Mobile Web Service Architecture for Software Development Peer Inspection

Gilda Pour
College of Engineering
San Jose State University
San Jose, CA, U.S.A.

Ashwini Ramachandra
IBM
San Jose, CA, U.S.A.

Panchali Sarkar
College of Engineering
San Jose State University
San Jose, CA, U.S.A.

Abstract - *Improving software quality remains a key challenge. Software development formal peer inspection has emerged as an effective approach to address this challenge. Software peer inspection aims at detecting and removing software development defects efficiently and early while defects are less expensive to correct. The key research issue is to develop an architecture that supports the "systematic" process of software development formal peer inspection. In a major research project sponsored by National Academies' National Research Council (NRC) and NASA, the author has led the research to develop Web-based enterprise architecture for software development formal peer inspection. The research results have been used as the foundation in this research project, led by the author and co-sponsored by NASA, IBM, and HP. The research focus is to enhance the mobility features in the Web service-oriented system for software development formal peer inspection. This paper presents the new architecture and its prototype.*

Keywords: Mobile Web Service, Web Service-Oriented Architecture, Software Development Peer Inspection.

1 Introduction and Motivation

A major challenge facing the software community is to improve software quality. Software development formal peer inspection has emerged as an effective approach to address this challenge. Software development formal peer inspection process was first introduced were by Michael Fagan at IBM.

Software development formal peer inspection is widely used, particularly in enterprise software development. This process aims at efficiently and effectively detecting and eliminating or drastically reducing defects at every phase of the software development lifecycle while defects are less expensive to correct. It is intended to:

- reduce the number of errors detected during integrated testing and operations where fixes are much more costly to make;
- provide the inspection participants with the opportunity to learn effective software practices, and to reinforce and refine existing programming standards; and
- provide insight into software development processes and areas where process improvements can help defect prevention [1].

Formal peer inspections may be applied to any product or work products, including requirements, design, code, and test plans. Software development peer inspection includes the following stages: planning, overview, preparation, inspection meetings, rework, re-inspection, and follow-up [2].

The system is required to allow only authorized users (members of a peer inspection team, for example, author, product area leads, moderators, inspectors, and lead inspector) to have location-transparent access to the system. Each team member's access to peer inspection checklists, reports, defect logs, test plans, and other test-related documents depends on the team member's role and responsibilities.

The inspection forms and data often have different formats and structures, and are located on different platforms and in different physical and logical locations over the Internet.

Thus, the two main categories of research issues in development of a system for the systematic process of software development peer inspection are:

1. Location-transparent access of authorized users (peer inspection team members) to heterogeneous inspection forms, data and tools

located at different physical and logical locations on different platforms over the Internet.
2. Heterogeneity of data, forms, and tools involved in the software development peer inspection process [1].

Systematic peer inspection process requires an adaptable, flexible, and robust IT architecture. In a research project, led by the author and sponsored by National Academies' National Research Council (NRC) and NASA, the author has developed new Web-based enterprise architecture for software development formal peer inspection [1]. The findings and results of that research project have been used in this project.

The focus of this research project, led by the author and co-sponsored by NASA, IBM, and HP, is to provide mobility features in the Web service-oriented system that supports systematic process of software development formal peer inspection. The use of Web services and component-based development approach helps address the issues of software reusability, adaptability, extensibility, and flexibility [1] [3-5].

This paper is organized as follows. Section 2 provides background information. Section 3 presents the design of the mobile Web service-oriented application for software development peer inspection, and its architecture. Section 4 presents our demonstration prototype of the system for PDAs. Section 5 provides concluding remarks and the plan for further enhancement.

2 Background

The currently available commercial tools such as ReviewPro, ASSIST, LEAP, NnMega DevPartner, CSRS, CodeWizard, and Scrutiny can be used only for limited peer inspection effort in some phases of software development. With an exception of ReviewPro, they are not Web-based applications. More importantly, no currently available system supports mobility features for software peer inspection.

