4th ICSE Workshop on "Software Engineering over the Internet"

Frank Maurer & Carmen Zannier
Department of Computer Science
University of Calgary
2500 University Drive NW
Calgary, Alberta, Canada, T2N 1N4
maurer@cpsc.ucalgary.ca, czannier@ucalgary.ca

Abstract

The 4th ICSE workshop on "Software Engineering over the Internet" brought together researchers and practitioners that are trying to use Internet technologies to overcome problems in distributed software development. The goal of the workshop was to exchange ideas how distributed projects can utilize the Internet to overcome communication, collaboration, and coordination problems. We summarize the presented papers of the workshop

Keywords: Distributed Software Development, Internet

Introduction

Following the successful ICSE 1998-2000 workshops on "Software Engineering over the Internet" [1, 2, 3], the 4th workshop of the series will focus on experience reports and evaluation.

As large companies are widely deploying web-based software process support environments, the goal of the workshop was to discuss lessons learnt using industrial case studies as a major focus. Distributed projects are always a challenge for project managers and developers. Web-based software process support environments help people from physically distributed locations to collaborate in the development of software systems. They cover management issues as well as the improving the execution of technical tasks.

In this workshop, researchers and practitioners shared a vision of "Software engineering over the Internet" and discussed how distributed teams can effectively and efficiently work together. Questions that were addressed during the workshop were:

- What lessons were learnt from deploying web-based process support environments? What are their benefits and shortcomings? What empirical data is available on their successes and failures?
  Several papers reported initial industry experiences with Internet-based software development. Initial results are encouraging.
- How can virtual software enterprises be efficiently formed and effectively work?
  The presentation by Kötting & Maurer dealt with a virtual marketplace for software development tasks.
- How effective is distributed software development compared to co-located development?
  Damian and Eberlein reported compared distributed and co-located requirements negotiation in their presentation.
- What are the problems special to distributed software engineering projects and how can we make use of the Internet to solve them?
  Sveral presentations discussed specific approaches to apply Internet technologies for enacting standard SE practices, e.g. inspections and configuration management.
- What methods and tools do we need to support software projects over the Internet?
  Several presentations dealt with tool support. Obviously, without tools an Internet-based software development process is not possible.

Presentations covered several phases of software engineering: including requirements engineering, design, and testing.

Paper Summaries

In the following, we summarize the papers that were presented at the workshop. The full papers are accessible on the Web at [4].

Setting Up Distributed Software Inspections by Danilo Caivano, Filippo Lanubile, Giuseppe Visaggio

One of software engineering's "best practices" is software inspection. Inspections are used to discover and correct problems in software as early as possible in the software development process. The problem however of clerical overhead, time, costs etc. never disappears. Thus software inspection is simple and useful in theory but more difficult to actually practice. It has become clear that the more pairs of eyes searching for defects, the higher the likelihood of finding defects in software. Still, when discussing distributed software development, a problem arises in how to manage an inspection involving as many members of a software development team as possible. Software inspections differ from other kinds of reviews and to be effective, a meeting may be required. The paper by Danilo Caivano, Filippo Lanubile, and Giuseppe Visaggio discusses a reorganization of the inspection process. Specifically, the preparation and meeting phases of the original inspection process are replaced with defect discovery, defect collection and defect discrimination phases, in that order. An Internet-based inspection system is described as an implementation of a approach that supports distributed software inspections on the Internet. A UML deployment diagram is provided and adapting to a redesigned inspection process is discussed.

Web-based Inspection Process Support Environment for Software Engineering Education by Atsuo Hazeyama and Akiko Nakano

The influx of information technology into everyday life demands an increasing number of software systems and more importantly, people to develop and maintain such systems. People from varying social and cultural backgrounds must learn to work together, allocate tasks to members and fulfill all aspects of project management. Such skills however, are not frequently learned in a formal education. Consequently, a System Design class has been intro-

duced to provide students with the necessary skills of "real world" software development. Due to restraints on university schedules however, time for sufficient inspections of the students' work is limited. Thus, a Web-based Inspection Process Support Environment has been developed to allow for inspection within a university's time constraints. The system focuses on two issues not considered in previous inspection methods: monitoring the progress of several groups and managing the comments associated with the artifacts. The support system is broken into three components, the first providing support for the inspection process, the second monitoring progress and the third supporting version and configuration management. The paper concludes with a discussion of the students' evaluation, the major issues appearing as time and comprehension of demands from the teacher's side.

