IOSR Journal of Computer Engineering (IOSR-JCE)
e-ISSN: 2278-0661, p- ISSN: 2278-8727Volume 10, Issue 3 (Mar. - Apr. 2013), PP 24-26
www.iosrjournals.org

Evaluvation of Applying Knowledge Management System Architecture in Software Development Environment

Mrs. M. Vasumathy**
***Research Scholar, Dept. of Comp. Science, Manonmaniam Sundaranar University, Tirunelveli.*

Abstract: *Knowledge management (KM) become important for organization to take advantage on the information produced and can be brought to bear on present decision. Software Development (SD) is a process that requires lots of knowledge. Developers must know what should be done while developing software, and what to do when changes occurs and how those changes can affect other modules of the system. Knowledge management system (KMS) can support the processes of knowledge creation, storage or retrieval, transfer and application. KMS in SD could help the organization to make tacit knowledge into explicit and therefore decrease the dependency on employees' cognition. This paper is to apply KMS architecture in SD environment to analyze the problems faced by software developers during the software development process and how various companies apply the tools of KM to improve the work situation for software developers and managers.*

I. Introduction

KM can be used as a recognition that employees in an organization, as part of their daily activities, knowledge that valuable to the organization. Software Development(SD) is an activity that analyze, design, and produce code for the problems. During SD, developers need knowledge of the system they work on, its application domain, the organization using it, past and present software engineering practices, different programming languages, programming skills. Among the different knowledge needs, developers must identify the following:

• Knowledge about the system developed emerges as a prominent necessity.
• The design decision of the knowledge about software development applied in the process of transforming the knowledge on the application domain into a production of a source code.

1.1 Major problems related to software development projects

a) Software is an immaterial product, and it can be difficult to get an overview of a total program system, which can be millions of lines of code, to identify all possible error sources. Also, a very small defect might have a lot of influence in safety-critical systems. Thus, just a few lines of code that was lacking will have severe consequences.
b) Other problems can be that the communication between the end-users and the software developers is lacking, or that project management is difficult in an environment where a small bug can take a very long time to correct, and where it is often difficult to determine how much work is left to do on a software module.
c) There are a lot of software projects that deliver software that is highly usable and working. we tend to focus on the unusual things that go wrong because they're more interesting or important than the run-of-the-mill things that go right"..

In Software Engineering, to reuse life cycle experience, processes and products for software development is often referred to as having an "Experience Factory". In this framework, experience is collected from software development projects, and are packaged and stored in an *experience base* . By packing, we mean generalising, tailoring and formalising experience so that it is easy to reuse.

Knowledge Management can be : "To make the enterprise act as intelligently as possible to secure its viability and overall success". Thomas Davenport has defined it as "a method that simplifies the process of sharing, distributing, creating, capturing and understanding of a company's knowledge". If we look a bit more into knowledge management, we find that some important aspects are :

- Survey, develop, maintain and secure the intellectual and knowledge resources of the enterprise.
- Determine the knowledge and expertise required to perform work tasks, organise it, make the requisite knowledge available, "package it", and distribute it to the relevant points of action.
- Provide knowledge architecture so that the enterprise's facilities, procedures, guidelines, standards, examples, and practices facilitate and support active Knowledge management as part of the organisation's practices and culture.

Strategies for knowledge management :

- Codification – to systematise and store inf ormation that represents the knowledge of the company, and make this available for the people in the company.
- Personalisation – to support the flow of information in a company by storing information about knowledge sources, like a ”yellow pages” of who knows what in a company.

Another strategy than the two mentioned above could be to support the growth of knowledge – the creation of new knowledge by arranging for innovation through special learning environments or expert networks. When we go on to discuss computer systems that support knowledge management, we will restrict the scope to systems supporting the first two strategies.

1.2 The Experience Factory

One way to manage knowledge is by giving the responsibility for capturing and reusing experience to a separate part of the development organisation. This is the idea behind the “Experience Factory”; a technical and social knowledge management infrastructure to reuse life cycle experience, processes and products, which has been very much referred to in the software.

II. Knowledge Management In Software Engineering

We can say that a knowledge management “program” or “system” in a company can consist of engineering field. Experience is collected from software development projects, and are packaged and stored in “knowledgebase” for knowledge management, that is, what are the company, and how does it proceed to achieve them. Usually, the goals within software engineering are to develop software with less cost, or with a higher quality. But it can also be to make the work of software engineers easier.

2.1 KM TOOLS

A *tool* to support knowledge management is a software system where operational information, or “knowledge”, can be found by different practitioner groups of a software company , usually on an Intranet. The knowledge can be represented in databases, web-pages or files. Another way to represent knowledge in such a system to make it easy to find relevant information later is to use Case Based Reasoning. We see knowledge as something dynamic, that might be changing over time, so a knowledge management tool, must offer possibilities for revising and discarding knowledge, as well as supplying new knowledge into the system.

