A State of the Art: Software Configuration Management Tools for Global Software Development

S.S.M.Fauzi*,1, A.J.Suali1, W.A.W.M.Sobri1
Faculty of Computer and Mathematical Sciences
Universiti Teknologi MARA, Perlis Campus, 02600 Arau, Perlis, Malaysia
E-mail: shukorsanim@uitm.edu.my

Abstract. Global Software Development (GSD) has emerged widely in today’s software development environment. The emergence of GSD has create difficulty to several technical areas such as Requirement Engineering, Validation and Verification; not to mention Software Configuration Management (SCM) practices. To date, many SCM tools has been proposed by the researchers to reduce the difficulties faced by GSD project developers when dealing with SCM practices. Therefore, there is a need to update on the current trend of SCM tools in GSD context in order to assist practitioners in selecting the best tools for their project. To perform the study, Systematic Mapping Studies (SMS) has been used to identify and provide an overview of the SCM tools which has been developed specifically for GSD environment. This paper contributes the following: 1) This paper synthesized the SCM tools in GSD and 2) Current trend of SCM tools in today’s software development environment also is identified.

1. Introduction

Global Software Development (GSD) has become an enabler in today’s world IT industry. Software and Information Industry Association (SIIA) in their survey has identified seven important factors that lead the decision makers to move towards GSD environment. Cost Advantage remains one of the most essential aspects, whereas increase speed to market, increase productivity, growth strategy, improve operational effectiveness, competitive pressure and improve product quality also has drive IT companies to shift to this trend. Numerous Information Technology (IT) mainstream companies lead by Lucent [1], IBM, Motorola [2], and Philips [3] have make a move to this new era of software development environment.

Several researchers have identified a bundle of issues and challenges when dealing with the GSD environment. Prior study has identified three main threats to GSD environment 1) Communication – the exchange of complete and unambiguous information that is the sender and receiver can reach a common understanding 2) Coordination – the act of integrating each task with each organizational unit, so the unit contributes to the overall objective 3) control issues – the process of adhering to goals, policies, standards or quality levels [4].

* Corresponding author: shukorsanim@uitm.edu.my.
Software Configuration Management (SCM) also suffered with the emergence of GSD environment [5]. That has lead various researchers to introduce a range of strategies, approaches, techniques and tools that can be used to improve SCM in GSD environment [6].

By looking at the GSD development trend which is incredibly complex, we believe that there is an evolutionary of SCM tools which has been developed specifically for GSD environment. Therefore, the objective of the study is to identify and synthesize the SCM tools which have been developed for GSD environment. Next, we also would like to identify the current trend of SCM tools development.

Next section of this paper will explain about the methodology used to conduct this research. It then follows by the discussion of the results achieved in this study. Finally, we will conclude our paper.

2. Methodology

We have carried out a systematic mapping study; an approach that provides a structure of research results that has been published by categorizing them in a visual summary or the map [7].

2.1. Developing Research Questions

While the objective of this paper is to identify the SCM tools for GSD environment, the research questions which drive this study are:

**RQ1:** What is the evolution of SCM tools in GSD environment?

**RQ2:** What is the current trend of SCM tools?

2.2. Search Strings and Source Selection

We have performed two separate stage of extensive search to answer our research questions. In stage 1, we have identified a keyword which has been alienated into two categories as shown below. Categories A comprise a keyword which is related to “Software Configuration Management”, while category B consists of the keyword which has interrelated to “Global Software Development”. Both categories were combined using the Boolean “AND” expression: *(A1 OR A2 OR A3 OR A4 OR A5 OR A6 OR A7 OR A8 OR A9) AND (B1 OR B2 OR B3 OR B4 OR B5 OR B6 OR B7 OR B8 OR B9 OR B10 OR B11)*.

