Extending Global Tool Integration Environment towards Lifecycle Management

Jukka Kääriäinen[1], Juho Eskeli[1], Susanna Teppola[1], Antti Välimäki[2], Pekka Tuuttila[3], and Markus Piippola[4]

[1] VTT, Oulu, Finland
jukka.kaariainen@vtt.fi, juho.eskeli@vtt.fi, susanna.teppola@vtt.fi
[2] Metso Automation Inc, Tampere, Finland
antti.valimaki@metso.com
[3] Nokia Siemens Networks, Oulu, Finland
pekka.tuuttila@nsn.com
[4] Espotel, Oulu, Finland
markus.piippola@espotel.fi

Abstract. Development and verification of complex systems requires close collaboration between different disciplines and specialists operating in a global development environment with various tools and product data storage. Fluent integration of the tools and databases facilitate a productive development environment by enabling the user to easily launch tools and transfer information between the disconnected databases and tools. The concept of Application Lifecycle Management (ALM) was established to indicate the coordination of activities and the management of artefacts during the software product's lifecycle. This paper presents the analysis of an open source global tool integration environment called ToolChain, and proposes improvement ideas for it towards application lifecycle management. The demonstration of ToolChain and the collection of improvement proposals were carried out in the telecommunication industry. The analysis was made using the ALM framework and Global Software Development (GSD) patterns developed in previous studies in the automation industry.

Keywords: Tool integration, Application Lifecycle Management, Lifecycle Management, Eclipse.

1 Introduction

Development and verification of complex systems requires close collaboration between different disciplines and specialists. For example, product managers, architects, developers, project managers and testers produce different kinds of information during product development, such as abstractions from a product, project management data and testing/analysis data. Typically, products are simply too complex to represent only one single type of representation and are thus often described at multiple levels of abstraction [1]. For instance, SW can be presented

R. Meersman, P. Herrero, and T. Dillon (Eds.): OTM 2009 Workshops, LNCS 5872, pp. 238–247, 2009.

using abstractions, such as requirements, design, source code and object code. Therefore, the development of multidisciplinary products is supported with various development and product information management systems, for example, Requirements Management tools, Configuration Management tools, Development tools, Testing tools, Test Data databases, etc. These systems are targeted for certain development lifecycle phases. The lifecycle artefacts from the systems should be interlinked in product information management databases in order to support, for example, reporting of lifecycle artefacts like testing coverage of requirements. Usually preserving the consistency of data would require integrations between existing systems. The need for integration of various product information management systems has been discussed, e.g. in [2] and [3]. Instead of copying the product data from one database to another, it is more reasonable to use item identifiers (IDs) to compose traceability information and then to use this traceability information for retrieving up-to-date information from the source database, for example, for reporting purposes. One of the responses to the challenges in the development lifecycle is the rise of the so called Application Lifecycle Management (ALM) solutions. The roots of ALM are in Configuration Management (CM) and therefore CM systems are usually the foundations of ALM solutions [4].

In this paper our study of ALM is especially focused on the development phase of the SW lifecycle. One of the roles of ALM is to store artefacts produced in the different stages of the development process. These artefacts can be associated with each other, communicated to the relevant stakeholders and relevant real-time reports can be generated from this data to support product development and administration. Because SW development tends to be a global activity nowadays, ALM aims to support global software development with information visibility and consistency in projects. Therefore, in practice, an ALM solution may become a complicated development environment that integrates different kinds of global product information management databases. In our previous industrial studies in the automation industry [5] we discovered that ALM can be supported with a dedicated ALM suite or by integrating a set of proprietary product information management databases.

This paper presents the results of the ToolChain analysis and its demonstration in a telecommunication company. ToolChain is a Proof-of-Concept tool integration solution [6]. The aim of the solution, besides product information transparency, is to improve the traceability of information during global software development. The implementation of ToolChain is composed of a set of Eclipse plug-ins. Eclipse is a well-known platform for tool integration. For instance, Yang & Jiang [7] present a summary of experiences of the use of open Eclipse tool integration framework. In the article, they discuss the benefits and also challenges encountered while integrating tools specific to different lifecycle development phases. The basic intention of ToolChain is to provide an overall frame for traceability and project data visibility where different kinds of tools, especially Open Source tools and databases, can be connected as needed. In its current version, ToolChain has been extended to provide support for test data management, test analysis and simple workflow guidance [6]. The aim of the research is to apply an ALM framework and GSD patterns for analysing a tool integration environment. The analysis clarifies how the current ToolChain version supports ALM features in a global development environment.