2.1.1 Case-Based Reasoning

Many tools have been designed to support knowledge management in software development, for example the Experience Management System. Many have used Case-Based Reasoning (CBR), for retaining and retrieving experience, and for building learning software organisations.

Yet other work has been done on using ideas from Experience Factory in the construction of CBR systems, for process improvement in developing educational software.

2.1.2 BORE

The University of Nebraska-Lincoln has developed BORE, a research prototype system for knowledge management support in software development . This is a tool which contains information in cases about some problem solving experience, and in descriptions of resources like tools, projects, people and development methods. These descriptions are used to find which solutions are relevant when software developers are faced with a new problem.

2.1.3 CODE

Another prototype system, is CODE – a general-purpose knowledge management system – which serves as a medium for knowledge capture and transfer, as well as editing or “packaging”. Yet another technical implementation of a knowledge management system for software engineering is developed at the University of Kaiserslautern. Here, a comprehensive reuse repository has been developed, with possibilities for advanced search and retrieval mechanisms.

2.2 CASE STUDIES OF KNOWLEDGE MANAGEMENT IN SOFTWARE ENGINEERING

2.2.1 The NASA Software Engineering Laboratory

The first implementation of an Experience Factory was at the NASA Software Engineering Laboratory. Experience in forms of cost data, process data as project methodology information and information on tools and technology used, as well as product data such as change and error information and results on static analysis on

delivered code was collected, and used to develop predictive models and to refine the software processes that is used. The results of this activity is reported as defect rates that went dramatically down (75% from 1987-91, and 37% from 1991-95) ; the cost of producing software went down by 55% from 1987-91 and 42% from 1991-95. Reuse was improved by 300% from 1987-91 and 8% from 1991-95 Finally, functionality was increased five-fold from 1976-92.

2.2.2 Telenor Telecom Software

In an effort to reuse software development experience, Telenor Telecom Software, a company with 400 software developers in five geographical locations, decided to improve the estimation of software development effort, as well as risk management . To achieve this, they set up:

- An experience reuse process, with new and modified role descriptions.
- An experience database tool, available on the Intranet.
- Resources allocated for experience reuse and for experience database administration.

The experience database was available as an "expert system" which would ask you questions on the nature of a new project, and recommend an estimation model, based on data from earlier projects in the company. It would also give you information on company experts on estimation. This database was linked to a risk management module, which included risk factors found from interviewing experienced project managers. This module consisted of a set of "best practise" processes, a tool to identify, assess and store risk factors, and a tool to visualise risk exposure over time. In addition to this, new roles for "experience database administrators" were set up – responsible for technical and editorial contents, as well as several roles for "process analysts", responsible for analysing information from processes such as the estimation process, project management process and the testing process.

2.2.3 Ericsson Software Technology

Ericsson Software Technology in Sweden have experimented with transfer of experience on a site that develops a wide range of software applications, having around 1600 employees who work in business units of 20 to 30 people. They develop software for telephone switches, base stations and mobile phone management systems. The company has formal communication channels such as meetings, e-mail and written reports, but wanted to establish a corporate culture that facilitate more oral communication of experience. Two organisational roles were invented: "Experience brokers" keep track of what other people in the company know, and match people who can have a benefit from talking to each other. "Experience communicators" help other people solve problems, by teaching them how to solve the problems on their own. The study reports that employees are more motivated when they know that there is a system for
transferring experience that works.

III. Conclusion

We find several companies that wanted to improve the situation for their software developers, but did not have clear goals with respect to quality or development costs. Ericsson Software Technology would come in this category. At NASA, they had cost reduction and quality improvement as a primary goal for their knowledge management activity.

We find claims like that the Knowledge Management systems have saved time, made work easier, and removed problems due to new personnel that existed before. In addition, we can expect most of the companies in the software business to be "more effective" every year, because computers and software tools work faster.

References

[1] T. Dingsoyr, " An evaluation of Research on Experience Factory ," *Proc. Of the workshop on Learning Software Organisations at the International conference on Product- Focused Software Process Improvement,* 2000.
[2] I. Sommerville, *Software Engineering,* Addison Wesley, 1996.
[3] "Chaos", Dennis, Massachusets, The Standish Group Report 1995.
[4] B. Joy and K. Kennedy, "Information Technology Research : Investing in our Future ," Report from the President'sInformation Technology Advisory Committee, 1999.
[5] R. L. Glass, "Talk About a Software Crisis – Not!" *The Journal of Systems and Software,* 2000.
[6] T. Dingsoyr, " A lifecycle process for experience databases," *Proc. Of the ICCBR'99 workshops : Challenges for case-based reasoning :",* 1999.
[7] R. Bergemann and M. Goker, "Knowledge Management of Software Engineering Lessons Learned," *Proc. Of the 10th International conference on Knowledge Engineering,* 1998.