Providing Software Engineering Services for Virtual Software Organizations by Frank Maurer and Jeremiah Wittevrongel

When looking at e-business applications, one cannot ignore the challenge of keeping up with fast changing technology. Economically, it makes little sense for small development companies to purchase the latest software, yet these products provide the services the companies require to be successful. A better solution for such companies would be to lease, or borrow, the required services and use them only for the time the services are needed. To illustrate this idea, the authors implemented a tool, SCENTOR, that supports the generation of test drivers. SCENTOR provides scenario-based testing of Java applications, specifically EJB-based Java applications, to virtual groups. The service is available online and requires no installation. SCENTOR is able to load UML sequence diagrams from XMI files. It is ideal for lightweight development processes that use these. The paper describes the details of SCENTOR and briefly discusses future work as using SCENTOR as a link between functional tests based on user scenarios and load test drivers.

An Architecture fo Collaborations: a Case Study by Heather L. Oppenheimer and Dennis Mancl

This paper presented some experiences from using Internet technologies to support distributed software development. It is apparent that software is best developed when all members of the software team are co-located, but it is also apparent that this is not possible, especially with large and complex companies. Consequently, people have become used to net meetings, conference calls and the like. Bell Laboratories Software Technology Centre (STC) works to enhance the collaboration obstacles that exist in software companies where members are not co-located. The focus is a knowledge transfer initiative to assist in the transfer of information between, among and across varying groups. As the members of the STC, Bell Laboratories and their customers are located across America and Europe, they have become ideal candidates for the knowledge transfer initiative. The paper discusses the varying groups and the challenges faced in the transfer of knowledge. Issues key to distributed members "getting along" are discussed as well as solutions to the challenges the company faces. It becomes clear that collaborative tools are used quite frequently and these are discussed briefly.

Integrating Software Engineering Tools and Repositories with XML and XSLT by Henrik Hedberg

Various tools are used in a distributed software development environment, but all must be able to access the same data without compromising its integrity. Thus interoperability is required so that applications can continue to operate as they were, and still communicate with other applications. XSQML is based on XML and is a data query and manipulation language. The key is to perform data transfers without manual operations. XSQML helps accomplish this by organizing data into entities and attributes. Xfi is an eXtensible Framework for Interoperability. It incorporates applications using XSQML and legacy repositories and supports the use of a generic vocabulary. The paper describes in detail the XSQML data structure transactions, schema transformations and transport layer as well as Xfi data structure and implementation. A prototype is provided as a viable solution to support web based inspection and software development tools.

Groupware Support for Requirements Negotiations in Distributed Software Development by Daniela Damian & Armin Eberlein

Distributed software development creates challenges in collaboration of and negotiations between software development team members. The importance of including all relevant decision makers and stakeholders in discussions involving distributed software development has led to the need for systems which can support communication over large distances. As well as ensuring good communication, one must look at other aspects of distributed negotiations. One specific question that was addressed in the presentation was: do conflict resolution and group performance change depending on whether the communication is face-to-face or distributed? It has been believed that face-to-face meetings result in the best level of performance. An empirical study at the University of Calgary answered such questions. It has been found that co-location of two negotiators helps, but is – surprisingly – not always beneficial. Future work includes looking at personality, gender and level of acquaintance as factors that may influence distributed requirements negotiations.

Distributed Software Development using Jini by Jörg Niere, Matthias Gehrke, Albert Zundorf

Members of software development teams must coordinate themselves concerning task scheduling and access to common documents such as source code. Part of this coordination can be accomplished through the use of version management and configuration management systems (VCM). In distributed software development teams, coordination must be done across countries and even the world. The authors discuss that current systems are not sufficient for this task. The paper uses three aspects as a basis for a distributed software development process with sound coordination. Using a VCM with optimistic locking, close interaction between the VCM and workflow and special offline project access allows for sound group coordination. The paper describes a mail based workflow engine that uses these aspects to enhance coordination. A 'distributed software development' (DSD) clerk is used to communicate information between members of the team. Reviews of modifications to the system are done anonymously but kept in a database. The paper also discusses some practical experiences and some problems that arose in implementation. A

stand-alone application, DSD Client, was implemented but required installation on every member's machine. A solution to use Jini is provided to allow different kinds of programs to connect automatically.