Furthermore, it proposes improvement ideas for future development and research related to tool integration environments towards ALM for GSD.

This paper is organised as follows: The next section introduces demonstration settings comprising an ALM framework, GSD pattern language and the research process of demonstration. Then we introduce the results of the ToolChain analysis and demonstration. In section four we discuss the results of this research. Finally, section five concludes the paper.

2 Demonstration

This section introduces the analysis frameworks used in ToolChain analysis, the setting of ToolChain demonstration and methods used for experience collection from the demonstration project.

2.1 ALM Framework and ALM Related GSD Patterns

In our studies, the need to better understand the elements of ALM in order to support the development of ALM in an organization has emerged [8, 9, 5]. In these studies we developed a framework consisting of six principal elements that characterize ALM. This framework can be used for documenting and analyzing organizations ALM solution and thus to support the practical development of ALM in an organization. This framework was constructed and successfully applied in an automation company to support its ALM documentation and improvement efforts. The current framework version contains six elements [5] as presented in Figure 1.

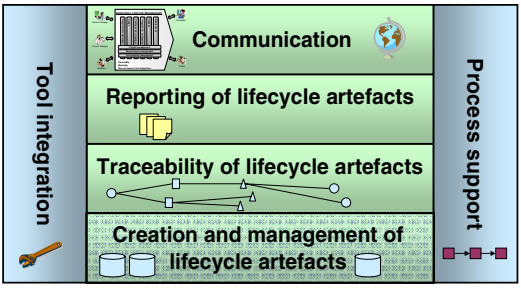

Fig. 1. Principal elements of Application Lifecycle Management [5]

"Creation and management of lifecycle artefacts" is the foundation for ALM. The product information collected and managed by this element is needed, for instance, for traceability and reporting activities. "Traceability of lifecycle artefacts" provides a means to identify and maintain relationships between managed lifecycle artefacts and, therefore, facilitates reporting, change impact analysis and information visibility throughout the development lifecycle. "Reporting of lifecycle artefacts" utilises managed lifecycle artefacts and traceability information to generate needed reports from the lifecycle product information to support SW development and management. "Communication" provides communication tools (e.g. chat) as well as channels for

Table 1. Mapping between ALM elements and related GSD patterns

ALM elements	Related GSD patterns	How GSD patterns cover ALM elements
Creation and management of lifecycle artefacts	Common Repositories and Tools (ID 07)	Global databases to support the management and visibility of lifecycle artefacts.
Traceability of lifecycle artefacts	Common Repositories and Tools (ID 07)	Traceability of lifecycle artefacts in GSD environment.
Reporting of lifecycle artefacts	Common Repositories and Tools (ID 07)	Reporting of lifecycle artefacts and traces in GSD environment.
Communication	Common Repositories and Tools (ID 07)	Asynchronous communication (visibility of lifecycle artefacts)
	Communication Tools (ID 06)	Synchronous/asynchronous communication tools (e.g. net meeting, chat, conference phone, discussion forum)
Process support	Common Repositories and Tools (ID 07)	Process support features such as state models, workflows or process templates.
	Use Common Processes (ID 12)	Common upper level GSD process and ability to tailor the process support for a project or a team on site level.
Tool integration	Common Repositories and Tools (ID 07)	In practice, common repository can be a single central database or several integrated databases.

distributing information about product lifecycle artefacts, links and reports and thus facilitates product information visibility for the whole SW project. “Process support” and “Tool integration” are the elements that are used to configure the ALM solution to support SW development procedures, guide the user through development activities and to facilitate a productive development environment by enabling the user to easily launch tools and transfer information between different tools and databases.

ALM support for Global Software Development can be analysed by using GSD for Project Management Pattern Language [10]. This GSD Pattern Language includes 18 process patterns which have been found to be important in the area of project management in GSD. If these GSD patterns are compared to ALM elements, it can be noted that some patterns relate to ALM elements [10] (Table 1). Patterns named “Communication Tools” and “Common Repositories and Tools” relate to ALM elements. Furthermore, “Use Common Processes” relates to an ALM element called “Process support”. Some other patterns relate indirectly to ALM elements. For instance, an “Iteration planning” pattern uses “Communication Tools” and “Common Repositories and Tools” patterns to support synchronous communication and information visibility during the planning meeting.

