The Application of Knowledge-Sharing Workspace Paradigm for Software Architecture Processes

Muhammad Ali Babar
Lero, the Irish Software Engineering Research Centre, University of Limerick, Ireland
malibaba@lero.ie

ABSTRACT
Management of architectural knowledge is vital for improving an organisation's capabilities in software architecture. Recently, there have been many efforts to develop various kinds of tools for managing architectural knowledge. However, most of these efforts overlook the fact that most of the working teams in today's organisations are distributed. This paper proposes the application of electronic workspace paradigm for capturing and sharing knowledge to support the software architecture processes.

Categories and Subject Descriptors
D.2.2 [**Software Engineering**]: Design Tools and Techniques.

General Terms: Design, experimentation, management

Keywords
Architectural knowledge, workspaces, architecture evaluation.

1. INTRODUCTION
The Software Architecture (SA) process consists of several activities (such as design, documentation, and evaluation), which involve complex knowledge intensive tasks [1, 2]. The knowledge that is required to make suitable architectural choices and to rigorously assess those design decisions is broad, complex, and evolving. The requisite knowledge can be technical (such as patterns, tactics, and quality attribute analysis models) or contextual, also called Design Rationale (DR), (such as design options considered, tradeoffs made, assumptions, and design reasoning) [3]. The former type of knowledge is required to identify, assess, and select suitable design options for design decisions. The latter is required to provide the answers about a particular design option or the process followed to select that design option [4, 5]. If not documented, knowledge concerning the domain analysis, patterns used, design options evaluated, and decisions made is lost, and hence is unavailable to support subsequent decisions in development lifecycle.

Though, researchers emphasized the importance of Design Rationale (DR) in software design and the challenges involved in documenting it [6, 7] more than 20 years ago, it was Bosch's paper [8] that has drawn significant attention to architectural knowledge management research. Apart from researchers, practitioners also reported that design decisions and their rationale are not rigorously documented [9]. Lack of suitable methodological and tool support has been described as one of the main reason for this situation [10]. In response to the increasing realization of the importance of providing suitable tooling support for capturing and sharing Architectural Knowledge (AK), several researchers have developed various tools [11-14] for Architectural Knowledge Management (AKM) and others have identified requirements with the intention of providing such tools [15].

However, most of these efforts overlook the fact that increasingly working teams in today's organisations are distributed in the context of global software development [16]. Many organizations are implementing a virtual team strategy as the primary focus of their global software development policy. This situation necessitates that team members are usually loosely connected to the various teams to perform their respective tasks. It is quite hard to anticipate the processes, artifacts, roles, and tools such teams will need to perform different tasks in a process. Like any process of software development lifecycle, the software architecture processes also faces new challenges posed by the increasing trend of virtual teams in the context of distributed software development. That is why we question the ongoing trend of developing tools based on the traditional workflow paradigm. We propose the application of electronic workspace paradigm to capture and share AK. We present a structure of an electronic workspace for providing an effective mechanism of AKM in global software development. The process of modeling different activities of one of the software architecture processes is also illustrated. Then, it is demonstrated how to map those activities onto workspaces provided by a collaborative tool for capturing and sharing AK to support the architecture evaluation process.

2. KNOWLEDGE-SHARING WORKSPACE
Members of a traditional work group perform their individual and collective tasks using a physical space where all the required objects, tools, and guidelines are made available. However, teams of geographically dispersed members use a virtual place as a substitute to the physical space. These virtual spaces are called workspaces, which are conceived as a logical counterpart of physical spaces and are based on physical metaphors [17, 18]. Like physical spaces, these workspaces are expected to make all the required objects, tools, people, and guidelines available along with all the necessary communication channels and coordination mechanisms. A workspace is expected to create opportunities for the users to turn them into a place of collaboration [17] as in a virtual world it is not the spatial features of a space that matters the most, rather what the users of such a space can do within it and that is what turns such a space into a place. That is why it is vital that a virtual space provides its users with an opportunity to turn it into a place for collaboration [18].

