Available online at www.sciencedirect.com

SCIENCE DIRECT®

The Journal of Systems and Software 79 (2006) 912–925

The Journal of Systems and Software

www.elsevier.com/locate/jss

An empirical study of groupware support for distributed software architecture evaluation process

Muhammad Ali Babar [*], Barbara Kitchenham, Liming Zhu, Ian Gorton, Ross Jeffery

Empirical Software Engineering, National ICT Australia, University of New South Wales, Sydney, Australia

Received 6 December 2004; accepted 20 June 2005
Available online 2 February 2006

Abstract

Software architecture evaluation is an effective means of addressing quality related issues early in the software development lifecycle. Scenario-based approaches to evaluate architecture usually involve a large number of stakeholders, who need to be collocated for face-to-face evaluation meetings. Collocating a large number of stakeholders is an expensive and time-consuming exercise, which may prove to be a hurdle in the wide-spread adoption of disciplined architectural evaluation practices. Drawing upon the successful introduction of groupware applications to support geographically distributed teams in software inspection, and requirements engineering disciplines, we propose the concept of distributed architectural evaluation using Internet-based collaborative technologies. This paper presents a pilot study used to assess the viability of a larger experiment intended to investigate the feasibility of groupware support for distributed software architecture evaluation. In addition, the results of the pilot study provide some preliminary findings on the viability of groupware-supported software architectural evaluation process.

Keywords: Software architecture evaluation; Groupware systems; Empirical software engineering; Distributed software development; Software process improvement

1. Introduction

Software architecture (SA) evaluation is an effective mechanism for improving the quality of software intensive systems. The main objective of SA evaluation is to consider and address quality requirements at the SA level (Bass et al., 2003; Maranzano et al., 2005). There are various techniques and tools to assess the potential of the chosen architecture to deliver a system capable of satisfying desired quality requirements and identify potential risks. Most of the well-known SA assessment approaches are scenario-based methods (Ali-Babar et al., 2004) such as Architecture Tradeoff Analysis Method (ATAM) (Kazman et al., 1999), Software Architecture Analysis Method (SAAM) (Kazman et al., 1994) and Architecture-Level Maintainability Analysis (ALMA) (Lassing et al., 2002).

Scenario-based SA evaluation is a collaborative exercise that involves a number of stakeholders. Currently, it requires all the major stakeholders to be collocated for face-to-face (F2F) meeting to perform various activities, such as defining and refining business drivers, generating quality sensitive scenarios, and mapping the scenarios on to the proposed architecture. This is an expensive and time consuming process. Besides setting aside significant amount of time, stakeholders may have to travel if they are geographically distributed, which is highly likely as companies increasingly develop software using geographically distributed teams (Carmel and Agarwal, 2001; Herbsleb and Moitra, 2001; Mashayekhi et al., 1994; Perry et al., 2002). Organizational concerns about the cost and scheduling difficulties for collocating large number of stakeholders have been widely reported (Layzell et al., 2000; Perry et al.,

[*] Corresponding author. Tel.: +61 3 8374 5515; fax: +61 3 8374 5520.
E-mail addresses: malibaba@nicta.com.au (M.A. Babar), Barbara.kitchenham@nicta.com.au (B. Kitchenham), liming.zhu@nicta.com.au (L. Zhu), ian.gorton@nicta.com.au (I. Gorton), ross.jeffery@nicta.com.au (R. Jeffery).

0164-1212/$ - see front matter © 2005 Elsevier Inc. All rights reserved.
doi:10.1016/j.jss.2005.06.043

2002). These difficulties may hinder the wide-spread adoption of SA evaluation practices.

In an attempt to find a cost effective and efficient alternative to F2F meeting-based SA evaluation, we suggest that Internet-based collaborative technologies may provide a mechanism of addressing some of above-mentioned issues (Collaborative technologies include web-based applications that support collaboration, e.g., groupware systems, collaborative and CSCW applications, etc.). Researchers and practitioners in various sub-disciplines of software engineering (such as requirements engineering, inspections and others) have successfully evaluated groupware supported processes as a promising way to introduce software shift-work, minimize meeting costs, maximize asynchronous work and conserve a number of precious organizational resources (Boeham et al., 2001; Gorton et al., 1996; Halling et al., 2001; Perry et al., 2002). Drawing on the positive results of using groupware systems in similar domains, we propose that the collaborative applications can be used to improve the SA evaluation process without compromising the quality of the artifacts and results.

However, there are a number of important issues that should be explored before making any conclusive claim about the effectiveness of the collaborative applications for distributed SA evaluation. For example, we need to understand the changes required in the existing SA evaluation approaches to allow for distributed environments. We also need to identify appropriate collaborative technologies to support distributed SA assessment and gain a better understanding of how they facilitate or hinder social processes. We intend to use experimentation to study these issues (Perry et al., 2002).

In order to evaluate the effectiveness of distributed SA evaluation, we have designed an empirical research program based on a framework of experimentation (Basili et al., 1986) and guidelines provided in Kitchenham et al. (2002). The experimental program consists of a pilot study followed by a large-scale experiment. This paper reports the results of the pilot study from two viewpoints. Firstly, the pilot study has provided some initial information about the use of groupware to support SA evaluation in distributed arrangement and secondly it has allowed us to refine our subsequent experimental program.

The salient features of this paper are:

- It briefly discusses the concept of distributed SA evaluation using collaborative technologies.
- The pilot study results provide an initial assessment of the effect of using distributed meeting for SA evaluation activities.
- We show how the results of the pilot study can be used to assess the number of experimental units needed in experiments.

The remainder of the paper is organized as follows. In the next section, we briefly review the work that has motivated our research program. We then present our idea of a distributed SA evaluation process. We describe experiment details in Section 4. Analysis and interpretation are presented in Section 5. We close the paper with the conclusions and plans for future research.

2. Background

2.1. Software architecture evaluation

Recently it has been widely recognized that quality attributes (such as maintainability, reliability, etc.) of complex software intensive systems largely depend on the overall SA of such systems (Bass et al., 2003). Since SA plays a vital role in achieving system wide quality attributes, it is important to evaluate a system's architecture with regard to desired quality requirements. SA community has developed several methods to support disciplined architecture evaluation practices. Most of the mature architectural evaluation methods are scenario-based such as Architecture Tradeoff Analysis Method (ATAM) (Kazman et al., 1999), Software Architecture Analysis Method (SAAM) (Kazman et al., 1994) and Architecture-Level Maintainability Analysis (ALMA) (Lassing et al., 2002).