We present a new system architecture that supports formal peer inspection in all phases of software development. It also supports location-transparent and role-based access of authorized users (inspection team members) to the software peer inspection system through mobile devices such as PDAs.

3 Mobile Web Service-Oriented System

In this project, a mobile Web service-oriented system is designed to allow software development peer inspection team members (authorized users) to access the software development peer inspection application using mobile devices such as PDAs and cell phones.

The system has Web service-oriented architecture. It accesses the Web server hosting Web services for the software development peer inspection system. Database implements all the application rules using stored procedures. XML-based Web services enable access to remote and enterprise-level functions and data on mobile devices. The services use Web standards to access functions via HTTP on the Internet or Intranet. This allows application logic to be accessed over the Internet as well.

3.1 The Mobile Web Service-Oriented Architecture

Figure 1 illustrates the high-level design of mobile Web service-oriented architecture for software development formal peer inspection. It is 3-layer architecture. Layer 1 allows users to access the application system. It receives user's requests, sends it to layer 2 for processing, and provides a response to the user's request once it receives the response from layer 2.

A client uses a Web service proxy to access a Web service. The Web proxy maps available Web services to the user application. Thus, it hides implementation details from the user. The Web proxy handles all the underlying communications using a variety of standards such as SOAP and XML over HTTP.

Layer 2 publishes Web services and provides application logic of the application. XML-based Web services are published using the Internet Information Services (IIS) or any other Web server engine. Through layer 1, they receive users' requests and send the results (output) to the user application over the Internet using SOAP and XML standards over HTTP. Layer 2 also includes ADO.NET Web services. Layer 2 uses SQL stored procedures and database triggers. The stored procedures are fired by the Web Services using ADO.NET connectivity.

