Software Engineering and Building Multi-Tier E-Business Applications Using Web Services Technology: Forget Everything You Know?

Jane Huffman Hayes *(corresponding author)*
Laboratory for Advanced Networking
Computer Science
University of Kentucky
301 Rose Street
Lexington, KY 40506-0495
(859) 257-3171 (859) 323-3740fax
hayes@cs.uky.edu
and Science Applications International Corporation
jane.e.hayes@saic.com

Henry (Hui) Cai
Computer Science
University of Kentucky
1608 University Court, C310
Lexington, KY 40503
(859) 323-9748
hcai0@hotmail.com

Abstract

The development of the Web has dramatically changed how we develop software, or has it? E-computing represents a new paradigm. All the knowledge we have gained and tools and methods we invented for developing traditional software systems are not applicable to this new movement, or are they?

In order to examine these questions, we undertook a project to become familiar with the state of the art in web service technology and the popular J2EE technology. We investigated the state of technology of e-commerce, including examining the various categories of software, their significance, and their current and future application. We looked at the popular e-commerce technologies, including platforms for the development of e-commerce software. We examined the software engineering methods and tools available to help develop e-commerce relevant software. Next, we planned and implemented an e-commerce project, using traditional software engineering as well as web service technology-specific tools and methods.

The project is an on-line pizza order system. It comprises 4 sub-systems: a portal, two virtual back-end pizza stores, and a Universal Description, Discovery and Integration (UDDI) registry. These 4 sub-systems are loosely connected through Simple Object Access Protocol (SOAP) messages.

We used "traditional" computer-aided software engineering tools such as Rational Rose to assist us. We also used e-commerce specific tools such as WebSphere Application Server, VisualAge for Java, and Web Services Toolkit. We found that it is not necessary to "forget everything you know" when developing e-business applications using web service technology, but that traditional tools are not as immediately applicable as hoped. For example, traditional tools can help perform requirement specification and analysis and requirement driven testing of the developed application. But visual model capabilities have limited utility: it can not be used to model the web pages and their hierarchy. This paper presents our experience using traditional tools and methods to develop leading edge e-business applications and suggests some areas for future research.

Keywords

e-computing; software engineering; software engineering tools; software engineering methods; e-business; web services technology; computer-aided software engineering

1. Introduction

The development of the Web has dramatically changed our world. In the e-computing area, new concepts and technologies emerge in a continuous and fast pace. Their impact on the traditional software engineering methods and computer aided software engineering (CASE) tools is a common concern. Meanwhile, many new software and tools occur simultaneously, most of which are technology-specific. We wonder if it is possible to make use of these domain-oriented tools and general-purpose CASE tools effectively for e-computing software development.

In order to examine these questions, we decided to undertake a project that satisfied three requirements: use the latest technology in e-business, apply a standard software engineering approach using both technology-specific and general-purpose tools, and use typical technologies, tools and methods. After investigation, it was decided to develop an online pizza store demo using Web Services and J2EE technologies. The whole project was implemented in the Rational Unified Process approach. Rational Rose was used as a representative of "traditional" CASE tools. Many e-commerce specific tools were also used, including WebSphere Application Server Advanced Single Server Edition, VisualAge for Java Entry Enterprise Edition, and Web Services Toolkit (WSTK). We found that traditional software engineering methods and CASE tools are still very useful in developing e-computing software, but there are some unique challenges in the development of e-computing software.

2. Technology Background

2.1 Web Service

Last June, Bill Gates announced Microsoft's ambitious .NET strategy. They envision the day when businesses and consumers will be able to rent Internet-based service and software instead of purchasing proprietary packaged software. Meanwhile, IBM put forward their dynamic e-business offering. They concentrate on streamlining the integration of systems across the internet and intranet. Both of them depend on the Web Service technology, which is a way for businesses to expose their services programmatically over the Internet, leveraging standards-based protocols and document interchange formats to execute any kind of online transaction.

A number of open standards have been developed to promote the popularity of the Web Service. They include Web Services Description Language (WSDL)[1], Simple Object Access Protocol (SOAP) [2], and Universal Description, Discovery and Integration (UDDI) [3].

Services are deployed on the Web, and published at the service UDDI registry (service broker) by service providers. The functions provided by a given Web service are described using the WSDL. A service requestor first finds the services it needs in the UDDI registry, then consumes these services provided by service providers, as shown in Figure 1 [4].

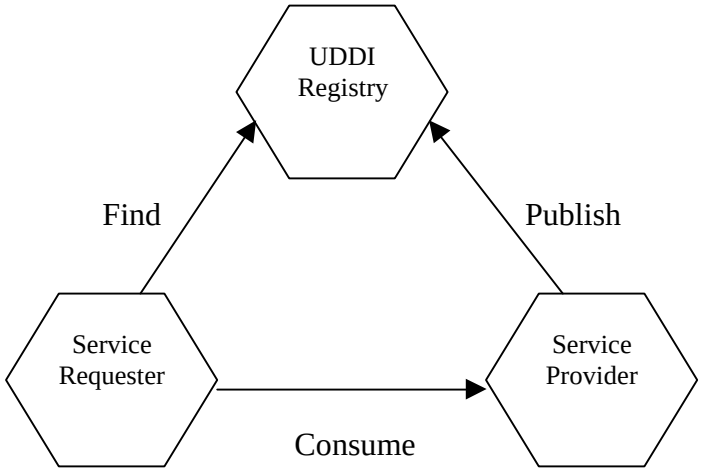

Figure 1. Web Service Architecture.