| A1 - Configuration management tool | B1 - Global software development |
|----------------------------------|----------------------------------|
| A2 - Software configuration management tool | B2 - Global software engineering |
| A3 - Software change management tool | B3 - Globally distributed development |
| A4 - Software change request tool | B4 - Geographically distributed development |
| A5 - Change management tool | B5 - Distributed software development |
| A6 - Change request tool | B6 - Collaborative software development |
| A7 - Change control tool | B7 - Collaborative software engineering |
| A8 - Version control tool | B8 - Offshore software development |
| A9 - Versioning tool | B9 - Off shoring |
| A9 - Modification request tool | B10 - Dispersed locations |
| B11 - Dispersed teams |
We then performed a search to several electronic databases; for instance:

1) IEEEXplore (http://ieeexplore.ieee.org)
2) ACM Digital Library (http://www.portal.acm.org/dl.cfm)
3) Compendex EI (http://www.engineeringvillage2.org)
4) INSPEC (http://www.engineeringvillage2.org)
5) Google Scholar (http://scholar.google.com)
6) Elsevier ScienceDirect (http://www.sciencedirect.com)

2.3. Screening of Papers

As a first step, we have decided to analyze the title, abstract and keywords for each paper. In some condition, we also have to read through the entire of paper to determine its relevancy to the research question. However, both situations are lie on the inclusion and exclusion criteria that we have governed before the screening process. It will helps us to ensure only paper which has relevancy to the research questions is being considered. Thus, the inclusion criteria that we have applied are:

11. Papers published should directly relate to SCM tools and the studies should have focus on GSD environment context.

As such, the exclusion criteria that we applied for this research consist of:

E1. The paper which lies on difference aspect of SCM. For instance cultural, time zone, language issues; which is extremely differ from our focus;
E2. We excluded any posters, panels, abstract, power point presentation or even the summaries of articles.
E3. Papers which consist of opinions, claims, personal evaluation and reported recommendations will not be considered since there is a lack of empirical work that focus unswervingly to SCM in GSD environment context.
E4. Some authors may publish more than one paper related to their tools. Therefore, we have decided to exclude any redundancy paper which mainly discuss about the same subject.

3. Results and Discussion

3.1. Overview of studies

Details of the papers found have been entered to online bibliographic information; RefWorks. Our early record indicates there are 125 papers related to our research. However, only 7 papers are selected and it has resulted 8 SCM tools. Table 1 depicts most of the SCM tools have been developed in 2007 (33%). We can consider that, the attentiveness to develop SCM tools for GSD project developers just started in 4 years back.

Most of the SCM tools has been published in a leading conference; for instance International Workshop on Cooperative and Human Aspects of Software Engineering - CHASE and International Conference on Software Engineering – ICSE (25%) , whereas International Conference on Global Software Engineering – ICGSE, SIGCHI Conference on Human Factors in Computing Systems – CHI, ACM SIGSOFT International Symposium on Foundations of Software Engineering – ESEC/FSE and European Conference on Computer Supported Cooperative Work – ECSCW (13%).
Table 1. Number of papers published

| Year | Percentage | Frequency |
|------|------------|-----------|
| 1999 | 13         | 1         |
| 2000 | 0          | 0         |
| 2001 | 0          | 0         |
| 2002 | 0          | 0         |
| 2003 | 13         | 1         |
| 2004 | 0          | 0         |
| 2005 | 0          | 0         |
| 2006 | 13         | 1         |
| 2007 | 38         | 3         |
| 2008 | 13         | 1         |
| 2009 | 0          | 0         |
| 2010 | 13         | 1         |

Based on our systematic study, we found that most of the researchers employed several methods to develop their tools; for instance – survey, observation, interview and also video recording. We also notice that, researchers have tested their tools in two difference environment 1) academic environment - engage a certain number of students which then been alienated to several groups or pairs 2) iterative development experiment; actual environment experiment - to gather the actual practices of GSD project developers, lab experiment – to develop and simulate the tools.

3.2. SCM tools in GSD environment (RQ1)

Tools are becoming one of the essential parts in GSD environment. It can assist GSD project developers to collaborate, facilitate the works and even to get communicate among other project developers [8]. As such, there are numerous tools which have been developed. To name a few; collaboration tools [8][9][10] and communication tools [11].

In SCM context, adequate tools to support GSD project developers performing their task are vital. This is for reason that, GSD environment has limit the group awareness [6]. This therefore can lead to a several difficulties such as problem to make a change [3][5][12][13] and even to control the versioning of the artifacts [5].

