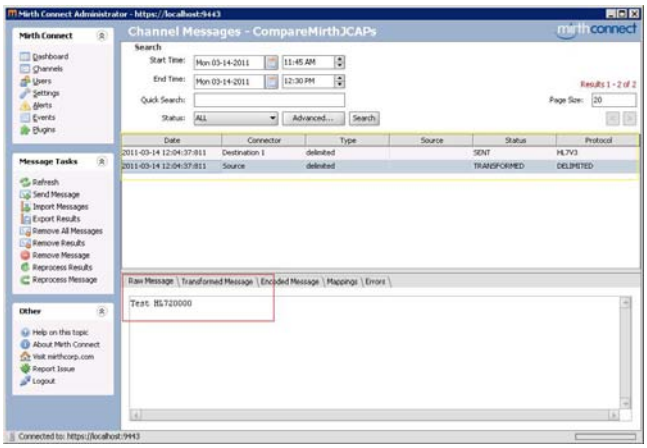
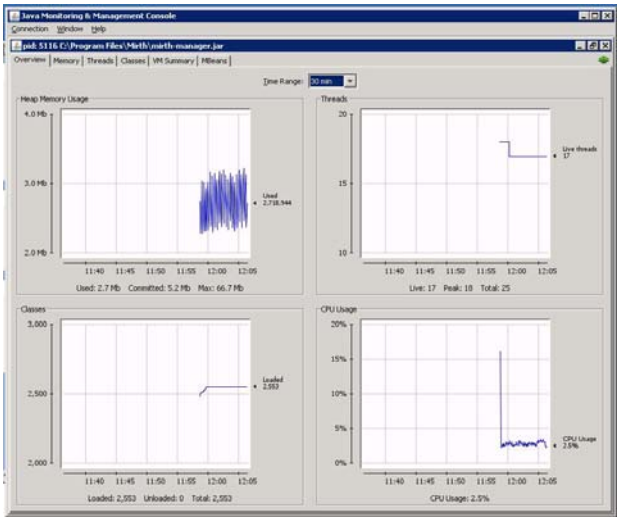


Figure 5&6 : Mirth Connect, Start time and End time



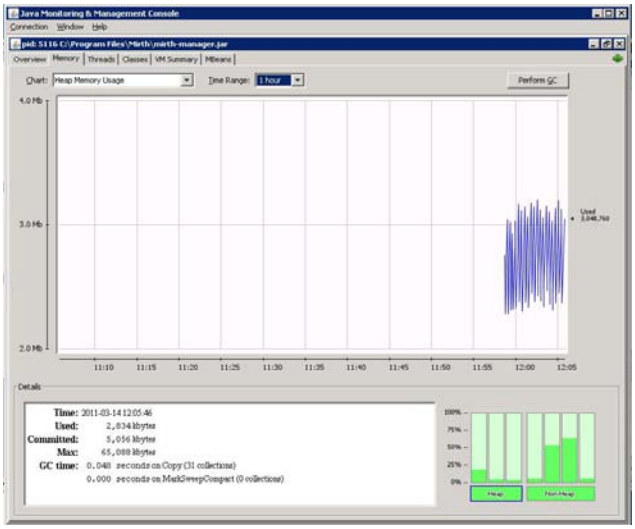
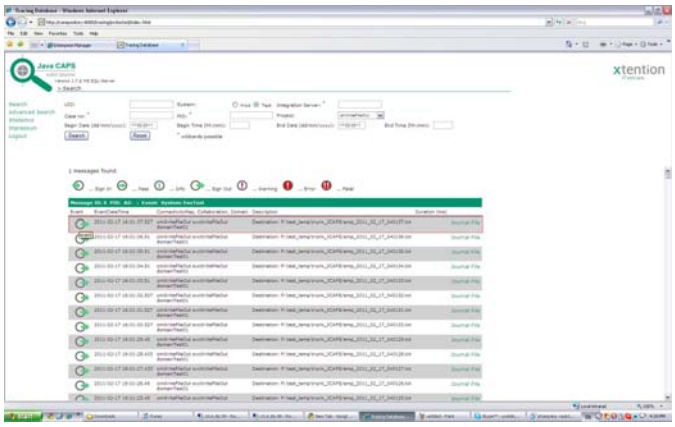