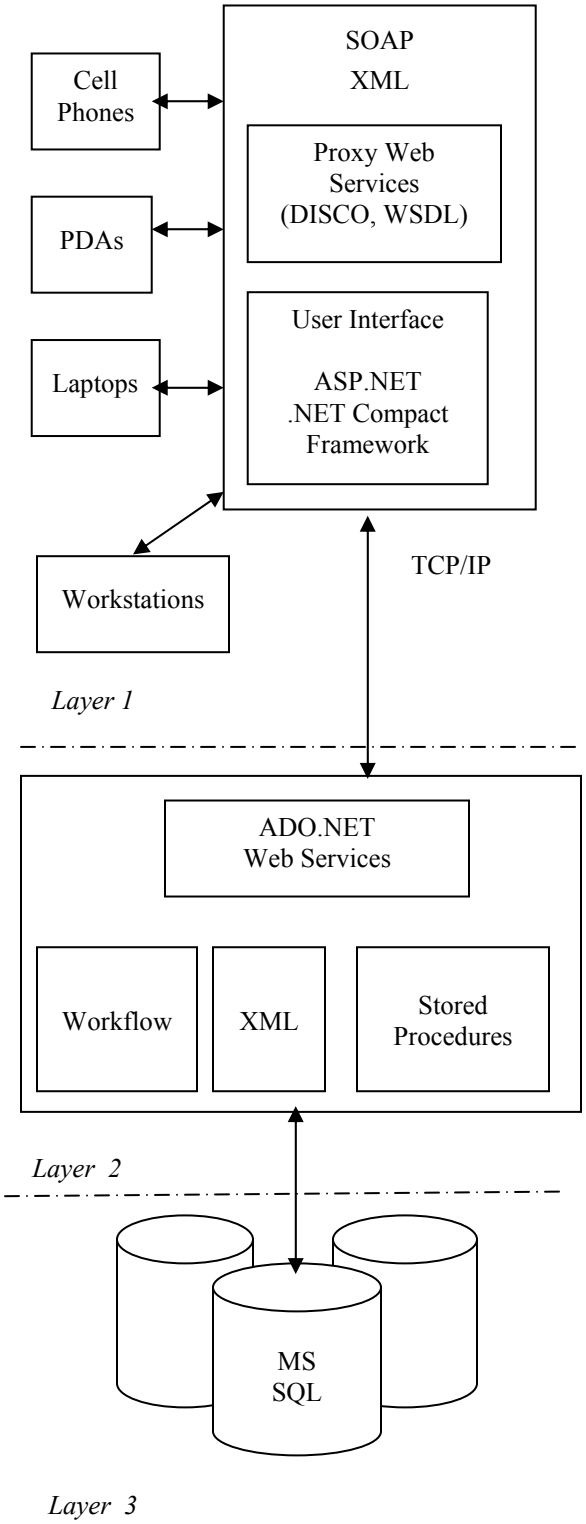

Figure 1. High-Level Design of the Mobile Web Service-Oriented Architecture for Software Development Peer Inspection

The application on the client device connects to the server via the Internet. The server IP address is used by the client system to obtain the description of the Web services available on the server.

3.2 Workflow Design

In software development peer inspection process, inspection team members have different formal roles, such as author, moderator, inspector, lead inspector (a.k.a. reader), and recorder. Each team member has a specific, defined role. We have developed workflow using the role-based responsibilities and respective checklists [2].

The following list includes the specific responsibilities for each role in the peer inspection process.

An author is responsible to:

- generate all work products to be inspected and provide required reference materials for overview and inspection meeting;
- interact with moderator in preparation of inspection package;
- respond to questions raised about the work products during the inspection process;
- resolve the issues raised and the defects detected during the inspection process;
- prepare a rework estimate and schedule; and
- review the corrections with the moderator according to the requirements of "Follow-up".

A moderator is responsible to:

- select peer inspection team members, assign roles to each team member, and prepare the team for the inspection;
- ensure that the product has met the required “entry criteria” prior to the inspection;
- conduct the inspection meeting, and ensure that the inspection meetings focus on detected defects in the product being under inspection;
- classify defects according to the requirements in "Defect Classification";
- ensure that information generated from any open issues or action items are communicated to and addressed by the author;
- work with the team to identify any need for re-inspection of the work products;
- verify, personally or by delegation to inspectors, to ensure all detected defects are corrected prior to re-inspecting and/or authorizing placement of the inspected product under configuration

control for delivery to the next phase of the software life cycle; and also no new defects are inserted during the correction; authorize placement of the inspected product under configuration control for delivery to the next phase in the software life cycle; and

- collect the data, and generate and files the inspection summary report and a survey report for the project under inspection.

An inspector is responsible to detect the technical defects in the product being under inspection. Lead inspector is responsible to:

- guide the team through the work product;
- participate in detecting defects in the product presented for inspection; and
- lead inspectors through the inspected product and related materials in a logical progression, paraphrase and summarize each section.

A recorder is responsible to:

- accurately record all defects and meeting discussions, and return the resulting list to moderator at the end of the inspection meeting; and
- participate in detecting defects.

4 The Demonstration Prototype

In this research project, we have built a demonstration prototype of the new system using Java-, .NET-, and XML-based technologies and SOAP. .NET-based technologies, such as MS Visual Studio.Net, C#, and a PocketPC emulator were used.

Microsoft Visual Studio .NET for developing ASP.NET Applications and .NET Compact Framework for developing client applications in Pocket PC were used. The other parts of the system were developed J2EE-based technologies and Java-based Web services and components. MS SQL Server is also used for database management.

Figure 2 shows the UML Class Diagram for the inspection services. Figures 3-8 illustrate the screenshots of the prototype system.