2.2 ToolChain Demonstration in the Telecommunication Industry

The ALM features of ToolChain (Figure 2) were analysed by using an ALM framework (for ALM framework see the previous section). The results of the analysis are presented in section 3. ToolChain was demonstrated in a telecommunication company. The solution was introduced to the members of the demonstration group. One person from the company was responsible for the set-up of the ToolChain for their IT-environment. Two of the persons demonstrated the solution in the company.

The following figure 3 shows the activities that are covered by the current version of ToolChain. It further enumerates the plug-ins that exist in the current toolbox and presents the traceability model of the ToolChain traceability database. The demonstration was set-up as follows:

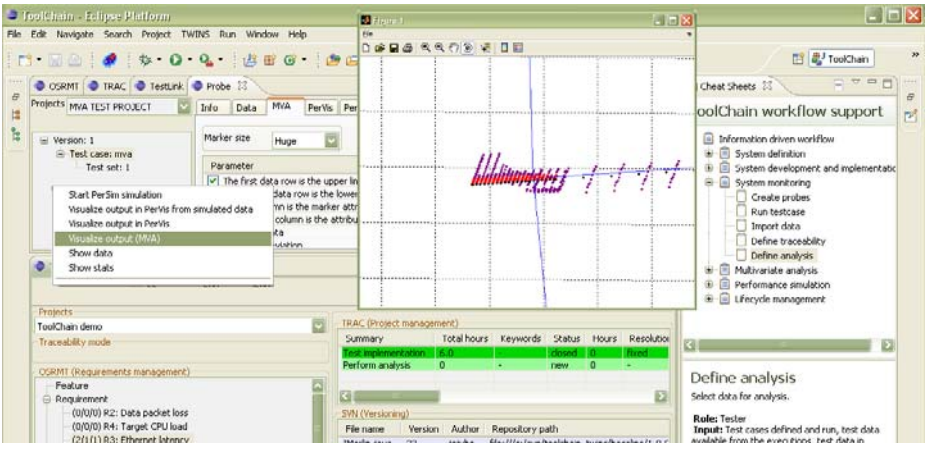

Fig. 2. ToolChain screenshot with MVA (MultiVariate Analysis) tool launched with test dataset

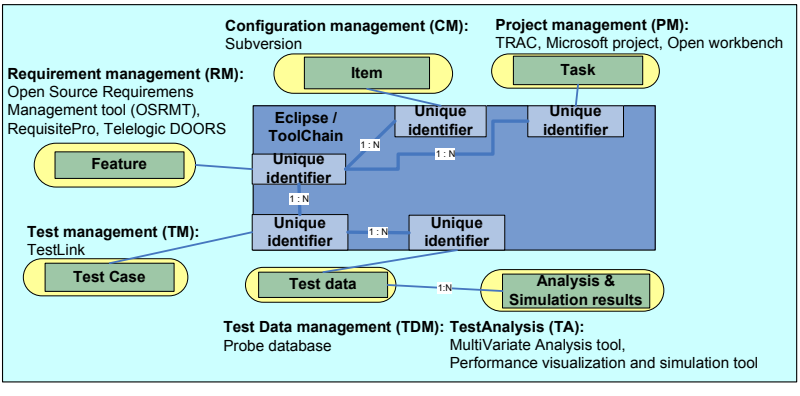

Fig. 3. ToolChain tools, items and traceability model

Demonstration project: The aim of the demonstration project was to measure the performance of the Espotel JiVe platform's Ethernet connection**.** Espotel JiVe is a portable platform with a touch screen and several interfaces [11]. It is also a product that can be tailored to customer specifications and brand. Furthermore, the target of the demonstration project was to study the applicability of ToolChain with test data management and analysis features. The demonstration project defined performance requirements with OSRMT-tool (Open Source Requirements Management Tool) and related testing tasks with Trac-tool. The tasks were associated with the requirements in ToolChain. Test cases were defined with TestLink and associated in ToolChain

with the requirements. The tests were run on the JiVe platform and test data was collected, processed and further imported into ProbeDB [12]. The test data was associated with the corresponding test case in ToolChain. Test analysis parameters were defined in ToolChain. MultiVariate Analysis tool [13] was launched with the selected ProbeDB dataset and appropriate parameters from ToolChain (e.g. check how varying of packet/s & frame size affects packet loss on the target platform). The results of the MVA analysis have been given to Espotel and they can be used in future JiVe platform optimisation projects.