Although there are differences among these methods (Ali-Babar et al., 2004), we have identified five common activities by comparing four main approaches to evaluate architecture (Ali-Babar and Gorton, 2004). Fig. 1 presents these five activities, which can make up a generic scenario-based SA evaluation process that can be supported by a groupware application. Following is a brief description of each activity in this generic SA evaluation process:

1. *Evaluation planning and preparation*—This is concerned with allocating organizational resources and setting goals for evaluation, selecting stakeholders, preparing

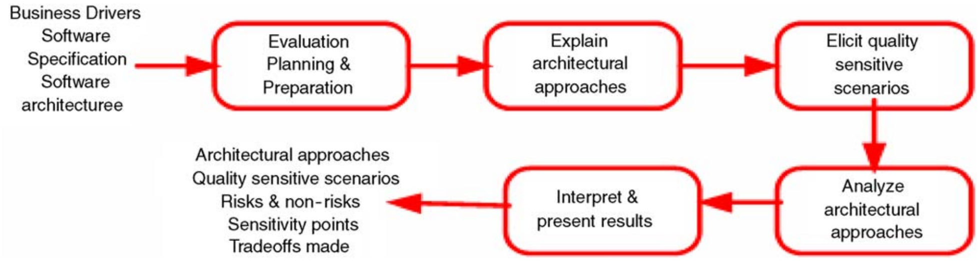

Fig. 1. A generic software architecture evaluation process.

inputs and deciding on the valuation team. This activity is vital as it ensures that required documentation and resources are available and provides the roadmap of the process and identifies expected outcomes.

2. *Explain architectural approaches*—During this activity, a software architect presents the architecture of the system under consideration to the evaluation team and explains architectural decisions and how those decisions can satisfy business goals. He/she also identifies the known architectural style or patterns being used in the designed architecture and justifies their use.
3. *Elicit quality sensitive scenarios*—The purpose of this activity is to develop scenarios to characterize the quality attributes for a system. For example, a maintainability quality attribute can be specified by change scenarios. Scenarios are also prioritized before being used for architecture evaluation. Sometimes scenarios are also ranked according to their level of complexity.
4. *Analyze architectural approaches*—This activity is aimed at analyzing architectural approaches with respect to the scenarios developed during the previous stage. Findings are categorized into risks, non-risks, trade-off points, sensitivity points and others and rationale is documented.
5. *Interpret and present results*—This activity is concerned with summarizing the results of all previous activities, interpreting the deliverables and presenting results to the sponsors.

Architectural assessment normally requires expertise and knowledge of different quality attribute experts such as performance engineers and usability specialists. Furthermore, the affect of a particular quality attribute cannot be analyzed in isolation as quality attributes have positive or negative influences on each other, which may require trade-offs among quality attributes. All these activities require group discussions and decision making processes, which necessitate meetings.

However, we argue that most of these activities do not necessarily need to be performed in a co-located arrangement. Rather, most of them can be done in asynchronous mode without affecting the quality of the outcome. The need for synchronous discussion can be supported by an electronic meeting system (EMS) (Nunamaker et al., Winter, 1996–1997).

2.2. Groupware systems

Groupware systems are computer-based applications that support communication, collaboration, and coordination among a group of people working towards a common goal; much of the time these people are geographically distributed (Ellis et al., 1991). A groupware system usually has a very diverse set of tools (such as E-mail, audio video conferencing, calendar, content management, workflow management, electronic meetings) that complement each other (Nunamaker et al., Winter 1996–1997). These systems have emerged over the past decade as mainstream business applications to support a variety of problem solving and planning activities in a wide variety of organizations. A key benefit of groupware systems is to increase efficiency compared with F2F meetings by creating positive changes in group interactions and dynamics (Genuchten et al., 2001; Nunamaker et al., Winter, 1996–1997).

Groupware systems have proven effective in reducing the time and resources required to complete a project by minimizing the inter-activity intervals and delays (Nunamaker et al., Winter 1996–1997; Perry et al., 2002). Researchers have shown that teams using groupware systems can reduce their labour costs by up to 50% and project cycle times by up to 90% (Grohowski et al., 1990). Groupware systems also have the potential to effectively support meeting processes involving large groups (Nunamaker et al., 1991) and to increase the number and quality of the ideas generated (Valachich and Dennis, 1994). Groupware systems also provide a set of tools that can efficiently process large amount of information consumed or generated during meetings. Other notable attributes of such systems include anonymity, simultaneity, process structuring, process support, and task support (Nunamaker et al., 1991).

2.3. Groupware support for inspection and requirements engineering processes

It has been shown that F2F meetings for software inspections incur substantial cost and lengthen the development process (Perry et al., 2002). Some studies have called into question the value of F2F inspection meetings (Porter and Johnson, 1997). Studies have also indicated that computer tools, including groupware, may improve inspections (Sauer et al., 2000). Groupware-supported inspections have been successfully evaluated as a promising way to minimize meeting costs, maximize asynchronous work and conserve a number of precious organizational resources (Halling et al., 2001; van Genuchten et al., 2001). Moreover, it has also been shown that the software inspection process can be improved with group process support (Tyran and George, 2002).

Requirements engineering (RE) community has also successfully used groupware applications to enable distributed teams of stakeholders to perform different tasks of RE process. For example, Liou and Chen (1993) integrated joint application development (JAD) and group support systems (GSS) to support requirements acquisition and specification activities. Damian and her colleagues reported successful experiments with using a web-based collaborative tool to support requirements negotiation meetings (Damian et al., 2000). Boehm and his colleagues developed a groupware tool to support their EasyWinWin requirements negotiation methodology (Boeham et al., 2001) and integrated a case tool to improve the support for requirements engineering tasks (Gruenbacher, 2000).

3. Distributed SA evaluation process

We have mentioned that evaluation of software architectures is usually performed by stakeholders in a F2F meeting. Collocating a large numbers of stakeholders is an expensive and time-consuming exercise, particularly for geographically distributed teams of software developers. We are mainly interested in finding and assessing an effective and efficient way of enabling physically dispersed stakeholders to participate in architecture processes without having to travel, while improving the overall process of architecture design and evaluation. Previous experimental studies found that groupware applications provide an appropriate support mechanism to introduce the shift work concept in software development activities to exploit organizational knowledge (mainly workforce) distributed across different time zones and geographical locations (Gorton et al., 1996). This work also identified the technological and organizational issues, which existed at that time, that needed to be addressed in order to maximize the "over night gains" and minimize the "over night losses" (Gorton and Motwani, 1996). Others have reported that Internet and groupware systems have played a vital role in improving collaborative processes in a number of disciplines by minimizing dysfunctional behaviour and enhancing group productivity (Griffith et al., 2003; Grohowski et al., 1990; Nunamaker et al., Winter, 1996–1997; Piccoli et al., 2001).

Based on these previous results, we are developing the concept of groupware supported distributed software architecture evaluation processes, which are aimed at addressing a number of above-mentioned logistical issues that characterize the current evaluation approaches. We posit that a number of activities (such as evaluation planning, scenario gathering, scenario prioritization, and scenario mapping,) in the general process model of SA evaluation (Fig. 1) can successfully be performed in a distributed environment using web-based groupware applications.

In the proposed process, groupware supported electronic workspaces are used to enable geographically dispersed stakeholders perform the various tasks in architecture evaluation. These findings and our previous successful trials of using collaborative tool to support distributed software development teams (Gorton et al., 1996; Hawryszkiewycz and Gorton, 1996) give us confidence that groupware systems can greatly benefit SA evaluation.