Our study reveal numerous of SCM tools which have been developed to cater the need of today’s development environment. Table 2 shows the list of tools which has been developed specifically to accommodate the GSD environment needs. Subsequently, we extend our discussion on the tools listed in Table 2 in the next paragraph.

Table 2. List of SCM tools in GSD environment context

| Title of Paper | Author | Tools | Year | Source | Focus of the tools |
|----------------|--------|-------|------|--------|-------------------|
| Supporting distributed software development with fine-grained artefact management | [14] | ADAMS | 2006 | ICGSE | Focus on high level documentation versioning |
| Palantir: Raising awareness among configuration management workspaces | [15] | Palantir | 2003 | ICSE | To raise the awareness among the project developers in dispersed location |
| Tool                  | Year | Conference | Description                                                                                                                                 |
|----------------------|------|------------|---------------------------------------------------------------------------------------------------------------------------------------------|
| FASTDash             | 2007 | CHI        | Determine which team members have source files checked out, which files being viewed and what are the task currently being done by other project developers. |
| CHIME                | 1999 | ESEC/FSE  | Intend to collaborate large software development project which are geographically dispersed via the use of virtual environment technology.            |
| Syde                 | 2010 | ICSE      | To establish group awareness by sharing change and conflict information across project developer's workspace.                                   |
| Continuous coordination within the context of cooperative and human aspects of software engineering | 2007 | CHASE     | Visualizes the developers and artifacts in a project using 3D metaphor.                                                                     |
| Ariadne              | 2007 | CHASE     | Highlight on the socio-technical relationships between source code and the developers implementing the artifacts.                             |
| Semi-synchronous conflict detection and resolution in asynchronous software development | 2007 | ECSCW     | Focus on semi-synchronous distributed model that allows project developers to detect early conflict.                                        |

**Advanced Artifact Management System (ADAMS)** - ADAMS is developed to accommodate software artifacts; not only source code, but all the documentations in software development. It can be associated with a template which has been defined in each project. Next, ADAMS also has a facility to create and store traceability links between each artifacts. Similar like other version control system, this tool also enable group of people working on the same artifacts. The developer will be notified once there is any changes happened in the artifacts.
Columbia Hypermedia Immersion Environment (CHIME) - CHIME is dedicated to support large software development project which are geographically dispersed. It comprises three main components; Groupspace which contain all the associated software artifacts and tools for the project. Groupviews is used to describe multiuser. As is ADAMS, Groupviews also will keep other team members informed about the work done by others. While Software Immersion is a third component in CHIME and is used to create a sub task from Groupviews. This will allow other team members to learn any difficulties faced.

Workspace Activity Viewer (WAV) - WAV has been developed using the data gathered from the actual GSD environment. It can visualize the developers and dependencies of the artifacts by using the stack of cylinders. Each cylinders will represent each modified artifacts and which developers has made a changes to the artifacts.

Ariadne - Ariadne is dedicated to deal with socio-technical relationship between source code and developer who own and implement the code. The tool can help the team members to get an idea on what has been done by other team members at other location, as well.

FastDash - FastDash is designed specifically for project of 3 to 8 team members. The specific purpose of this tool is to create a real time awareness and access to key information such as which code is changing, who is changing the code and how the code is being used.

Palantir - Palantir is another tool that provides insight to other team members’ workspaces. This tool concentrates more on the collection, distribution, organization and presentation of the relevant information. It means that Palantir can react as a platform to inform the developers about the changes made by other developers in a certain artifacts, able to measure the severity of the changes and organize the report in a graphical way.

Syde – Syde has the ability to detect merge conflicts by showing the differences between two versions of the artifacts and provide information about the conflict and changes to the project developers in a real time.

CollabVS – CollabVS used a semi-synchronous model which allows a group of project developers to create a code asynchronously. Next, this tool also has the ability to detect early conflict, and resolve the conflict before project developers completed the task.