ToolChain set-up: ToolChain was used with the following set-up:

- Requirements Management: *OSRMT (Open Source Requirements Management Tool)*
- Project Management: *Trac*
- Test Management: *TestLink*
- Test Data Management: *Probe database (ProbeDB)*
- Test Analysis: *MultiVariate Analysis* (*MVA)*
- Configuration Management (CM): *Subversion*

Data collection: The experience data from the usage of ToolChain was collected via a questionnaire. The questions were chosen based on the ToolChain tools and ALM framework elements. A comment field was included in the questionnaire where respondents could also give general comments about the solution and other improvement proposals. Furthermore, the results of the questionnaire were presented, discussed and complemented in a workshop session. In this session we discussed the results and clarified issues that were unclear after the questionnaire. Industrial participants in the workshop session represented specialists in SW development and testing and software process improvement (SPI). The summary of results of the questionnaire and the workshop session are presented in this paper. Results were reviewed by the demonstration group members and workshop participants.

3 Results

We analysed the ToolChain prototype with an ALM framework (Table 2) and collected experiences from ToolChain demonstration (Table 3) using the framework.

The current version of ToolChain is a proof-of-concept showing that product development can be supported with test data management and analysis in an integrated development environment. However, different organisations and projects tend to have different needs for development and management tools. Therefore, the ToolChain concept should be refined so that it could be easily adapted into the context of an organisation or project where the needs for ALM features can vary (e.g. tool integrations, traceability needs, reports, development methods). During the workshop session workflow support was identified as an important improvement issue that needs more research.

Originally, ToolChain was developed to support especially global software development. Therefore, current extensions have been technically designed so that they support data visibility and traceability also for distributed development teams. If

Table 2. Summary of ToolChain features according to an ALM framework

ALM elements	ToolChain features
Creation and management of lifecycle artefacts	Tools available for creation and management of lifecycle artefacts. For each development phase, tools and lifecycle items are listed in Figure 3. Development lifecycle activities covered are: Requirements Management, Project Management, Test Management, Test Data Management, Test Analysis, CM.
Traceability of lifecycle artefacts	Traceability is implemented by using a traceability database (MySQL/JDBC database) where the configuration item IDs and links between the IDs are used to store the traceability information. ToolChain uses Drag-and-Drop functionality for traceability collection. Item IDs are stored in traceability database and any other information related to items in the traceability view is retrieved from up-to-date tool databases. Therefore, possible information inconsistencies are avoided. Figure 3 presents the traceability model of the ToolChain.
Reporting of lifecycle artefacts	ToolChain does not contain reporting features yet. However, good potential for this feature since previous two features are covered and thus provides possibility to use lifecycle and traceability data for reporting purposes.
Communication	Communication is supported through information visibility in ToolChain user interface for project members (Figure 2) (asynchronous). There are no communication tools integrated into ToolChain (synchronous (e.g. chat, Internet phone) or asynchronous (e.g. discussion forums).
Process support	Each tool in ToolChain has its own features for process support. Usually items have some kind of state transition model that support the operation according to defined procedures. ToolChain also contains modifiable textual workflow guidance implemented with CheatSheets (see Figure 2) that guides the user through the development process. However, there are no process templates that could be used to configure the tool environment to support different development processes.
Tool integration	A number of tools are integrated as plug-ins or interfaces in ToolChain (Figure 3). Therefore, it is possible to launch several tools from ToolChain environment and transfer data between the tools. However, it is not possible to launch all tools (stand-alone) from ToolChain.