4. Empirical research program

4.1. Introduction

We intend to evaluate the effectiveness of distributed SA evaluation by means of a series of laboratory experiments. Agarwal et al. (1999) suggest that laboratory-based experiments are appropriate when the cumulative body of knowledge related to a specific phenomenon is limited, as is the case with the groupware supported distributed SA evaluation process. If our ideas prove successful in a laboratory setting, we will then attempt to evaluate the proposed approach in industrial settings.

During the first phase of the empirical studies, we limit our inquiry to scenario profiles development activity of SA evaluation process. We focus on creating scenario profiles activity for several reasons. Developing quality sensitive scenario profiles is considered the most expensive and time-consuming activity of the SA evaluation process. The accuracy of the results of SA evaluation exercise is largely dependent on the quality of the scenarios used (Bass et al., 2003; Bengtsson and Bosch, 2000). Thus, our controlled experiments are designed to compare the performance of Collocated Groups (CGs) and Distributed Groups (DGs) based on the quality of the scenario profiles, developed by both types of groups.

Our laboratory-based experimental program consists of a pilot study followed by a large-scale experiment. The pilot study itself was run as a formal experiment using an experimental design that was being considered for the subsequent experiments. In the following sections, we describe the design, conduct, and results of the pilot study.

4.2. Research questions and hypotheses

The major purpose of research program is to gain an understanding of the opportunities and challenges of conducting SA architecture evaluation in a distributed environment using collaborative technologies. There is no solid theory on distributed SA evaluation process. Hence we relied on the literature reporting the success and failure of introducing groupware for organizational processes in general (Grudin, 1994; Grudin and Palen, 1995; Nunamaker et al., 1991) and software processes in particular (Genuchten et al., 2001; Sakthivel, 2005) to developed the following research questions:

- How much do scenario profiles created by collocated groups (CGs) vary from scenarios profiles created by distributed groups (DGs).
- What changes should be made in the existing SA evaluation methods to support geographically distributed stakeholders?
- What type of features should a groupware application provide to successfully support a distributed architecture evaluation process?
- How does a distributed arrangement affect the socio-psychological and organizational aspects of the evaluation process?

The aims of the pilot study were to:

- Provide initial estimates of the effect size and variability, so that we could perform a power analysis to estimate the sample sizes necessary for the various experimental

designs that could be adopted in the subsequent experiments. In particular, we wanted to check whether a parallel or sequential design was necessary.
- Ensure that the experimental materials and protocols (e.g., training times, interaction with subjects, tool facilities, etc.) were appropriate for subsequent experiments.
- Provide the student researchers with experience conducting laboratory experiments.

4.3. Experiment design

We used an AB/BA cross-over design for our pilot study (Senn, 2002). The design is shown in Table 1. In a cross-over study design, the participants are assigned to sequence of treatments in order to study differences between individual treatments. Our pilot study design is a balanced design in which each experimental unit (i.e., group of three participants) performed two scenario development tasks. Half of the groups used a F2F meeting arrangement for their first task followed by a distributed arrangement for the second task. The other groups used a distributed meeting for the first task and a F2F task for the second meeting.

The advantages of cross-over designs are that they require fewer subjects than parallel designs and when there is no interaction between treatments and order, they are resilient to subject differences and maturation effects. The most significant disadvantage of a cross-over design is that it is inappropriate if there is a large interaction between treatment and order. A treatment-order interaction occurs when doing one treatment first has an effect that is different from doing the other treatment first. There is no interaction if there is an order effect, such as learning effects, that influences both treatments equally (Kitchenham et al., 2004).

The independent variable manipulated by this study is type of meeting arrangement (group interaction), with two treatments, F2F meeting (CGs) and distributed meeting (DGs).

Dependent variable is quality of the scenario profiles developed by CGs and DGs.

4.4. Participants

We required the participants of this study to be similar to the potential participants of our future series of experiments (third and fourth year students of software engineering or computer engineering). We invited graduate researchers in the areas of information and communication technologies to participate. We selected 24 participants, who possessed different values of the selection criteria, e.g., length of work experience, exposure to web-based content management system, familiarity with functional and non-functional requirements concepts, etc. We did not choose those volunteers who had prior knowledge of our research program or were associated with the course during planning and design stages.

Table 1
Systems and group interaction assignment

Material	Treatments	
	F2F arrangement	Distributed arrangement
Zwiki system	G2, G5, G6, G8	G1, G3, G4, G7
InspectAnyWhere	G1, G3, G4, G7	G2, G5, G6, G8

4.5. Experimental materials and apparatus

4.5.1. Software requirements specifications

This study used SRS for two different applications, a web-based content management system and a web-based software inspection tool. The former is an open source web-based content management system called Zwiki. The later is a web-based inspection support tool called Inspect AnyWhere (Lanubile and Mallardo, 2002). This system provides collaborative features to support the different activities of software inspection process, e.g., planning, defect detection, defect collection, follow up, etc. We prepared a simplified version of the SRS of each system together with some more detailed descriptions and screen shots.

4.5.2. Collaborative application

The groups using distributed meeting arrangements were required to brainstorm and structure their scenario profiles using a web-based groupware system. We selected a generic collaborative application, LiveNet, based on its features and ease of availability for research purposes. LiveNet provides a generic workflow engine and different features to support collaboration among geographically distributed members of a team. LiveNet enables users to create workspaces and define elements of a particular workspace. LiveNet also supports emergent processes. For further details, see Biuk-Aghai and Hawryszkiewyez (1999) and Hawryszkiewycz (Last accessed on 8th June, 2004).

4.6. Measuring quality of scenario profiles

In order to evaluate the effectiveness of a distributed arrangement compared with F2F arrangement, we needed to compare the quality of the artifacts, scenario profiles, developed by both CGs and DGs. We used a method of ranking scenario profiles to measure their quality (Bengtsson, 2002). In order to use this method, the actual profiles for each group must be recoded into a standard format for analysis. The quality of each of the recoded profiles was evaluated by comparison with a "reference profile" constructed from all the recoded profiles identified for a particular SRS. The construction of the recoded profiles relied on researchers coding free-format text. The reliability of the coding was assessed by comparing the profiles obtained independently by two researchers. The method of measuring the quality of scenario profile is described in Bengtsson (2002).

4.7. Experimental validity

4.7.1. Threats to internal validity

Wholin et al. (2000) identify four main threats to internal validity: selection effects, maturation effects, instrumentation effects, and presentation effects. These threats are a problem for quasi-experiments (Shadish et al., 2001), but are not relevant when using a cross-over design with random allocation to sequence which is designed to avoid these problems (Senn, 2002). However, using a cross-over design is problematic if there is an interaction between sequence and treatment. Since the treatment is about meeting structure not about the intellectual task of constructing the scenarios, we believe the probability of a significant interaction effect is low.

Another threat to the internal validity of our experiment can be the method used to measure the quality of the scenarios developed by the participants. This method is highly dependent on the way the data are analyzed and interpreted. The method was developed and validated for another experiment and various threats to its internal validity have been discussed and addressed in Bengtsson (2002). However, one of the potential threats, skill, knowledge, and bias of reference profile builder, associated with this method was addressed by having two independent reference profile developers.