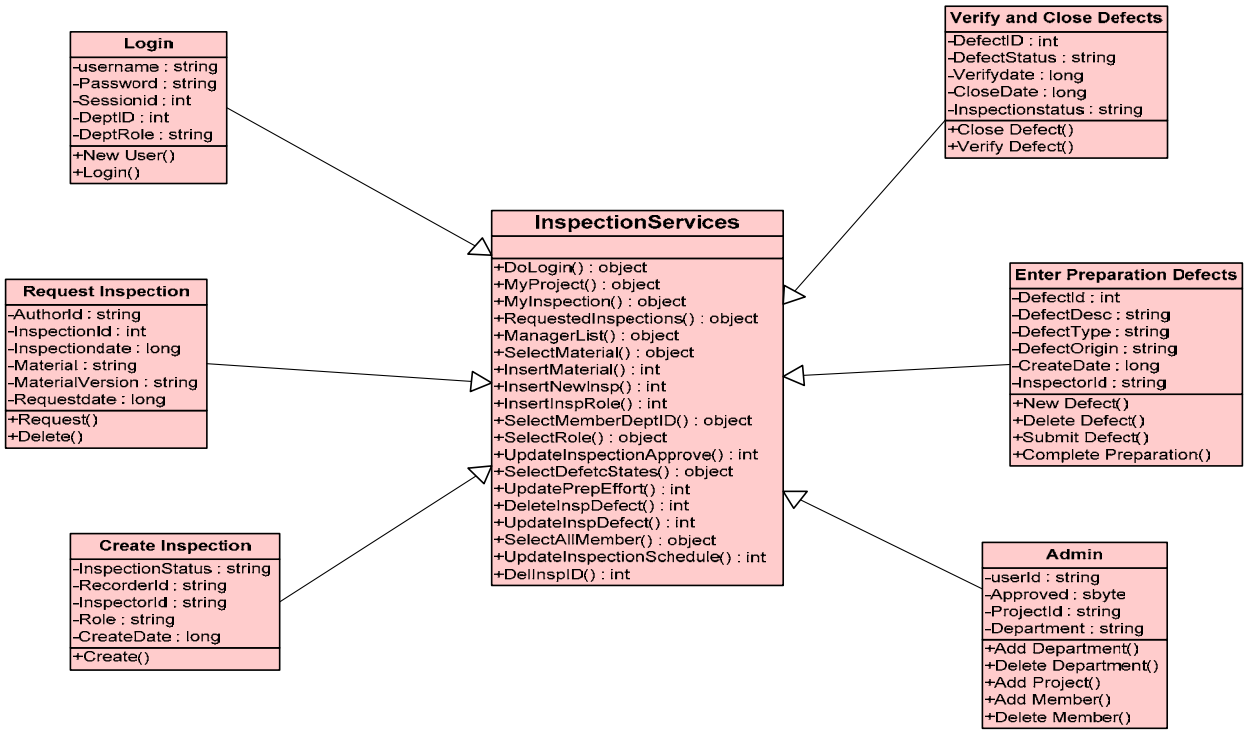

Figure 2. The UML Class Diagram for Software Development Peer Inspection Services