Table 3. Summary of results from industrial demonstration

ALM elements	+ (positive comment)	- (what to improve)
Creation and management of lifecycle artefacts	Basically tools integrated in ToolChain are easy to use.	Support for test automation is missing. Tools have different user interfaces. This complicates the usage of the tool environment.
Traceability of lifecycle artefacts	Strength of the ToolChain. Traceability that usually happens on a "mental" level is supported by the ToolChain.	Ability to compare test data from different development iterations is needed. Study the possibility to assign traces between test parameters and test data.
Reporting of lifecycle artefacts	No specific comments, since this feature is missing.	Reporting would bring additional value to ToolChain. For example, report about requirements/features, related test cases, related test results and related analysis figures. The reports could be made visible through a Project Portal that would facilitate the real-time information visibility for the project.
Communication	Common project traceability and status information can be seen from ToolChain environment.	Communication means would bring additional value to ToolChain. E.g. chat for discussion or Project Portal as a forum for exchanging project information, searching information and storing experiences.

Table 3. (*continued*)

Process support	Workflow replaces traditional Help-files and tool instruction documents. Makes it easier for new users to start using the system	Now allows just textual descriptions. Support for e.g. pictures is needed to support new users. Workflow variants should exist for users with different skills (new user, advanced user).
Tool integration	Important feature. Possibility to use/integrate different databases. Possibility to collect information from several databases into one view.	Harmonisation of user interfaces. Quick launch for all applications would increase the usability (launching stand-alone tools from ToolChain).

compared with key GSD patterns that relate to ALM elements [10], it can be noted that the current version of ToolChain provides only partial support for them. The GSD pattern, named "Communication Tools", refers primarily to tools such as chat, discussion forum, teleconference, etc. ToolChain does not support this pattern since there are no communication tools integrated into the ToolChain environment. However, ToolChain provides moderate support for the GSD pattern "Common Repositories and Tools" as it supports all other pattern-related ALM elements except "Reporting of lifecycle artefacts". However, the GSD pattern "Use Common Processes" is not supported since in the current version of ToolChain, the workflow feature is static and local, demonstrating a workflow description that is closer to help-file that is embedded into Eclipse as a view. Therefore, the functionality of ToolChain needs to be extended to support a common GSD process and its local adaptations as described in "Use Common Processes" –pattern [10].

4 Discussion

Yang & Jiang [7] argue that Eclipse is a cost-effective and productive development environment to support lifecycle software development. Our demonstration results showed that Eclipse-based ToolChain has many advantages during the development and testing process. A general positive comment related to the ToolChain's ability to collect different tools and databases together that are needed during product development. The traceability that usually happens on a "mental" level is now facilitated with the ToolChain traceability feature. Crnkovic et al. [2] states that it is important that the use of many different tools does not introduce new complexity into the entire (development) process and that information is accessible in a smooth and uniform way. To support this, ToolChain targets provide easy access to interlinked information originated from different product information management databases through a central global traceability view. According to demonstration results, this was identified as a strength of the ToolChain. Negative ToolChain comments related to the number of different user interfaces that is quite confusing and therefore ToolChain should be able to somehow harmonise user interfaces from different tools. The lack of test automation support was defined as a critical deficiency. Furthermore, features that would provide additional value for stakeholders operating in a global development environment are missing. The results of demonstration, ALM analysis and GSD pattern analysis indicate that further extensions are needed, especially related to lifecycle reporting, synchronous communication and workflow support. ToolChain has a good basis for extensions towards lifecycle reporting since it

contains good support for the ALM elements, “creation and management of lifecycle artefacts” and “traceability of lifecycle artefacts”, that have been identified as foundations for effective lifecycle reporting [5].

The concept of workflow was discussed during the workshop. Workflows and workflow management systems offer user-specific guidance and coordination enabling the person to perform his/her work tasks independently by using the help of a workflow system. According to the literature, there are several studies related to workflows, workflow applications and workflow management systems. A number of papers were written in the middle of the 90’s about workflow technology [14, 15, 16]. Based on the literature, it can be noted that the research area of workflows is wide. Despite that, there seems to be challenges. Many challenges relate to the functionality of workflow systems; workflow tools’ capabilities in supporting the performance of complex technical tasks are inadequate [14, 17, 18].

5 Conclusions

This paper presents the analysis of an open source global tool integration environment, called ToolChain, and proposes improvement ideas for it towards application lifecycle management. The demonstration of ToolChain and the collection of improvement proposals were carried out in the telecommunication industry. The analysis was made by using an ALM framework and GSD patterns developed in previous studies in the automation industry. The results of this analysis can be used for further improvement of ToolChain towards features of lifecycle management in a global development environment. The study also showed that process support needs more research especially from a workflow point of view.