4.7.2. Threats to external validity

The major threat to external validity of our pilot project is that postgraduate students will be systematically different from the third and fourth year undergraduates who will take part in the subsequent large-scale experiments. We selected such postgraduate students who had no experience of SA evaluation, and little experience of quality sensitive scenarios. In these respects the postgraduates should be similar the undergraduates. In addition, for our power analysis, we have assumed that the groups of postgraduate students would exhibit less variability than groups of undergraduate students.

4.8. Pilot study operation

The flow of the pilot study implementation is shown in Fig. 2 and the experimental design and execution plan is shown in Table 2.

Selection: Selection of the 24 participants took place prior to the experiment.

Briefing and training: After selection the participants were given a 30 min training session to provide an overview of the collaborative tool, the software architecture evaluation process, the process of generating quality sensitive scenarios, and the software inspection process. Furthermore, the participants were shown screen dumps of the systems for which they were supposed to develop change scenarios. They were also given time to familiarize themselves with the collaborative tool, LiveNet. However, our study did not require the participants to have any experience in SA evaluation. The duration and format of our training was designed to make the participants representatives of most of the stakeholders involved in real world SA evaluation, where stakeholders normally receive minimum training.

A document describing the content management system, inspection management system and example scenarios was also made available to the participants during the experiment.

Assignment to groups: Prior to the experiment we identified eight group names and allocated each group to each

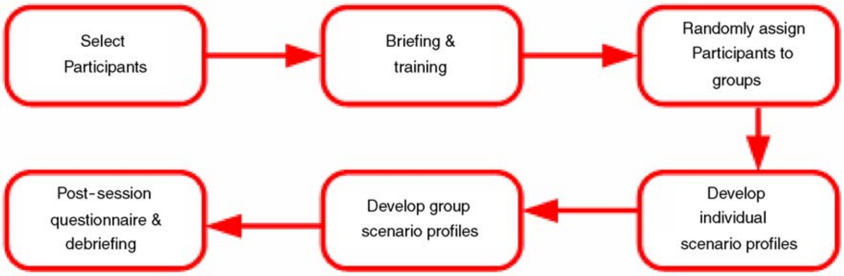
Select Participants
Briefing & training
Randomly assign Participants to groups
Develop individual scenario profiles
Develop group scenario profiles
Post-session questionnaire & debriefing

Fig. 2. A diagram showing the flow of the experiment.

Table 2
Experimental study execution plan

Amount of time (min)	Distributed group	F2F group	Type of system
30	A brief introduction and training		
15	Develop individual profile	Develop individual profile	Zwiki system
30	Develop group profile (g1, g3, g4, g7)	Develop group profile (g2, g5, g6, g8)	
15	Develop individual profile	Develop individual profile	InspectAnyWhere
30	Develop group profile (g2, g5, g6, g8)	Develop group profile (g1, g3, g4, g7)	
15	Post-session questionnaire and debriefing		

treatment and condition combination randomly (name out of the hat). Three subjects were allocated to each group on the day of the experiment without using any formal randomisation process. However, we believe there was no bias in subjects' allocation.

Developing individual scenario profiles: Participants were given a simplified version of requirements for a web-based content management system (Zwiki) and asked to developed system change scenarios individually for 15 min. When 15 min of time had passed the profile of individuals were collected, photocopied and returned to them. All the participants were asked to join their respective groups to develop group scenarios. All eight groups were randomly assigned to distributed (intervention) and F2F (control) settings and asked to develop group scenario profiles for 30 min.

Group profile development: Participants were instructed to follow a process to develop group scenario profile. For the web-based content management system, groups g1, g3, g4, and g7 developed their group scenario profiles in a distributed arrangement using the collaborative tool. The only means of synchronous communication was the chat-room of the collaborative tool. Groups g2, g5, g6, and g8 developed their group profile scenarios in F2F meeting arrangement. After 30 min of time elapsed, the group scenario profiles were collected.

Iteration of profile development: After a short a break, the process started again for a web-based system to support distributed software inspection, InspectAnyWhere. The only difference for creating change scenario profiles for this system was during the group activity, when the groups which worked in a F2F arrangement for content management system were asked to work in a distributed arrangement and groups which had worked in the distributed arrangement were asked to work in the F2F arrangement. The amount of the time allowed for individual and group tasks was the same, 15 and 30 min, respectively.

During this part of the experiment, we encountered a problem. The members of one group mistakenly used the SRS for the Zwiki system as a DG, when they had already used that SRS as a CG. The results from this group were therefore invalid and were omitted from the analysis.

Post-session questionnaire: After developing individual and group scenario profiles for both systems, the participants filled a post-session questionnaire to provide both their demographic information and their subjective experiences with both types of meeting arrangements.

Debriefing: The experiment finished with a debriefing session, which was aimed at explaining the objectives of the study and answering participants' questions on any aspect of the research program.

4.9. Data collection

Three sets of data are important to our study; the individual scenario profiles, group scenario profiles, and questionnaire filled by all the participants at the end of the experiment. Though our results are based on the comparison of group scenario profiles, we needed both individual as well as group scenario profiles to develop the reference profile. Participants were encouraged to structure their scenarios using a scenario structuring framework (Bass et al., 2003). However, they were allowed to compose scenarios as free text as long as those scenarios were concrete enough to characterize the future changes in the existing systems. Participants submitted files of their individual scenario profiles, group scenario profiles and chat logs.

Finally, each participant filled out a questionnaire at the end of the study. The questionnaire is designed to collect information on the participants' attitude towards F2F verses distributed SA evaluation. Most of the questions required the participants to respond by circling a choice on a three point scale and providing a short explanation of their respective response to a particular question. The questionnaire also collected demographic data such as experience level, gender, age, etc. on a nominal scale.

Information collected during the experiment can be tracked by an identification code present on the individual scenario profiles, group scenario profiles, and post-session questionnaire. Thus, the data is not anonymous. In our judgment, lack of anonymity is not problem in our type of experiment as mentioned in Bengtsson (2002). Furthermore, our future experiments are planned to be executed as part of academic assessments, where being able to identify individuals that create interesting data points is considered more important than the risk of getting unreliable data because of lack of anonymity.

5. Pilot study results

5.1. Reference profiles

We gathered 73 scenarios from 32 profiles for the Zwiki system and 60 scenarios from 28 profiles for the InspectAnyWhere system. We discarded four data points, three individual and one group, for building the reference profile and one data point, group only, for data analysis because the group did not follow the correct experimental procedure. That means we collected 133 scenarios from 60 profiles instead of 64 profiles. We developed two reference profiles, one for each system, to rank the scenario profiles developed by the participants. The process of developing a reference profile to measure the quality of the scenarios has been extensively documented in Bengtsson (2002).

We will provide a brief description of this process here. To build a reference profile, we identified unique scenarios and put them together. We noted the frequency for each unique scenario by counting the number of times it had been reported in various profiles. Then, we calculated a score for each scenario profile developed during the experiment by summarizing the frequency of each scenario in the scenario profile. Tables 3 and 4 show the top 10 scenarios of each reference profile.

