New Challenges for Version Control and Configuration Management: a Framework and Evaluation

Tapani Kilpi
Department of Information Processing Science, the University of Oulu
Linnanmaa 90570 Oulu, FINLAND
email: tapani@rieska.oulu.fi
fax: +358 8 553 1890

Abstract

Software production of today lays high demands on the capabilities of software product management (SPM) process. SPM is defined as a process basing strongly on the well-known approaches of version control (VC) and configuration management (CM). The improvement of SPM demands well designed and implemented VC- and CM-processes on bottom. For this reason a selection of a suitable VC/CM-tool is an important part of the SPM improvement process. It is necessary to be able to specify the right tool selection criteria for each specific case.

1. Introduction

Software industry is developing strongly towards becoming a product business. This lays increasing demands for the development of software product management process and its tool support in the companies. Software product management (SPM) is defined as a process consisting of version control (VC), configuration management (CM), product management (PM) and total product management (TPM). Each of these sub-processes manages an archive containing product related information. Version control manages and keeps track of the *configuration items* which are any documents created during a software development process, and which are found necessary to be placed under configuration control: requirements documents, data flow diagrams, design documents, source code, test results etc. Configuration management controls *configuration data*, i.e. which components belong to a certain product, and also means of building the product. Product management controls *delivery data*, i.e. what exact combination of product components and to which environment have been delivered to some specific customer. Finally, total product management controls *customer data* and *bug records*. This covers all customer contact and feedback data as well as the identified bugs. In an ideal situation both of these can also be tracked back to the source code level.

The problems companies have with SPM can normally be classified into two main-groups: process-related and tool-related problems. In this paper a real world example of SPM analysis and improvement plan is presented. The ideas presented here base on the findings of TPM-Project (Towards Total Product Management in Technopolis Oulu) that is run by three companies each having an objective of improving the maturity of software product management. This paper describes the selection of VC/CM-tool for the TPM-companies. The VC/CM-tool selection was found to be of primary meaning as the first actual development step after the analysis of SPM maturity in TPM companies. This can also be derived from the definition of SPM that presents VC and CM as the core of SPM. In TPM the first step of SPM improvement plan consisted of the definition of the VC/CM-processes model and the implementation of them. The selection of suitable VC/CM-tool for the needs of TPM-companies was an essential part of this. This paper describes the VC/CM-tool selection process, the ideas behind the framework used for the selection and also the result of the selection.

2. Problems explored in software product management

The ideas presented in this paper base on the findings in TPM project (Towards Total Product Management in Technopolis Oulu). The objective of the project is to define and implement an ideal and at the same time a realistic software product management (SPM) model for three companies participating the project [6, Kilpi 1996.] Software product management is defined to consist of the following sub-processes: version control (VC), configuration management (CM), product management (PM) and total product management (TPM) [1, Auer

1996]. The sub-processes form together a hierarchy that is illustrated with the necessary data storages in figure 1.