Figure 1 shows the structure of a knowledge-sharing workspace along with its major components, which we have designed to

SHARK'08, May 13, 2008, Leipzig, Germany.

support the different activities (such as design, evaluation, and documentation) of the software architecture process in the context of global software development. All the objects of the workspace are related with one another according to a meta-model of collaboration presented in [18]. Actors assuming organisational roles collaborate with other roles within same workspace or in another workspace to perform designated actions on artifacts to achieve desired organisational goals [17]. Artifacts are either inputs for or outputs from different actions. Roles are attached to a particular workspace and are unique within that workspace; more than one workspace may have the same roles attached to them. For example, a manager role may be attached to the workspaces designed to support the plan architecture evaluation, and prepare and manage results activities of the proposed process.

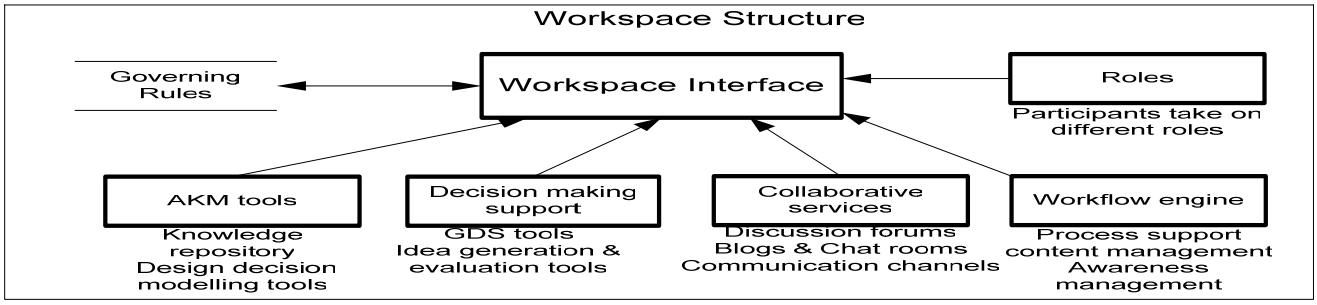

Figure 1: The structure of a knowledge-sharing workspace and its major components

Each workspace also has certain governance rules, which control the relationships among objects of that particular workspace. For example, a rule may provide read only access to software architecture documentation to the evaluation team, while the software architect has full access to that documentation. Each role can perform two types of actions on a workspace, namely individual actions (editing a document, sending a notification) or interaction actions (discussing ideas, prioritising scenarios). A workspace also provides a wide range of notification features to support synchronous and asynchronous communication. Apart from different tools for supporting the 3Cs (i.e., collaboration, coordination, and communication) of software development teams, each workspace should provide several kinds of tools for modeling and describing architectural design decisions and repositories of AK. Such tools can help build and manage knowledge repository of reusable architectural artifacts (such as general scenarios, patterns, general design decisions, tactics and so on) and can also support decision making by assessing different design options. Since there can be different kinds of AK that may need different kinds of tools and repositories based on organizational requirements and processes, we do not support the idea of prescribing any particular AK management tool, which is also consistent with the emerging community consensus [19]. Rather, we expect the users of the workspaces to select an appropriate AKM tool based on their own needs.

3. AN ILLUSTRATED EXAMPLE

We have used rich pictures to model a generic process of evaluating software architecture in the software architecture process. This section illustrates the use of rich picture modeling and workspace concepts for supporting architecture evaluation in a distributed arrangement by designing some of the activities, inputs, outputs, and roles involved in the generic process. For designing a process structure for the architecture evaluation process to be mapped on a workspace-based system, this research uses a diagrammatic notation, which is a modified version of Rich Picture modeling language [17]. According to the notation, activities, artifacts, and roles are represented by the clouds, rectangles, and human figures respectively. The arrows linking artifacts and activities represent inputs to and output from the activities. While the lines linking roles and activities represent their relationships.

A rich picture representation of a software architecture evaluation process is shown in Figure 2. It shows main roles, artifacts, and their interaction with different analysis activities. The artifacts are either inputs (e.g. business goals and architectural requirements) or outputs (e.g. prioritised scenarios, risks and non-risks) of different activities. Each of the artifacts and participants of a workspace are containers of AK used/generated during each of the activity shown in Figure 2. On these workspaces, the main source of explicit AK is the artifacts; while workspace participants are the main source of tacit AK. Moreover, the artifacts on a workspace represent the codification strategy of AKM and the participants are linked to each other and with artifacts to support the personalization strategy of AKM. Hence, we assert that the knowledge-sharing workspaces can support a hybrid strategy to managing knowledge [20].