Table 3
Top 10 scenarios for Zwiki system

No.	Scenarios in reference profile	Frequency
1	System shall provide various search functions	17
2	Pages have different types of permissions	15
3	User shall have different privileges	14
4	User shall be able to undo changes	12
5	System shall be able to handle more objects	10
6	User shall upload and download documents	10
7	System shall provide a updatable sitemap	8
8	Multiple ways of viewing change log	7
9	System supports more naming conventions	7
10	Comment on page change place	7

Table 4
Top 10 scenarios for InspectAnyWhere system

No.	Scenarios in reference profile	Frequency
1	Different levels of access control	19
2	User notification mechanism shall be provided	12
3	Different versions of artifacts managed	10
4	Online discussion facility that can be stored	9
5	User can perform planning tasks online	9
6	Change logs annotation shall be viewable	7
7	Feedback on artifacts and resources provided	7
8	Audio and video channels are available	7
9	Status of artifact being inspected viewable	6
10	Action can be undone	6

5.1.1. Inter-rater agreement

The method used to measure the quality of scenario profiles requires marking against a reference profile, which may be affected by the subjective judgement of the coder. In order to address this issue, two independent makers performed the task separately. In addition to assigning scenarios to reference profile, each coder could nominate new scenarios or split old scenarios in the reference profile developed by the other coder. Any disagreement regarding the scenario profile was discussed and resolved before the final marking of the scenario profiles.

5.2. Analysis of results

The results of the experiment are shown in Table 5. The pilot project can be analyzed simply in terms of the difference between the quality of the profile obtained from the F2F groups and the distributed groups. Although the groups used different SRS in each period, a balanced cross-over design is resilient to systematic differences between periods as long as there is no period treatment interaction (Senn, 2002). We anticipate a period effect as a result of two factors: firstly, the SRS used for the first evaluation was different from the SRS used for the second evaluation; secondly, the subjects would have more experience of the developing quality attribute scenarios when they develop scenarios for the second system. However, these effects should be the same for groups in each sequence implying no treatment by period interaction.

Summary statistics for the two sequence groups are shown in Table 6. The analysis is complicated by the drop out which leaves unequal sample sizes in each treatment sequence group. However, it can still be analyzed using the treatment group means and adjusted within-groups variance (Senn, 2002, Section 3.6). The result of this analysis indicates that the mean effect of the treatment (i.e., the mean difference between the quality of scenario profiles from distributed meeting and F2F meeting groups) after adjustment for the period effect is 18.53 with a standard error of 6.33. This value is significantly different from zero ($p = 0.037$, 95% confidence interval 2.30–34.87). This result suggests that the effect of a distributed meeting is not simply equivalent to a F2F meeting but is actually superior in terms of the quality of the resulting scenario profile.

5.3. Power analysis

We performed a power analysis to confirm that we would have enough participants in our subsequent experiments to properly test our experimental hypotheses. We

Table 5
The quality marks for scenario profiles

Group ID	Treatment used first	Distributed treatment	F2F treatment	Profile quality difference
G2	F2F	87	47	40
G5	F2F	96	88	8
G6	F2F	73	59	14
G8	F2F	Discarded	76	Discarded
G1	Distributed meeting	67	27	40
G3	Distributed meeting	85	70	15
G4	Distributed meeting	79	74	5
G7	Distributed meeting	65	59	6

Table 6
Summary statistics for treatments

Sequence	Number of groups	Quality profile difference (distributed-F2F)	Standard deviation	Standard error
F2F first	3	20.67	17.01	9.82
Distributed team first	4	16.5	16.03	8.15

assumed a Normal distribution for the mean difference in quality profile for the purpose of the power analysis. The central limit theory confirms that a linear combination of any random variables will be approximately Normal. For simplicity, we assume:

- Our null hypothesis is that there is no difference between the quality of the scenario profile for each treatment group. This is a one-sample test because the analysis is based on the difference between the two values for each experimental unit.
- The significance level is 0.05.
- The power is 0.80.

In order to undertake a power analysis it is necessary to consider the nature of the alternative hypothesis. One alternative hypothesis is that the difference between the quality of the scenario profile for the distributed meeting is superior to the quality of the scenario profile for the F2F meeting. Another alternative hypothesis is that the difference between the quality of the scenario profile for the distributed meeting is inferior to the quality of the scenario profile for the F2F meeting. The later alternative hypothesis is unlikely given the pilot study results but is important with respect to the aims of the full experiment. In both cases, this form of alternative hypothesis implies a one-sided test. Alternatively, we could simply use an alternative hypothesis that the difference between the quality profiles for each treatment is different from zero and use a two-sided test.

The choice of appropriate alternative hypotheses depends on the goals of our experiment. Our goal is to assess whether the benefits of distributed meetings (in terms of reduced costs) are not outweighed by potential losses (in terms of significantly poorer quality scenario profiles). Thus, we do not need to show that distributed meeting arrangements are superior to collated arrangement. We only need to show that they are not significantly worse than F2F meeting arrangements. For that reason our preference is for a one-sided alternative hypothesis that the distributed meeting arrangement is 10 units worse than the F2F meeting arrangement. Our assumption that distributed meeting arrangements are potentially useful will be supported if we are unable to reject the null hypothesis.

The results of the power analysis for one-sided are shown in Table 7. The values were calculated using the trial version of the StudySize application (Olofsson, 2004). In both cases, sample sizes of 50 are conservative compared with a sample size of 6 that is consistent with the values obtained in the pilot study (i.e., mean difference ≈18 and standard deviation ≈17). However, if we assume that undergraduate students will have a larger skill range than postgraduate students, a sample size of about 50 will protect against an increase of about 10 in the standard deviation.

5.3.1. Power analysis for a simple randomised experiment

In practice, we expect to have about 50 participant groups available for our subsequent experiments, so it is also worth considering whether it is necessary to perform a cross-over trial, rather than a simple randomised experiment without repeated values. This is equivalent to comparing the two treatment groups based on the values obtained from the first task, as shown in Table 8.

Analysis of the data in Table 8 does not allow us to reject the null hypothesis of no difference between treatments since the difference is 6.5 with a standard error of 10.25. In this case, we are concerned with having adequate power to reject the null hypothesis, if the F2F meetings are really superior which requires a one-sided test. However, because we are analysing the difference between the treatment groups directly we need a two sample power analysis as shown in Table 9. Table 9 illustrates how large our sample size needs to be for simple randomised experiments. Given the ratio of standard deviations (i.e., approximately 2) observed in pilot study, we would need nearly 1000 experimental units (i.e., groups of three students) to be sure of detecting a difference of about 5 in favour of the F2F meeting treatment. This confirms that our subsequent experiments must use a cross-over design to achieve an acceptable experimental power.