Figure 7&8 : JConsole of Mirth Connect

Java CAPS took 33 minutes and 26 seconds to process 5,000 messages



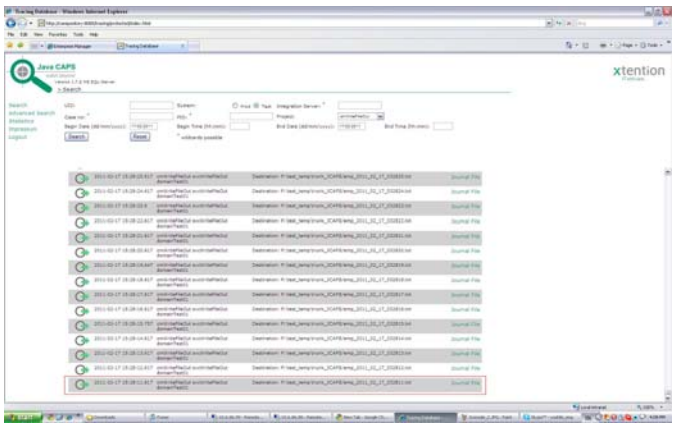
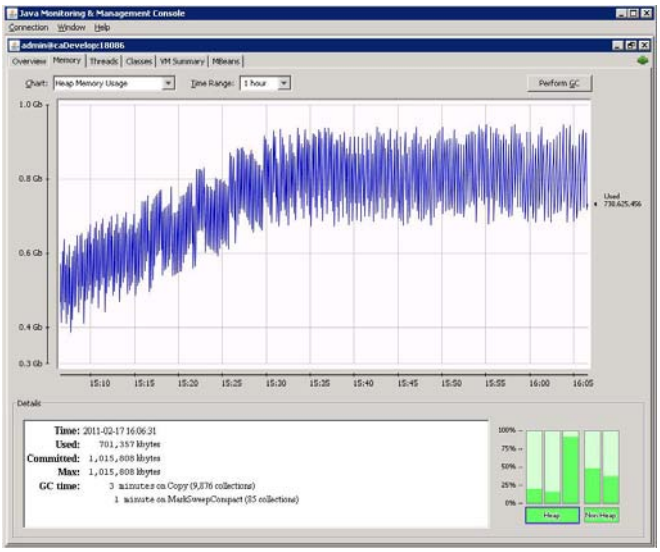


Figure 9&10 : Java CAPS, Start time and End time



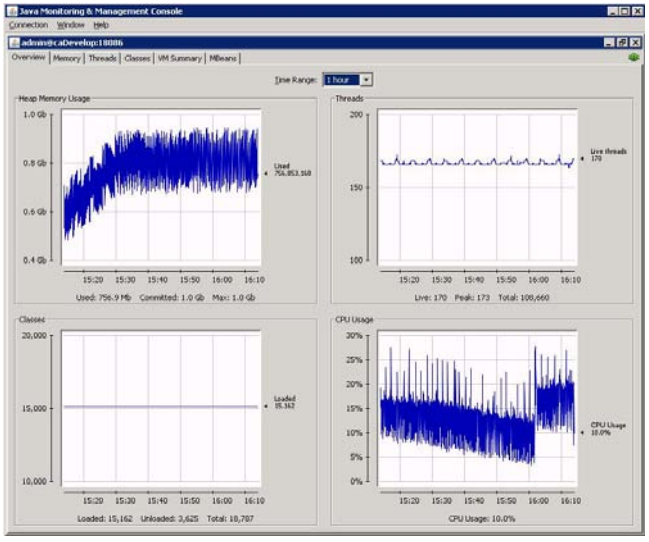
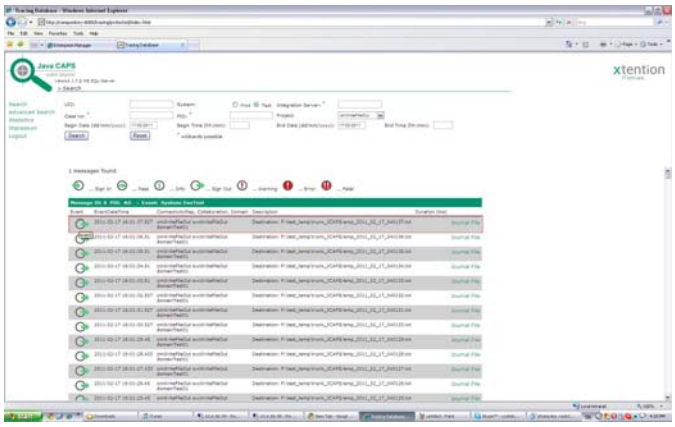