3.3. Trends in SCM Tools (RQ2)

Prior study has discussed in detail about the evolution of SCM tools from the early generation starting from 1970s to 2000 [20]. First, in 1970s until 1980s, the SCM tools have been used by a single person to manage a mainframe. Thus it requires applying versioning and building technique. It then continues (1980s-1990s), where at that time the software development involve only local team which develops large scale software and it runs using UNIX platform. Hence, workspace management has been introduced in SCM tools. From 1990s-2000s, the development environment has shift to distributed environment. The software is developed by project developers in difference locations (perhaps in difference building). Thus, it requires explicit process support in SCM tools.

Now, the SCM tools have emerged more widely to GSD environment. It involves a large scale software development but in difference locations (perhaps difference continents with a different time zone). The GSD environment has created a complexity to the SCM practices [21]. Thus, based on our systematic study, we found eight SCM tools have introduced additional features in SCM tools which
therefore help to improve the SCM practices in GSD environment. This features lead to another generation of SCM tools. Hence, it also creates a new trend in SCM tools.

As depicts in Table 3, two additional features introduced in today’s SCM tools are; dependency and group awareness. Group awareness is referred as understanding the activities of other project developers [22]. In SCM context, these activities includes the information on who creates the artifacts, who make changes, which artifacts affected with the changes, what has been changed and what other project developers do. Dependency refers to the situation where project developers are not able to proceed their work since waiting other project developers to complete their task. Both conditions thus, resulting a delay and time interval to make a changes in GSD environment.

Thus, several tools have been proposed to resolve the problems. [19] has introduced their tools Palantir and CollabVS. These tools provide an insight for project developers on other project developer’s workspace. To a certain extent, other group of researchers also has made an improvement for the tool by introducing 3D metaphor which allows virtual communication between dispersed team. Such studies include the Columbia Hypermedia Immersion Environment (CHIME); an internet and intranet-based distributed software development which allow users interact with project artifacts by “walking around” the virtual world [17]. [11] introduce Workspace Activity Viewer (WAV) and Ariadne; a similar tools which can visualizes the developers and artifacts in a project using 3D metaphor and gives managers an overview of ongoing activities in a project. The same goes to FastDash by [16] which has a capability to help a developer to quickly determine which team members have source files checked out, which files are being viewed and what methods and classes are currently being changed.

| Tools    | Features incorporated         | Team size |
|----------|------------------------------|-----------|
| ADAMS    | versioning control system    | 8         |
| Palantir | group awareness              | 3         |
| FASTDash | group awareness              | 3 to 8    |
| CHIME    | versioning control system    | Not mentioned |
| Syde     | group awareness              | Not mentioned |
| WAV      | dependency                   | Not mentioned |
| Ariadne  | group awareness              | Not mentioned |
| CollabVS | group awareness              | 8         |

Open source tools that focus on GSD also emerged rapidly. For instance; Git (Git.), Darcs (Darcs.), Bazaar (Bazaar.), Mercurial (Mercurial.). Whereas, other popular open source tools for instance Subversion (Subversion.) and CVS (CVS.) has been used widely in collocated software development environment. All of the mentioned tools are purposely used as version control system. Such tools for trackers are Jira (Jira.) and Bugzilla (Bugzilla.).

4. Conclusion

We consider following to be our initial contributions. First, we have identified the SCM tools which have been proposed by various researchers which then can be applied to GSD context. Second, we have recognized the current trend of SCM tools in today’s software development environment; GSD. Our future work will focus on comparing the feature of the tools identified.
References

[1] Herbsleb, J. D., & Grinter, R. E. (1999). Splitting the organization and integrating the code: Conway's law revisited. ICSE ’99: Proceedings of the 21st International Conference on Software Engineering, Los Angeles, California, United States. 85-95.

[2] Battin, R. D., Crocker, R., Kreidler, J., & Subramanian, K. (2001). Leveraging resources in global software development. IEEE Software, 18(2), 70-77.

[3] Kommeren, R., & Parviainen, P. (2007). Philips experiences in global distributed software development. Empirical Software Engineering, 12(6), 647-660.

[4] Ågerfalk, P. J., Fitzgerald, B., Holmström, H., Lings, B., Lundell, B., & Conchúir, E. Ó. (2005). A framework for considering opportunities and threats in distributed software development. International Workshop on Distributed Software Development, Paris. 47–61.