Figure 3. The System Login Screenshot on a PDA

Figure 5. Inspection Request Screenshot on a PDA

Figure 4. Project Status Screenshot on a PDA

Figure 6. Inspection Session Screenshot on a PDA

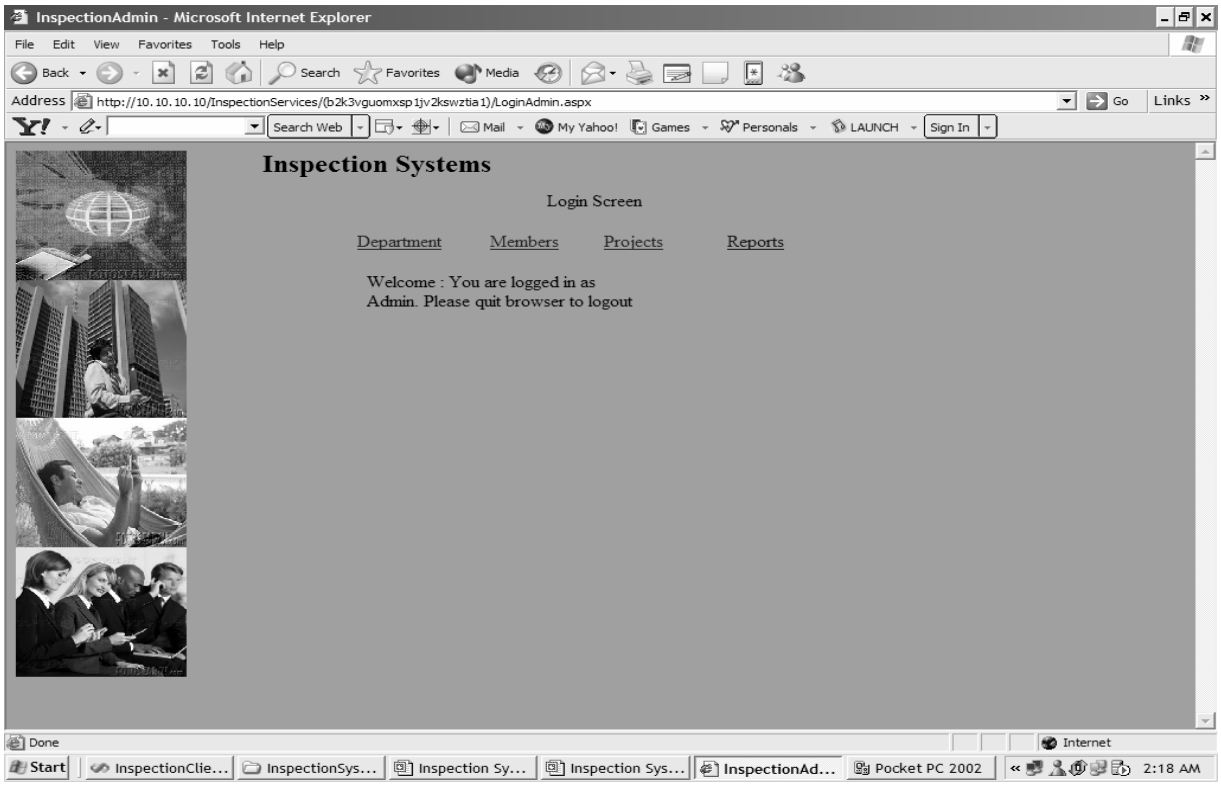

Figure 7. System Administrator Screenshot

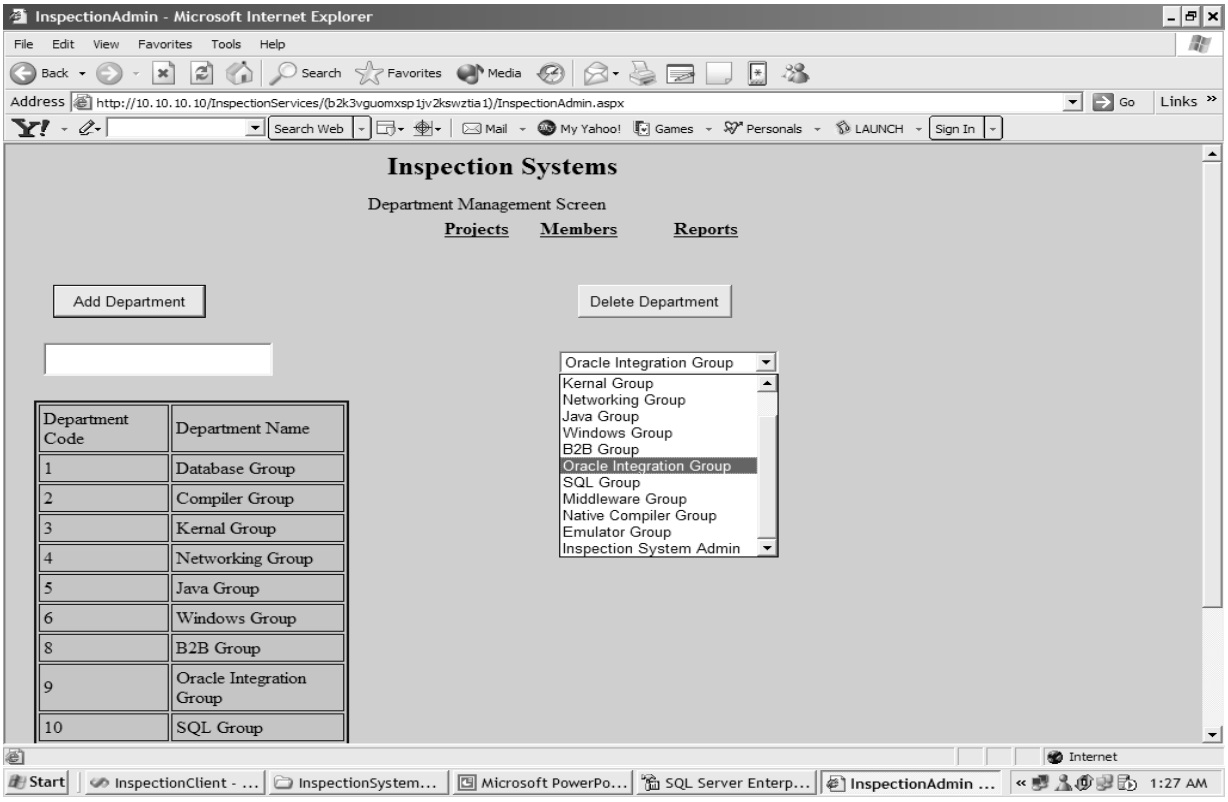

Figure 8. Department Manager Screenshot

5 Concluding Remarks

In this research project, we have developed mobile Web service-oriented architecture for software development formal peer inspection. This new mobile system offers the key benefits of Web services and software components.

In this project, we have also developed a demonstration prototype of the new mobile Web service-oriented system using .NET- and XML-based technologies. The other parts of the system have been developed using J2EE and Java-based technologies.

The results of our applied research are very promising. This mobile Web service-oriented architecture can be enhanced to include other mobile devices such as smart phones. We plan to explore the pervasive computing features in this system. For that, we use agent-oriented software engineering [6] and mobile agents and Java-based technologies and JADE-LEAP, J2ME, CLDC, MIDP [7-10].

The results of this research project can benefit research to develop mobility features for Web service-oriented systems for other problem domains.

6 Acknowledgment

We gratefully acknowledge the co-sponsorship of the research by NASA, IBM and Hewlett-Packard Company.

7 References