The functionality and tool set of the ToolChain prototype has evolved gradually. The demonstration project and ToolChain analysis according to an ALM framework and GSD patterns show that the ToolChain prototype provides basic features such as storing and managing lifecycle artefacts as well as traceability issues. However, more advanced features that would provide additional value for stakeholders operating in a global development environment are missing. The results of demonstration and analysis indicate that further extensions are needed especially related to test automation, lifecycle reporting, synchronous communication and workflow support.

Acknowledgements. This research has been done in ITEA projects named TWINS [19] and MERLIN [20]. This research is funded by Tekes, Espotel, Metso Automation, Nokia Siemens Networks and VTT. The authors would like to thank all contributors for their assistance and cooperation. This work is being supported by the Academy of Finland under grant 130685.

References

1. Van Den Hamer, P., Lepoeter, K.: Managing Design data: The Five Dimensions of CAD Frameworks, Configuration Management, and Product Data Management. Proceedings of the IEEE 84(1), 42–56 (1996)
2. Crnkovic, I., Asklund, U., Dahlqvist, A.: Implementing and Integrating Product Data Management and Software Configuration Management. Artech House, London (2003)

3. Svensson, D.: Towards Product Structure Management in Heterogeneous Environments. In: Product and Production Development. Engineering and Industrial Design. Chalmers University of Technology, Göteborg (2003)
4. Schwaber, C.: The Expanding Purview Of Software Configuration Management. Forrester Research Inc., White paper (July 22, 2005)
5. Kääriäinen, J., Välimäki, A.: Applying Application Lifecycle Management for the Development of Complex Systems: Experiences from the Automation Industry. Accepted to EuroSPI 2009 conference (2009)
6. Eskeli, J.: Integrated Tool Support For Hardware Related Software Development. Master's thesis, University of Oulu, Department of Electrical and Information Engineering, Oulu, Finland (2009)
7. Yang, Z., Jiang, M.: Using Eclipse as a Tool-Integration Platform for Software Development. IEEE Software 24(2), 87–89 (2007)
8. Kääriäinen, J., Välimäki, A.: Impact of Application Lifecycle Management – a Case Study. In: International Conference on Interoperability of Enterprise, Software and Applications (I-ESA), Berlin, German, March 25-28, pp. 55–67 (2008)
9. Kääriäinen, J., Välimäki, A.: Get a Grip on your Distributed Software Development with Application Lifecycle Management. Accepted to be published in International Journal of Computer Applications in Technology (IJCAT) (to be Publish, 2009)
10. Välimäki, A., Kääriäinen, J., Koskimies, K.: Global Software Development Patterns for Project Management. Accepted to EuroSPI 2009 conference (2009)
11. Espotel JiVe platform, http://www.espotel.fi/english/solutions_jive.htm (available 4.6.2009)
12. Vitikka, J.: Supporting Database Interface Development with Application Lifecycle Management Solution. Master's thesis, University of Oulu, Department of Electrical and Information Engineering, Oulu, Finland (2008)
13. Tuuttila, P., Kanstrén, T.: Experiences in Using Principal Component Analysis for Testing and Analysing Complex System Behaviour. In: ICSSEA 2008 (Internation Conference on Software & Systems Engineering and their Applications), Paris, France (2008)
14. Georgakopoulos, D., Hornick, M.: An Overview of Workflow Management: From Process Modeling to Workflow Automation Infrastructure. Distributed and Parallel Databases 3, 119–153 (1995)
15. McReady, S.: There is more than one kind of Work-flow Software. Computer World (November 2, 1992)
16. Jablonski, S., Bussler, C.: Workflow management: Modelling Concepts, Architecture, and Implementation. International Thomson Computer Press (1996)
17. Tan, D., Wandke, H.: Process-Oriented User Support for Workflow Applications. In: Jacko, J. (ed.) HCI 2007. LNCS, vol. 4553, pp. 752–761. Springer, Heidelberg (2007)
18. Workflow Problem Space, http://wiki.fluidproject.org/display/fluid/Workflow+Problem+Space (available 4.6.2009)
19. ITEA-TWINS project, Optimizing HW-SW Co-design flow for software intensive system development, http://www.twins-itea.org/ (available 4.6.2009)
20. ITEA-MERLIN project, Embedded Systems Engineering in Collaboration, http://www.merlinproject.org/ (available 4.6.2009)