[5] Prikladnicki, R., Jorge, A., & Evaristo, R. (2006). A reference model for global software development: Findings from a case study. Global Software Engineering, 2006. ICGSE ’06. International Conference on, 18-28.

[6] Sangwan, R., Mullick, N., Bass, M., Paulish, D. J., & Kazmeier, J. (2006). Global software development handbook Auerbach Publications Taylor & Francis Group.

[7] Petersen, K., Feldt, R., Mujtaba, S., & Mattsson, M. (2008). Systematic mapping studies in software engineering. 12th International Conference on Evaluation and Assessment in Software Engineering (EASE), Department of Informatics, University of Bari, Italy.

[8] Lanubile, F., Ebert, C., Prikladnicki, R., & Vizcaino, A. (2010). Collaboration tools for global software engineering. IEEE Software, 27 (2), 52-55.

[9] Aranda, G. N., Vizcaino, A., & Piattini, M. (2009, July). Which Groupware Tool is the most suitable for this group?. In Fourth IEEE International Conference on Global Software Engineering, 2009. pp. 400-405.

[10] Lamersdorf, A., & Munch, J. (2009, July). TAMRI: a tool for supporting task distribution in global software development projects. In Fourth IEEE International Conference on Global Software Engineering, 2009. 322-327.

[11] Al-Ani, B., Trainer, E., Ripley, R., Sarma, A., Van Der Hoek, A., & Redmiles, D. (2008). Continuous coordination within the context of cooperative and human aspects of software engineering. Proceedings of the 2008 International Workshop on Cooperative and Human Aspects of Software Engineering, 1-4.

[12] Herbsleb, J. D., & Mockus, A. (2003). An empirical study of speed and communication in globally distributed software development. IEEE Transactions on Software Engineering, 29: 29(6), 481-494.

[13] Cataldo, M., Bass, M., Herbsleb, J. D., & Bass, L. (2007). On coordination mechanisms in global software development. Second IEEE International Conference on Global Software Engineering, 2007, 71-80.
[14] Bernd Bruegge, Andrea De Lucia, Fausto Fasano, & Genoveffa Tortora. (2006). Supporting distributed software development with fine-grained artefact management. *International Conference on Global Software Engineering*, 2006. 213-222.

[15] Sarma, A., Noroozi, Z., & van der Hoek, A. (2003). Palantir: Raising awareness among configuration management workspaces. *ICSE ’03: Proceedings of the 25th International Conference on Software Engineering*. 444-454.

[16] Biehl, J. T., Czerwinski, M., Smith, G., & Robertson, G. G. (2007). FASTDash: A visual dashboard for fostering awareness in software teams. *CHI ’07: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. 1313-1322.

[17] Dossick, S. E., & Kaiser, G. E. (1999). CHIME: A metadata-based distributed software development environment. *ESEC/FSE-7: Proceedings of the 7th European Software Engineering Conference Held Jointly with the 7th ACM SIGSOFT International Symposium on Foundations of Software Engineering*. 464-475.

[18] Hattori, L., & Lanza, M. (2010, May). Syde: A tool for collaborative software development. In *Proceedings of the 32nd ACM/IEEE International Conference on Software Engineering-Volume 2* (pp. 235-238). ACM.

[19] Dewan, P., & Hegde, R. (2007). Semi-synchronous conflict detection and resolution in asynchronous software development. *ECSWC 2007*, 159-178.

[20] Estublier, J., Leblang, D., Hoek, A. v. d., Conradi, R., Clemm, G., Tichy, W., et al. (2005). Impact of software engineering research on the practice of software configuration management. *ACM Transaction on Software Engineering Methodology, 14*(4), 383-430.

[21] Prikladnicki, R., Audy, J. L. N., & Evaristo, R. (2003). Global software development in practice lessons learned. *Software Process: Improvement and Practice, 8*(4), 267-281.

[22] Dourish, P., & Bellotti, V. (1992). Awareness and coordination in shared workspaces. *Proceedings of the 1992 ACM Conference on Computer-Supported Cooperative Work*, 114.