The participants of each workspace take on their respective roles (e.g. architect or evaluation team) to perform individual as well as group tasks involving collaborative activities. For example, the rich picture shows that a participant of the software architecture evaluation process may take on the role of a manager to perform the architecture evaluation planning and preparation activity. This activity needs artifacts like business goals and project plan as input and produces or modifies artifacts like software requirements specifications (SRS), software architecture documentation, and the evaluation plan as the outputs, which are used as the inputs for other activities, e.g., present and explain architecture, gather quality-sensitive scenarios, and analyzing architectural approaches used in a proposed software architecture. The architecture evaluation process modeled using rich picture can be implemented on a system that supports the workspace paradigm ((e.g., systems reported by Hawryszkiewycz [21] and Stevens et al. [22]). This mapping follows simple rules, namely:

1. For each activity of rich picture create a top-level workspace.
2. Roles associated to a particular activity become roles on the workspace for that activity.
3. Artifacts consumed or produced by an activity are attached to it on the workspace.

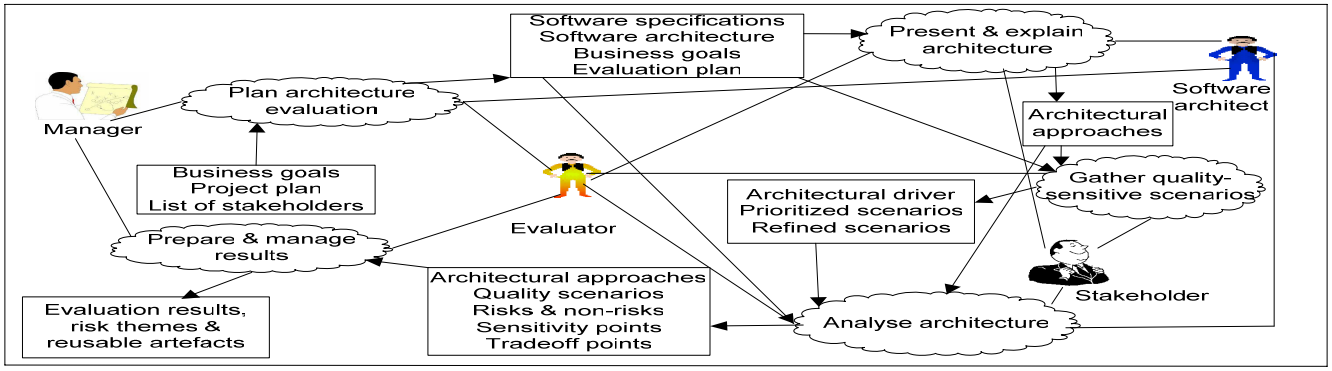

Figure 2: Rich picture showing a model of knowledge-sharing workspaces for architecture evaluation process

We now describe the implementation of the modeled process for evaluating software architecture shown in Figure 2 on workspaces provided by LiveNet [21]. We selected LiveNet as its workspaces provide majority of the features identified by the structure of knowledge-sharing workspace in section 2 and its free availability for research and education. It provides a generic workflow engine and different features to support geographically distributed team. LiveNet supports the workspace paradigm and emergent work process by providing different mechanisms for capturing and sharing explicit and tacit knowledge about processes, activities, and artifacts. LiveNet enables users to create and manage their own workspaces along with the required components. LiveNet's workspaces also provide a number of features for awareness and sharing views to keep the participants updated on different aspects of the activities being performing using the workspace.

Figure 3 shows the workspaces, roles, participants, and artifacts created in LiveNet to illustrate the mapping between the model of the software architecture evaluation process presented in Figure 3 and the workspaces that can be used to support different activities of the process involving various kinds of stakeholders who may be located at different physical locations. Figure 3 shows that each high-level activity of the architecture evaluation process (i.e., Plan Architecture Evaluation, Present and explain architectures, Gather Quality sensitive scenarios, Analyze Architecture, and Reporting) shown in Figure 2 has been implemented as a separate workspace.

Each of these workspaces may have sub-workspaces depending upon the requirements of a particular activity. For example, if an evaluation manager decides to conduct the "Gather Quality Sensitive Scenarios" activity using the process of Quality Attribute Workshop (QAW) [23], then there can be up to eight sub-workspaces; one for each of the eight activities of the QAW. However, the evaluation manager may decide to use only one workspace for all activities if the number of stakeholders and system being evaluated are relatively small.