Preliminary Results of an Industrial EPG Evaluation by L. Scott, R. Jeffery, U. Becker-Kornstaedt

This paper presented a case study on the usefulness of electronic process guides (EPGs). As companies define and/or modify their processes, it is important that the processes and their changes be communicated to all involved in the process. The usual and often ineffective method of communication is a paper-based memo. The Internet, however, provides an effective method of communication regarding processes within an organization. The web already provides distributed software teams with much assistance and these are discussed briefly by the authors. The focus, however, is on an evaluation of an electronic process guide. The paper looks at what the EPG is used for and how it is used, what the outcomes were, benefits and possible improvements. An EPG consists of a project navigation tree, a main page and an individual page for every artifact, activity, role and tool. A survey was taken to evaluate the outcome of the EPG and the results were encouraging from all sides but improvements were suggested. One primary use of the guide was to provide access to template documents for process steps. Evaluations of the presentation of the guide concerned the process rather than the actual presentation of the guide.

Approaching Negotiation Automation for Software Development Companies by Boris Kötting, Frank Maurer

In any company, better performance can be achieved by distributing tasks within the company efficiently. The same is true of globally operating software development teams. A problem arises though, when a manager cannot assign tasks according to the employee's skill set as the required skills are not available in her local team. In terms of software development, one must make many decisions regarding varying topics, thus negotiating tasks is complex. The paper discusses automating negotiation for agents in a distributed company. A virtual marketplace is created to enable negotiation between different companies. The paper provides four different strategies, a Yes/No offer, a sealed bid, an English auction and finally negotiation. To support automatic negotiation a utility function is used with a lower and an upper threshold. Values are determined based on user settings. The agent first checks the negotiation type, then determines the proper course of action. The paper notes a key point that the details of the utility function should remain hidden from the user. Finally, the possibility of multi-bids is discussed briefly.

Process Model Integration in Internet-based Virtual Software Corporations by Quentin Mair & Zsolt Haag

The Internet and increasing sizes of software development teams have changed the environments in which software is being developed, and virtual software companies (VSC) are a prime example. Tool support is required for process management and the desire to integrate and interoperate process models has defined the need for a VSC component process. Identifying the atomic processes performed within one organization helps to integrate varying processes between VSC members. A Process Specification Language defines a neutral language for process specification and assists in the exchange of process information. Finally, the Resource Description Framework provides interoperability between tools that exchange information over the Internet. The paper looks at four advantages in using RDF to define PSL and discusses five parts of the PSL declaration. A mapping of PSL/RDF to enactable process models is described and deficiencies such as homogeneity and lack of an industrial case study are discussed.

Organizing committee

People from Germany, Canada and New Zealand organized the workshop:

Frank Maurer, Department of Computer Science, University of Calgary, maurer@cpsc.ucalgary.ca, Canada, (Primary contact).

Barbara Dellen, Avinci - The Know-How Company, Barbara.Dellen@avinci.de, Germany .

John Grundy, Department of Computer Science, University of Auckland, john-g@cs.auckland.ac.nz , New Zealand.

Boris Kötting, Department of Computer Science, University of Kaiserlautern, koetting@informatik.uni-kl.de, Germany.

References

[1] First Workshop on "Software Engineering over the Internet". Web proceedings available at http://sern.ucalgary.ca/~maurer/ICSE98WS/ICSE98WS.html

[2] Second Workshop on "Software Engineering over the Internet". Web proceedings available at http://sern.ucalgary.ca/~maurer/ICSE99WS/ICSE99WS.html

[3] Third Workshop on "Software Engineering over the Internet". Web proceedings available at http://sern.ucalgary.ca/~maurer/icse2000ws/ICSE2000WS.htm

[4] Forth Workshop on "Software Engineering over the Internet". Web proceedings available at http://sern.ucalgary.ca/~maurer/icse2001ws/ICSE2001WS.html