[1] G. Pour, "Web Service-Based Framework for Software Development Formal Peer Inspection," *Internet Computing Series*, 2004.

[2] NASA Software Assurance Technology Center, NASA Software Formal Inspections, http://satc.gsfc.nasa.gov/fi/fipage.html.

[3] G. Pour, "Component Technologies: Expanding the Possibilities for Component Based Development of Web-Based Enterprise Applications," *Handbook of Internet Computing*, (B. Furht, ed.), CRC Press, 2000.

[4] G. Pour, A. Rao, C. Nguyen, and R. Mangoba, "Web Service-Oriented Enterprise Architecture for Mobility Support of Prescription Management," *IEEE Publication Series on E-Technology and E-Services*, Vol. 2, No. 1, IEEE Computer Society, 2005.

[5] F. Curbera, et al., "Unraveling the Web Services Web," *IEEE Internet Computing*, Vol. 6, No. 2, March-April 2002.

[6] M. Griss and G. Pour, "Accelerating Development with Agent Components," Cover feature article, *IEEE Computer*, Vol. 34, No. 5, 2001.

[7] G. Pour and N. Laad, "Enhancing the Horizons of Mobile Computing with Mobile Agent Components," IEEE Computer Society Press, 2006

[8] G. Pour, "Exploring Multi-Agent Architectures for Autonomic Systems," *Pervasive Computing*, Vol. 2, No. 1, 2005.

[9] G. Pour, F.C. Yao, and C.C. Yu., "A Mobile Agent-Based Architecture for Mobile Systems Supporting Distributed Software Project Management," *IEEE Publication series on Software, Telecommunications & Networks*, Vol. 5, No. 1, 2004.

[10] G. Pour, "Exploring Mobile-Agent-Based Architectures for M-Commerce Applications," *E-Commerce Series*, Vol. 1, No. 2, 2004.