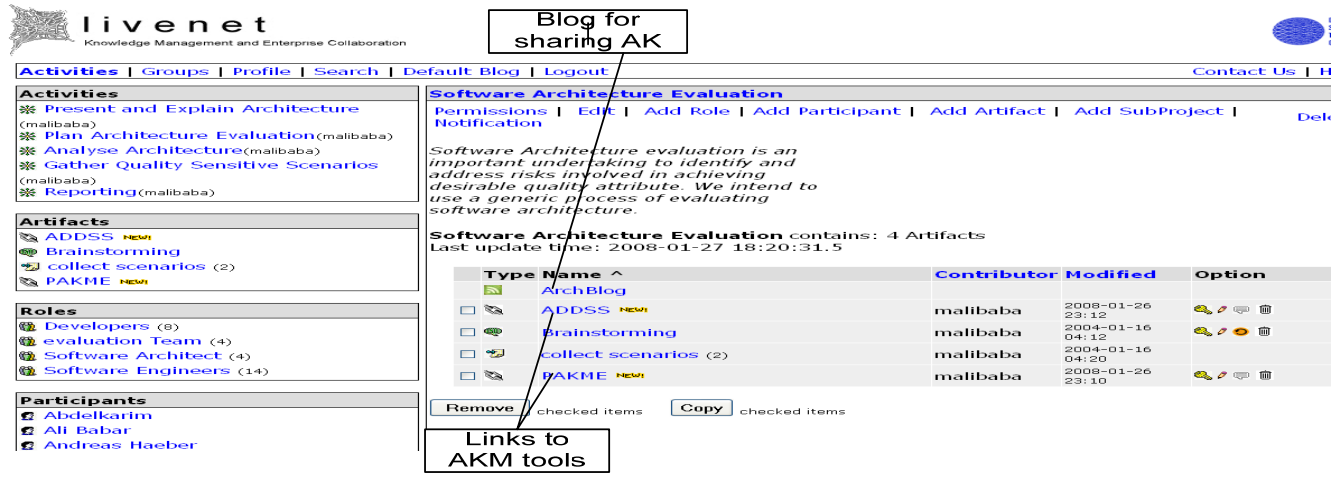

Figure 3: A view of different elements of a knowledge-sharing workspace for supporting software architecture evaluation process

Left-hand side of Figure 3 also shows the artifacts, roles, and participants on the top-level workspace for evaluating software architecture process. The artifacts are also mentioned in the middle of the workspace. So far this workspace contains five artifact; one blog (ArchBlog) for sharing views and opinions on different aspects of the process and artifacts involved in architecture evaluation, one chat room (Brainstorming) for synchronous communication, one forum (collect scenarios) for asynchronous communication, and two links to external AKM tools (ADDSS [11] and PAKME [13]), which can be used depending upon a project's requirements. We support the idea of not prescribing a particular AKM tool; rather letting a user to choose a tool. That is why currently LiveNet only support external AKM. However, we are assessing the pros and cons of integrating an AKM tool like PAKME into LiveNet workspaces. However, technically such integration does not appear too hard.

4. SUMMARY AND OUTLOOK

Several researchers have been developing tooling support for capturing and sharing AK. However, most of these tools have been designed based on the traditional workflow paradigm, which is not scaleable to the needs of distributed and virtual teams in the context of global software development. This paper proposes to apply the electronic workspace paradigm for capturing and sharing AK to support the software architecture processes. It asserts that the workspace networks supported by a collaborative system like LiveNet are particularly applicable to capture and encourage sharing AK in physically distributed teams. These workspaces allow participants to share knowledge in manners increasingly being used by different communities. Participants of such workspaces can produce or consume AK using codification strategy and/or easily consult known experts and/or peers wherever needed using personalization strategy. Moreover, teams can easily change the workspace objects and participants as the work progresses and teams evolve.

There are several points still open for discussion and experimentation before the proposed solution can be considered mature enough to support the architecture evaluation processes in practice. However, we are confident, based on the well-developed technological underpinning and findings from our research on using LiveNet for architecture evaluation [16], that the knowledge-sharing workspaces can provide an effective and efficient mechanism for capturing and sharing AK.

Acknowledgement: *This work is supported by Science Foundation Ireland under grant number 03/CE2/I303-1.*

5. REFERENCES

[1] P.N. Robillard, The role of knowledge in software development, *Communication of the ACM*, 1999. **42**(1): pp. 87-92.