Figure 11&12 : JConsole of Java CAPS

Java CAPS took 6 hours, 15 minutes and 15 seconds to process 20,000 messages



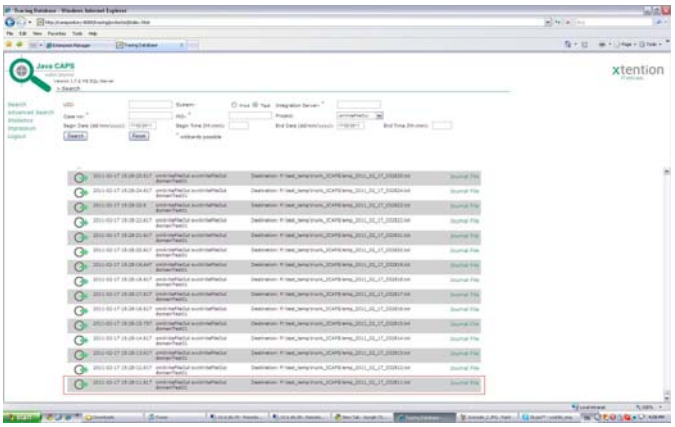
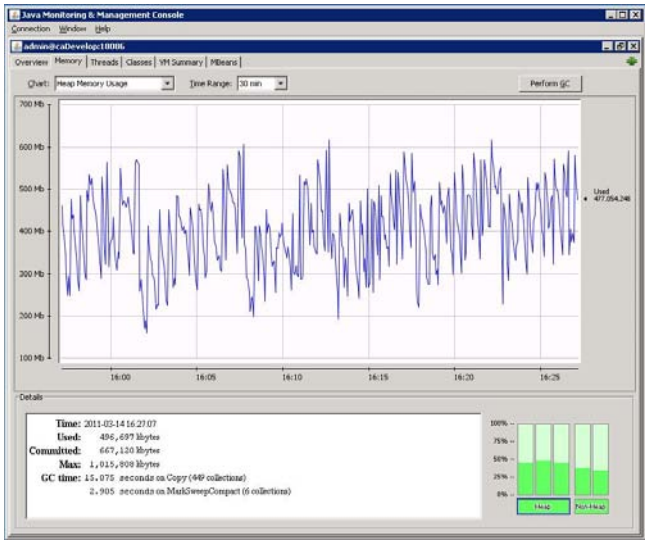


Figure 13&14 : Java CAPS, Start time and End time



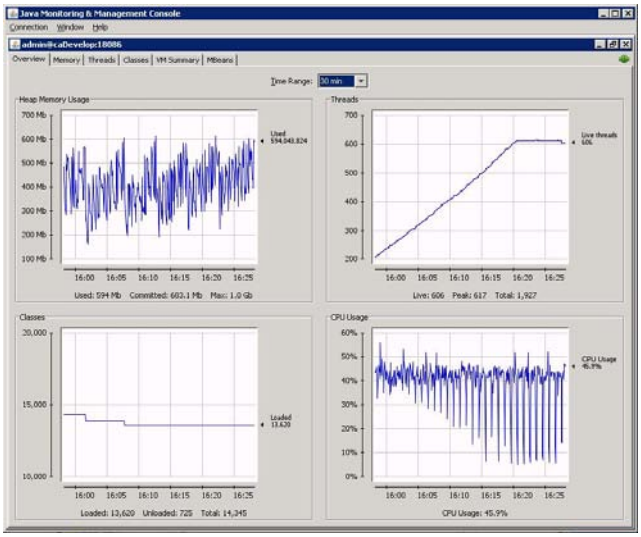


Figure 15&16 : JConsole of Java CAPS