Table 8
The quality marks for each profile for the first trial in the sequence

Group ID	Treatment	Profile quality
A	F2F	47
A1	F2F	88
D1	F2F	59
D	F2F	76
Mean		67.5
Standard deviation		18.12
B	Distributed meeting	67
B1	Distributed meeting	85
C	Distributed meeting	79
C1	Distributed meeting	65
Mean		74
Standard deviation		9.59

Table 7
Power analysis assuming a one-sided test with one treatment group

Difference	Sample Size for sd = 30	Sample size for sd = 20	Sample size for sd = 15	Sample size for sd = 10
5	223	99	56	25
10	56	25	14	6
15	14	6	6	na
20	9	4	3	na

Table 9
Power analysis assuming a one-sided test and two sample analysis

Difference	Sample size for each group for sd = 30	Sample size for each group sd = 20	Sample size for each group sd = 15	Sample size for each group sd = 10
5	1113	495	278	124
10	278	124	70	31
15	124	55	31	14
20	70	31	17	8

5.3. Analysis of post-session questionnaire

Analysis of the post-session questionnaire revealed a contradiction between the quantitative analysis and the opinions of the participants. The majority of participants preferred the F2F meeting arrangement and felt they performed better in the F2F meeting arrangement. Table 10 summarizes the results of the responses to different questions designed to gather self-reported data.

These findings from the self-reported data becomes more interesting when we take into account the fact that all of the participants were extensively using Internet-based collaborative tools (e.g., NetMeeting, Yahoo Messenger) for professional and personal purposes but there was hardly any support for groupware supported distributed meeting arrangement among them. The participants provided a number of reasons for disliking the groupware supported meeting arrangement, such as a lack of body language, F2F was more natural and conventional, typing problems, slow collaboration, time lag in communication, and so on. However, the findings based on the self-reported data are consistent with findings of the several studies, which found that quality of decisions made in groupware supported arrangement was better or equal to the decision made in F2F arrangement but participants' attitudes (satisfaction, perceived effectiveness, and cohesiveness) was usually lower in groupware supported decision making arrangement (Dennis and Garfield, 2003; Fjermestad, 2004).

Table 10
Summary of questionnaire responses

Question 1	22 (92%) of participants believed they performed better in the F2F arrangement
Question 2	18 (75%) of participants believed the group performed better in F2F arrangement
Question 3	15 (62%) of participants believed that the collaborative tool had a negative affect on group discussion 6 (25%) thought it had a positive effect 3 (13%) reported no effect
Question 4	15 (62%) of participants found F2F meeting arrangement more efficient 7 (33%) participants thought both types of meeting arrangement equally efficient 1 (4%) participant found the distributed meeting arrangement more efficient In this context efficiency was defined to be the number of individual scenarios discussed and integrated into group scenarios and the number of new ideas and scenarios developed during team meetings
Question 7	19 (79%) of participants preferred F2F meetings 3 (13%) had no preference 1 (4%) preferred distributed meetings

Table 11
Summary of findings from self-reported questionnaire-based data

Main themes	Frequencies	Some common comments
Non-technical		
F2F helps improve comprehension and clarity	27	F2F saves time by getting the instant response and it helps come up with new ideas and comprehends others ideas
Body language facilitate discussion	15	F2F is more convenient as you do not have to think about spelling mistakes and speaking is much faster than typing
F2F discussion conventional and fast	14	
Technical		
Tool decreases efficiency because of time lag	20	Collaborative tool does not give more benefit rather it makes discussion less efficient because of time lag
Lack of features, e.g., audio/video/asynchronous chat	19	Floor control protocol needs to be strictly implemented to control the flow of discussion and giving everyone opportunity
Lack of implementation of floor control protocols	14	
Slow typing speed or lack of typing skills	10	

In order to further probe the self-reported data, we encoded the participant's explanation for frequency analysis. We used two level of encoding scheme. The first level allocated the reasons for disliking tool-based meeting arrangements to technical and non-technical. Since each respondent provided more than one reason, thus allocation was not exclusive to one single code only. The second level allocated the responses to main themes that we identified in the responses and this allocation was not exclusive either as some of the respondents gave more than one reason for each of the non-technique or technical categories. For instance, lack of body language and being habitual of F2F meetings can be allocated to the non-technical code first and then Body language and F2F discussion conventional and fast themes. Table 11 present the findings.

6. Conclusion and future work

Software architecture evaluation is an effective approach for addressing quality related issues and identifying risks early in the development lifecycle. Existing evaluation methods requires collocating large number of stakeholders for F2F meetings. Collocating stakeholders is difficult and expensive to organize, particularly in distributed software development environments. Encouraged by the successful implementation of web-based groupware supported processes for software inspections and requirements negotiation (Boeham et al., 2001; Genuchten et al., 1997–1998), we have proposed a concept of groupware supported distributed SA evaluation process, which does not require the stakeholders to be physically co-located. Our proposed process is expected to address a number of logistical issues that characterize current SA evaluation approaches by taking advantage of the Internet-based groupware technologies.

We are undertaking an experimental research programme aimed at evaluating groupware supported distributed software architecture evaluation. The experimental program involves an initial pilot study that will be followed by a series of larger-scale experiments. In this paper we have reported the results of the pilot study. The pilot study has provided us both with information about how to undertake the larger-scale experiments and a preliminary evaluation of our research questions.

From the viewpoint of our future experiments, power analysis has confirmed that we must use a cross-over experiment to have sufficient power to detect a difference between meeting arrangements if one exists. Furthermore, 50 experiment units (i.e., groups of three subjects) will be a sufficient sample size that allows for more variable results from undergraduate students.

There appear to be no major problems with our experimental materials in terms of understanding the SRS's or the scenario development task. We have made some minor changes in the training material, SRS and increased the allocated time period for each task based on the feedback provided by the participants. One potential problem is that participants felt that the technology used to support distributed meetings had substantial limitations. This could imply that our experiment may be biased against the distributed meeting arrangement.

The results obtained in this study pose some problem for hypothesis formulation. The goal of distributed SA evaluation meetings is to avoid the overheads associated with F2F meetings and for that purpose it is sufficient to demonstrate that the distributed meeting arrangement does not decrease the quality of scenario profiles. However, our current results suggest that the quality of scenarios obtained from distributed meetings are better than those obtained from F2F meetings. Nonetheless, given the possibility that the results in this experiment may be atypical, we suggest the hypothesis remain:

Null hypothesis: That there is no difference between the quality of scenario profiles obtained from F2F meetings and distributed meetings.

Alternative hypothesis: The quality of scenarios obtained from F2F meetings is at least 10 units better than the quality of scenarios obtained from distributed meetings.

From the viewpoint of our experimental hypotheses, the results of our pilot study suggest that distributed meetings for SA evaluation are at least as good if not better than F2F meetings, although individual participants were not as satisfied with distributed meetings as with F2F meetings. This provides initial support for the original hypothesis and indicates that a further experiment is worthwhile. However, it also suggests that we need to look into the ways of improving the experiences of the participants of groupware supported distributed meetings.

In addition, there are issues with external validity (i.e., whether our experiments can be generalised to industrial situations) that have not been addressed by the pilot study and will therefore affect our subsequent experiments:

- Whether or not the participants (i.e., third and fourth year students) are representative of people who undertake scenario development in industry. In fact students may be more technically-oriented than stakeholder groups usually asked to perform such activities, so better able to manage the technology used to support distributed meeting arrangements.
- Whether our SRS's are representative of the requirements documents used in industrial architecture evaluations. They were certainly smaller and simpler than might be expected in industrial situations. However, there is no reason to believe that the relative simplicity of the SRS would effect the meeting arrangements for developing quality sensitive scenarios.
- Whether the scenario development process is equivalent to that followed in industry. The participants of our study followed a scenario development process that is quite similar to the one used for most of the scenario-based SA evaluation methods, e.g., ALMA (Lassing

et al., 2002) and ATAM (Kazman et al., 1999). Furthermore, the two-staged scenario development process has been evaluated as the most effective and efficient one in an experiment on eliciting scenarios (Bengtsson, 2002).