2.2 Java 2, Enterprise Edition

Java 2 Platform, Enterprise Edition (J2EE) technology is a standardized component-based approach to the design, development, assembly, and deployment of multi-tier enterprise applications [5]. Central to this component-based model is the dividing of the enterprise applications into containers and components. Since the vendors of J2EE application servers help to implement the system-level services in the container, the application developers can focus on the implementation of business logic through components. Applications built in this way can be deployed in any application servers that support the J2EE specification.

3. Online Pizza Application

3.1 Overview

The project is an online pizza store demo using J2EE and Web Services technologies. It comprises four sub-systems: a portal, two back-end virtual pizza stores: "Great Store" and "Super Pizza", and a UDDI registry, as shown in Figure 2. All of them are loosely connected through SOAP messages. The pizza stores publish their services in the UDDI registry. The portal searches the pizza stores and relevant services through the UDDI registry, gathers concrete and up-to-date information (like price) from pizza stores, and publishes it to the end users. When the user finds the services they need (like pizza delivery), they order them directly through the portal. The portal will transfer the orders to the corresponding pizza stores. After receiving the order, the pizza store will send the user a verification email.

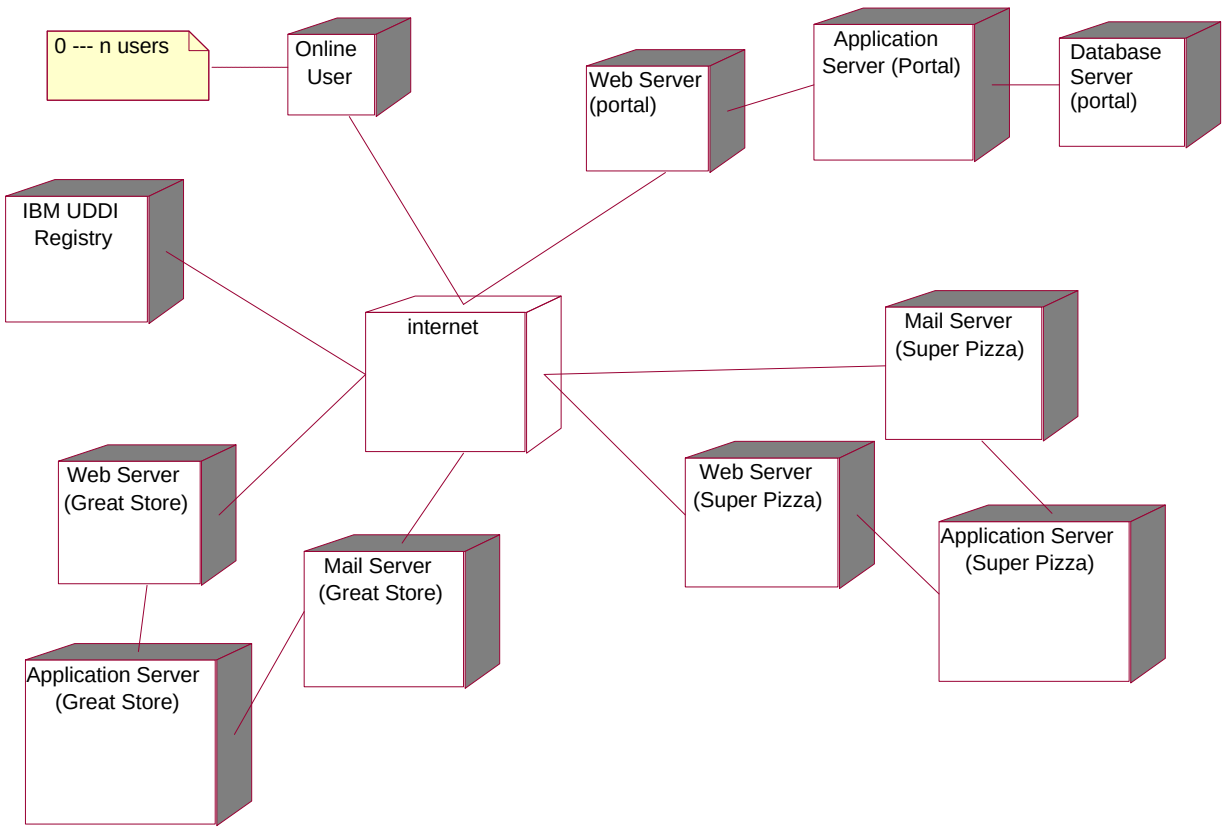

Figure 2. System Architecture.

3.2 Multi-tier Portal

Among these four sub-systems, the portal was a typical multi-tier application implemented on J2EE platform. As is typical, the model-view-controller pattern was applied, and the system was divided into four tiers: Client tier, Web tier, Enterprise JavaBeans (EJB) tier, and Enterprise Information System (EIS) tier.