[2] L.G. Terveen, P.G. Selfridge, and M.D. Long, Living Design Memory: Framework, Implementation, Lessons Learned, *Human-Computer Interaction*, 1995. **10**(1): pp. 1-37.

[3] M. Ali-Babar, I. Gorton, and B. Kitchenham, A Framework for Supporting Architecture Knowledge and Rationale Management, in Rationale Management in Software Engineering, A.H. Dutoit, et al., Editors. 2006, Springer. pp. 237-254.

[4] A.H. Dutoit and B. Paech, Rationale Management in Software Engineering, in Handbook of Software Engineering and Knowledge Engineering, S. Change, Editor. 2001, World Scientific Publishing, Singapore.

[5] T. Gruber and D. Russell, Design Knowledge and Design Rationale: A Framework for Representation, Capture, and Use, *Tech Report KSL 90-45*, Knowledge Laboratory, Stanford University, Standford, United States, 1991.

[6] D. Parnas and P. Clements, A Rationale Design Process: How and Why to Fake It, *IEEE Transactions of Software Engineering*, 1986. **12**(2): pp. 251-257.

[7] C. Potts and G. Burns, Recording the reasons for design decisions, *Proceedings of the 10th International Conference on Software Engineering*, 1988.

[8] J. Bosch, Software Architecture: The Next Step, *European Workshop on Software Architecture*, 2004.

[9] J. Tyree and A. Akerman, Architecture Decisions: Demystifying Architecture, *IEEE Software*, 2005. **22**(2): pp. 19-27.

[10] A. Tang, M. Ali-Babar, I. Gorton, and J. Han, A Survey of Architecture Design Rationale, *Journal of Systems and Software*, 2006. **79**(12): pp. 1792-1804.

[11] R. Capilla, F. Nava, S. Perez, and J.D. Duenas, A Web-based Tool for Managing Architectural Design Decisions, *Proceedings of the 1st Workshop on Sharing and Reusing Architectural Knowledge*, 2006.

[12] A. Tang, Y. Yin, and J. Han, A Rationale-based Architecture Model for Design Traceability and Reasoning, *Journal of Systems and Software*, 2007. **80**(6): pp. 918-934.

[13] M. Ali-Babar and I. Gorton, A Tool for Managing Software Architecture Knowledge, *Proceedings of the 2nd Workshop on SHAring and Reusing architectural knowledge - Architecture, rationale, and Design Intent (SHARK/ADI 2007), Collocated with ICSE 2007.*, 2007.

[14] A. Jansen, J.v.d. Ven, P. Avgeriou, and D.K. Hammer, Tool Support for Architectural Decisions, *Proceedings of the 6th Working IEEE/IFIP Conference on Software Architecture*, 2007.

[15] R. Farenhorst, P. Lago, and H.v. Vliet, Effective Tool Support for Architectural Knowledge Sharing, *Proceedings of the First European Conference on Software Architecture*, 2007.

[16] M. Ali-Babar, B. Kitchenham, and R. jeffery, Comparing distributed and face-to-face meetings for software architecture evaluation: A controlled experiment, *Empirical Software Engineering*, 2008. **13**(1): pp. 39-62.

[17] I. Hawryszkiewycz, Designing the Networked Enterprise. 1997: Artech House, Boston, USA.

[18] R.P. Biuk-Aghai and I.T. Hawryszkiewyez, Analysis of Virtual Workspaces, *Proceedings of the Database Applications in Non-Traditional Environments*, 1999.

[19] P. Avgeriou, et al., Architectural Knowledge and Rationale - Issues, Trends, Challenges, *ACM SIGSOFT Software Engineering Notes*, 2007. **32**(4): pp. 41-46.

[20] K. Desouza and Y. Awazu, Managing Knowledge in Global Software Development Efforts: Issues and Practices, *IEEE Software*, 2006. **23**(5): pp. 30-37.

[21] I.T. Hawryszkiewycz. LiveNet. Last accessed on January 20, 2008, Available from: http://livenet4.it.uts.edu.au/.

[22] W.P. Stevens, G.J. Myers, and L.L. Constantine, Structured Design, *IBM Systems Journal*, 1974. **13**(2): pp. 115-139.

[23] M.R. Barbacci, et al., Quality Attribute Workshops (QAWs), *Tech Report CMU/SEI-2003-TR-016*, SEI, Carnegie Mellon University, USA., 2003.