These issues cannot be resolved in laboratory experiments. This confirms the need to undertake industrial trials as well as laboratory experiments. Thus, if our subsequent experiments confirm the potential value of distributed meetings, out next validation activity should be based on industrial case studies and/or field experiments.

Appendix A. Questionnaire to gather self-reported data

Group Name:

A.1. Personal information

1. Working experience .... (Year/Month).
2. Scenario development experience .... (Year/Month).
3. Training in IT related discipline .... (Year/Month).

A.2. Study related questions

(1) Overall, did you feel you performed well in developing scenarios for non-functional requirements in?

(I) a distributed arrangement using the collaborative tool

(II) both arrangements

(III) a face-to-face arrangement

Your choice is -------, please explain the reason of your choice:

(2) Overall, did you feel your group performed well in developing scenarios for non-functional requirements in:

(I) a distributed arrangement using collaborative tool

(II) both arrangements

(III) a face-to-face arrangement

Your choice is -------, please explain your choice:

(3) Did you feel that using the collaborative tool had any positive or negative affect on your group discussion? e.g., you may have been able to discuss issues more quickly (a positive effect) or you may have found it more difficult to discuss issues (a negative effect).

Positive effect	No effect	Negative effect

(4) Compared with face-to-face group meeting, do you feel that a collaborative tool based group meeting is

(I) more efficient?

(II) equally efficient?

(III) less efficient?

Note: By efficiency, we mean the *number* of individual scenarios discussed and integrated into group scenarios and the *number* of new ideas and scenarios developed during team meetings.

Your choice is --------, Please explain the reason of your choice:

(5) Describe in detail (5–8 sentences) the *effect* that the collaborative tool had on your group meeting compared to a F2F meeting.

(6) Please use this space to detail any problems you had when using collaborative tool for the group meeting compared to face-to-face meeting, or any other comments you may have.

(7) Overall, what type of meeting arrangement you would like for generating scenarios, face-to-face or using collaborative tool? please give three reasons for your answer.

References

Agarwal, R., De, P., Sinha, A.P., 1999. Comprehending object and process models: an empirical study. IEEE Transactions of Software Engineering 25 (4), 541–556.

Ali-Babar, M., Gorton, I., 2004. Comparison of Scenario-Based Software Architecture Evaluation Methods. In: Proceedings of the 1st Asia-Pacific Workshop on Software Architecture and Component Technologies, Busan, South Korea.

Ali-Babar, M., Zhu, L., Jeffery, R., 2004. A Framework for Classifying and Comparing Software Architecture Evaluation Methods. In: Proceedings of the Australian Software Eng, Conference (ASWEC), Melbourne, Australia.

Basili, V.R., Selby, R.W., Hutchens, D.H., 1986. Experimentation in Software Engineering. IEEE Transactions on Software Engineering 12 (7), 733–743.

Bass, L., Clements, P., Kazman, R., 2003. Software Architecture in Practice. Addison-Wesley.

Bengtsson, P., 2002. Architecture-Level Modifiability Analysis. Blekinge Institute of Technology.

Bengtsson, P., Bosch, J., 2000. An experiment on creating scenario profiles for software change. Annals of Software Engineering 9, 59–78.

Biuk-Aghai, R.P., Hawryszkiewyez, I.T., 1999. Analysis of Virtual Workspaces. In: Proceedings of the Database Applications in Non-Traditional Environments, Japan.

Boeham, B., Grunbacher, P., Briggs, R.O., 2001. Developing groupware for requirements negotiation: lessons learned. IEEE Software 18 (3), 46–55.

Carmel, E., Agarwal, R., 2001. Tactical approaches for alleviating distance in global software development. Software, IEEE 18 (2), 22–29.

Damian, D.E., Eberlein, A., Shaw, M.L.G., Gaines, B.R., 2000. Using different communication media in requirements negotiation. IEEE Software 17 (3).

Dennis, A.R., Garfield, M.J., 2003. The adoption and use of GSS in projects teams: toward more participative processes and outcomes. MIS, Quarterly 27 (2), 289–323.

Ellis, C.A., Gibbs, S.J., Reln, G.L., 1991. Groupware: some issues and experiences. Communication of the ACM 34 (1).

Fjermestad, J., 2004. An analysis of communication mode in group support systems research. Decision Support Systems 37 (2), 239–263.

Genuchten, M.V., Cornelissen, W., Dijk, C.V., 1997–1998. Supporting Inspection with an electronic meeting system. Journal of Management Information Systems 14 (3), 165–178.

Genuchten, M.V., Van Dijk, C., Scholten, H., Vogel, D., 2001. Using group support systems for software inspections. IEEE Software 18 (3).

Gorton, I., Hawryszkiewycz, I., Fung, L., 1996. Enabling software shift work with groupware: a case study. In: Proceedings of the 29th Hawaii International Conference on System Sciences, pp. 72–81.

Gorton, I., Motwani, S., 1996. Issues in co-operative software engineering using globally distributed teams. Journal of Information and Software Technology 38 (10), 647–655.

Griffith, T., Sawyer, J.E., Neale, M.A., 2003. Virtualness and knowledge in teams: managing the love triangle of organisations, individuals, and information technology. MIS, Quarterly 27 (2), 265–287.

Grohowski, R., McGoff, C., Vogel, D., Martz, B., Nuamaker, J., 1990. Implementing electronic meeting systems at IBM: lessons learned and success factors. MIS, Quarterly 14 (4), 369–383.

Grudin, J., 1994. Groupware and social dynamics: eight challenges for developers. Communication of the ACM 37 (1), 92–105.

Grudin, J., Palen, L., 1995. Why groupware succeeds: discretion or mandate. In: Proceedings of the ECSCW, Dordrecht, the Netherlands, pp. 263–278.

Gruenbacher, P., 2000. Integrating groupware and CASE capabilities for improving stakeholder involvement in requirements engineering. In: Proceedings of the 26th Euromicro Conference, vol. 2, pp. 232–239.

Halling, M., Grunbacher, P., Biffl, S., 2001. Tailoring a COTS group support system for software requirements inspection. In: Proceedings of the 16th International Conference on Automated Software Engineering, pp. 201–208.

Hawryszkiewycz, I.T. LiveNet, <http://livenet.it.uts.edu.au/index.htm> (Last accessed on 8th June, 2004).

Hawryszkiewycz, I.T., Gorton, I., 1996. Distributing the Software Process. In: Proceedings of the Australia Software Engineering Conference.

Herbsleb, J.D., Moitra, D., 2001. Global software development. Software, IEEE 18 (2), 16–20.