1) Client tier: The web browser is the client in this case. It is the only part that can be seen by the end user. It communicates with the Web tier through HTTP.
2) Web tier: It is responsible for performing all web-related processing. After receiving the request from the user, it will decide the actions that should be performed, then calls the EJB tier to implement the business logic for the request. Then, it will select and return the suitable view to the user according to the request and the result of the actions. The Java Servlet was used to implement the controller, and Java Server Page (JSP) was used to implement the view, generating the dynamic content.
3) EJB tier: EJB tier represents user's account information, and conversational state, including the pizza searching criteria, available pizza stores, and corresponding price.
4) EIS tier: The EIS tier was the enterprise information infrastructure, using IBM's DB2 server, storing the users' account information.

3.3 Development Process

The whole project was implemented following the Rational Unified Process, which is a commercially proven and widely accepted approach for software development [6]. The software development lifecycle was broken into three cycles. In the first cycle, the back-end pizza store was implemented. In the second cycle, the portal sub-system was implemented to enable pizza browsing and ordering. In the last cycle, the user management module was added to the system to facilitate the user's ordering process. Each cycle went through the inception, elaboration and construction phases. Since it is not a commercial product, the transition phase was deliberately omitted.

There were four core engineering workflows involved in the project: requirements, analysis & design, implementation, and test. In the requirement workflow and analysis & design workflow, Rational Rose was used to generate a visual model (such as use-case model and design model) and model element (such as use-case diagram, sequence diagram, collaboration diagrams, state transition diagram, etc). In the implementation workflow and test workflow, many technology-specific tools were involved. Visual Age for Java was used to generate the skeleton code for Java Servlet and Enterprise JavaBean. Meanwhile, it was also used as the main editor and run time debugger. The WSTK was used to generate the WSDL documents, which was later published in the UDDI registry to enable public access. WebSphere was the application server, but also provided the tools that helped to wrap the code artifacts into web accessible components and assemble the software components into the final applications.

4. Applicability of Software Engineering Tools and Methods to Web Application

As discussed above, we used conventional CASE tools and many technology-specific tools in our project. We also followed the standard software engineering development lifecycle. Below we offer an evaluation of applicability of standard software engineering methods and tools:

Table 1. Applicability of Software Engineering Methods.

Software Engineering Methods	Applicable to e-business application?	Requires modification for use in e-business applications?
Requirement Analysis	Yes – applicable to whole application	No.
Design	Yes – whole application	Yes, need new mechanism to model the web pages and their hierarchy.
Implementation and Test	Yes – whole application, with tailoring	No.

Table 2. Applicability of CASE Tools.

Tools	Applicable to e-business application?	Requires modification for use in e-business applications?
Rational Rose	Yes – useful for requirement analysis, design, and code generation	Yes, needs to add support for modeling web pages and their hierarchy.
VisualAge for Java	Yes – code generation, debugging, and testing	No.
WebSphere	Yes – code generation, deployment, testing	No.
Web Service Toolkit	Yes – WSDL documents generation, service publishing	No.

5. Conclusions and Future Work

5.1 Effectiveness of CASE tools for e-computing software

Nowadays, CASE tools can generally provide the following functions: 1) visual modeling for business process, requirement analysis, and component architecture design, 2) change control, 3) quality control, and 4) code generation. There is no doubt that all of these functions are very useful for software development, including the development of e-computing software. Meanwhile, because of the popularity of the web, the e-computing software is studied more thoroughly. Many standards have been stipulated for e-computing software. This makes it easier for CASE tools to generate more meaningful code for e-computing software. For example, in the J2EE platform, because the interface between the containers and the components has been standardized, there are much more skeleton codes that can be generated for EJB components than other ordinary components. In essence, the level of standard maturity of an engineering process directly influences its possible automation level.

5.2 Technology Specific Tools and General Purpose CASE Tools

There is a trend for the development of CASE tools: with the advent of a new technology, some new tools always emerge to support it. In order for the traditional and general-purpose CASE tools to become more powerful and occupy more market share, they either integrate these new tools into their original framework, or start from scratch. This decides the effectiveness of these two kinds of tools in e-computing software development. For the activities in the software development lifecycle, the higher the abstract level, the more effective for traditional CASE tools, while the nearer to the source code, the more effective are the technology-specific tools. This gives us a clear suggestion for improving tools for e-computing software development.

5.3 Future work

From the above discussion, we can see that if the traditional CASE tools can quickly assimilate the new emerging technology specific tools into their framework, they would hold more market share. Therefore it is important to examine how to develop an open framework for CASE tools that is always ready to add new features. Also, it would be better if the CASE tools provide the ability to model the web pages in the same way that class diagrams are used to model the objects.

References

[1] http://www.w3.org/TR/wsdl
[2] http://www.w3.org/TR/SOAP/
[3] http://www.uddi.org/
[4] http://www.ibm.com/developerworks/webservices/
[5] http://java.sun.com/j2ee
[6] Rational Unified Process, a Rational Software Corporation White Paper, 1998.