Kazman, R., Bass, L., Abowd, G., Webb, M., 1994. SAAM: A method for analyzing the properties of software architectures. In: Proceedings of the 16th ICSE, pp. 81–90.

Kazman, R., Barbacci, M., klein, M., Carriere, S.J., 1999. Experience with performing architecture tradeoff analysis. In: Proceedings of the 21th International Conference on Software Engineering, New York, USA, pp. 54–63.

Kitchenham, B.A., Pfleeger, S.L., Pickard, L.M., Jones, P.W., Hoaglin, D.C., El Emam, K., Rosenberg, J., 2002. Preliminary guidelines for empirical research in software engineering. Software Engineering, IEEE Transactions on 28 (8).

Kitchenham, B., Fay, J., Linkman, S., 2004. The case against cross-over design in software engineering. In: Proceedings of the 11th International Workshop Technology and Engineering Practice.

Lanubile, F., Mallardo, T., 2002. Tool support for distributed inspection. In: Proceedings of the 26th Computer Software and Applications Conference, pp. 1071–1076.

Lassing, N., Bengtsson, P., Bosch, J., Vliet, H.V., 2002. Experience with ALMA: architecture-level modifiability analysis. Journal of Systems and Software 61 (1), 47–57.

Layzell, P., Brereton, O.P., French, A., 2000. Supporting collaboration in distributed software engineering teams. In: Proceedings of the 7th Asia Pacific Software Engineering Conference, pp. 38–45.

Liou, Y.I., Chen, M., 1993. Using group support systems and joint application development for requirements specification. Journal of Management Information Systems 10 (3), 25–41.

Maranzano, J.F., Rozsypal, S.A., Zimmerman, G.H., Warnken, G.W., Wirth, P.E., Weiss, D.M., 2005. Architecture reviews: practice and experience. IEEE Software 22 (2), 34–43.

Mashayekhi, V., Feulner, C., Riedl, J., 1994. CAIS: Collaborative Asynchronous Inspection of Software. In: Proceedings of the 2nd ACM SIGSOFT Symposium on Foundation of Software Engineering, USA.

Nunamaker, J.F., Briggs, R.O., Mittleman, D.D., Vogel, D.R., Balthazard, P.A., Winter, 1996–1997. Lessons from a dozen years of group support systems research: a discussion of lab and field findings. Journal of Management Information Systems 13 (3) 163–207.

Nunamaker, J.F., Dennis, A.R., Valacich, J.S., Vogel, D., George, J.F., 1991. Electronic meeting systems to support group work. Communication of the ACM 34 (7).

Olofsson, B., 2004. StudySize Trial, Version 1.0.8, CreoStat HB.

Perry, D.E., Porter, A., Wade, M.W., Votta, L.G., Perpich, J., 2002. Reducing inspection interval in large-scale software development. IEEE Transactions of Software Engineering 28 (7), 695–705.

Piccoli, G., Ahmad, R., Ives, B., 2001. Web-Based virtual learning environments: a research framework and a preliminary assessment of effectiveness in basic IT skills training. MIS, Quarterly 25 (4), 401–426.

Porter, A.A., Johnson, P.M., 1997. Assessing software review meetings: results of a comparative analysis of two experimental studies. IEEE Transactions on Software Engineering 23 (3), 129–145.

Sakthivel, S., 2005. Virtual workgroups in offshore systems development. Information and Software Technology 47 (5), 305–318.

Sauer, C., Jeffery, D.R., Land, L., Yetton, P., 2000. The effectiveness of software development technical reviews: a behaviorally motivated program of research. IEEE Transactions on Software Engineering 26 (1).

Senn, S., 2002. Cross-Over Trials in Clinical Research. John Wiley & Sons Ltd.

Shadish, W.R., Cook, T.D., Campbell, D.T., 2001. Experimental and Quasi-Experimental Design for Generalized Causal Inference. Houghton Mifflin Company.

Tyran, C.K., George, J.F., 2002. Improving software inspections with group process support. Communication of the ACM 45 (9), 87–92.

Valachich, J.S., Dennis, A.R., 1994. Idea generation in computer-based groups: a new ending to an old story. Organizational Behavior and Human Decision Processes 57 (3), 448–467.

van Genuchten, M., van Dijk, C., Scholten, H., Vogel, D., 2001. Using group support systems for software inspections. IEEE Software 18 (3).

Wholin, C., Runeson, P., Host, M., Ohlsson, M.C., Regnell, B., Wesslen, A., 2000. Experimentation in Software Engineering: An Introduction. Kluwer Academic Publications.

Zwiki System, <http://www.zwiki.org>, (Last accessed on 8th June 2004).

Muhammad Ali Babar is a Ph.D. candidate in the School of Computer Science and Engineering at UNSW and a research scientist with empirical software engineering program of National ICT Australia (NICTA). Previously, he worked in software developer and consultant roles for several years. He received an M.Sc. in computing sciences from the University of Technology, Sydney. His research activities include software architecting, evidence-based software engineering, global software development, requirements engineering, and software process improvement.

Barbara Kitchenham is Professor of Quantitative Software Engineering at Keele University and a senior principal researcher at the National ICT Australia. Her main research interest is software metrics and its application to project management, quality control, risk management and evaluation of software technologies. She is particularly interested in the limitations of technology and the practical problems associated with applying measurement technologies and experimental methods to software engineering. She is a Chartered Mathematician and Fellow of the Institute of Mathematics and Its Applications. She is also a Fellow of the Royal Statistical Society. She is a visiting professor at both the University of Bournemouth and the University of Ulster.

Liming Zhu is a Ph.D. candidate in the School of Computer Science and Engineering at University of New South Wales. He is also a member of the Empirical Software Engineering Group at National ICT Australia (NICTA). He obtained his BSc from Dalian University of Technology in China. After moving to Australia, he obtained his M.Sc. in computer science from University of New South Wales. His principle research interests include software architecture evaluation and empirical software engineering.

Ian Gorton is a researcher at National ICT Australia. Until Match 2004 he was Chief Architect in Information Sciences and Engineering at the US Department of Energy's Pacific Northwest National Laboratory. Previously he has worked at Microsoft and IBM, as well as in other research labs. His interests include software architectures, particularly those for large-scale, high-performance information systems that use commercial off-the-shelf (COTS) middleware technologies. He received a Ph.D. in Computer Science from Sheffield Hallam University.

Dr. Ross Jeffery is Professor of Software Engineering in the School of Computer Science and Engineering at UNSW and Program Leader for Empirical Software Engineering in National ICT Australia (NICTA). His current research interests are in Software engineering process and product modeling and improvement, electronic process guides and software knowledge management, software quality, software metrics, software technical and management reviews, and software resource modeling and estimation. His research has involved over fifty government and industry organizations over a period of 20 years and has been funded from industry, government and universities. He has co-authored four books and over 120 research papers. He has served on the editorial board of the IEEE Transactions on Software Engineering, and the Wiley International Series in Information Systems and he is an Associate Editor of the Journal of Empirical Software Engineering. He is a founding member of the International Software Engineering Research Network (ISERN). He was elected Fellow of the Australian Computer Society for his contribution to software engineering research.