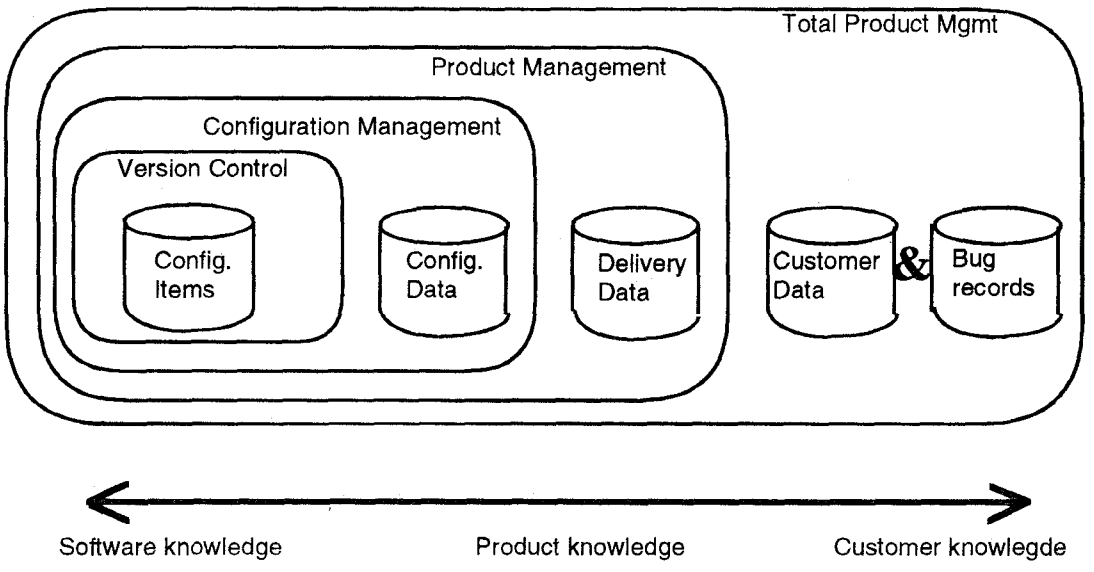

Figure 1. Software Product Management [1, Auer 1996].

The original status of SPM processes in the TPM companies were evaluated and analysed by using Pr²imer method [7, Pr²imer 1996] developed by VTT Electronics of Finland. The question series of Trillium method [8, Trillium 1994] were used in combination with Pr²imer for the evaluation part. Because the aim of the project was to define a common model for all the three companies, also the problems were analysed together. The problems of SPM found during the analysis phase are presented in table 1.

	Problems
Configuration Items	• version control is missing from source codes and other documents • tools and 3rd party components are not version controlled • reusable components are neither identified nor used
Product Data	• the decision of new product revisions is not documented • source codes are not completely module tested, nor have they any quality metrics • changes are neither tracked nor verified
Delivery Data	• traceability from specifications to source code does not exist • delivery and order processes are not documented • the identification of all deliveries (serial number) is not fully seamless • the delivery database is not complete in all deliveries • testing procedures are not integrated to the delivery process
Customer Data	• a lot of time is spent to solve customers' problems in their attempts to use the products in different environments • all customer data is not saved (for example customer environment data) • customer reports are neither formally documented nor analysed • service requests are not fully handled and tracked • minor bug fixes are not informed to all customers
Change Request Data	• change requests are neither formally documented nor analysed for process improvement

Table 1. The problems of Software Product Management Processes in TPM-companies

The three TPM-companies are all quite small-sized, and locate in Technopolis Oulu. **QPR** (Oy Quality Production & Research Ltd) specialises in developing, manufacturing and marketing of software tools on business management area. At the moment there are about 30 employees in QPR. The technical environment of QPR is based on a Novell network. The workstations are PC-based, using Windows95 and WindowsNT. The main tools in development are Borland Delphi and Visual Basic. **Modera** (Modera Point Oy) specialises in systems developed for waterworks, peat suppliers and telecommunications. The current number of employees is 10. The technical environment of Modera is based on a Novell network. The workstations are PC-based, using Windows. Internet-addresses are provided for almost all employees. **Prosoft** (Prosoft Oy) has specialised in developing, manufacturing and marketing the testing tools for embedded software. The number of employees is 5. The technical environment of Prosoft is based on TCP/IP network with Sun server. The workstations are PC-based using Windows95, WindowsNT and OS/2. Internet-addresses are provided for almost all employees.

3. Software product process in the TPM-companies

In the next phase of TPM-project a model of software product process was defined for managing the software change process in the TPM-companies efficiently. This was found necessary in order to define the objectives of the project on a detailed level. The redefined process model is to remove as many as possible of the problems found in the original status analysis of the companies. In software industry the requirement for changing products is caused by the needs of the market and the customers. The task of a software company is to satisfy the needs of their customers, and to be able to correspond to the changing demands in the market.

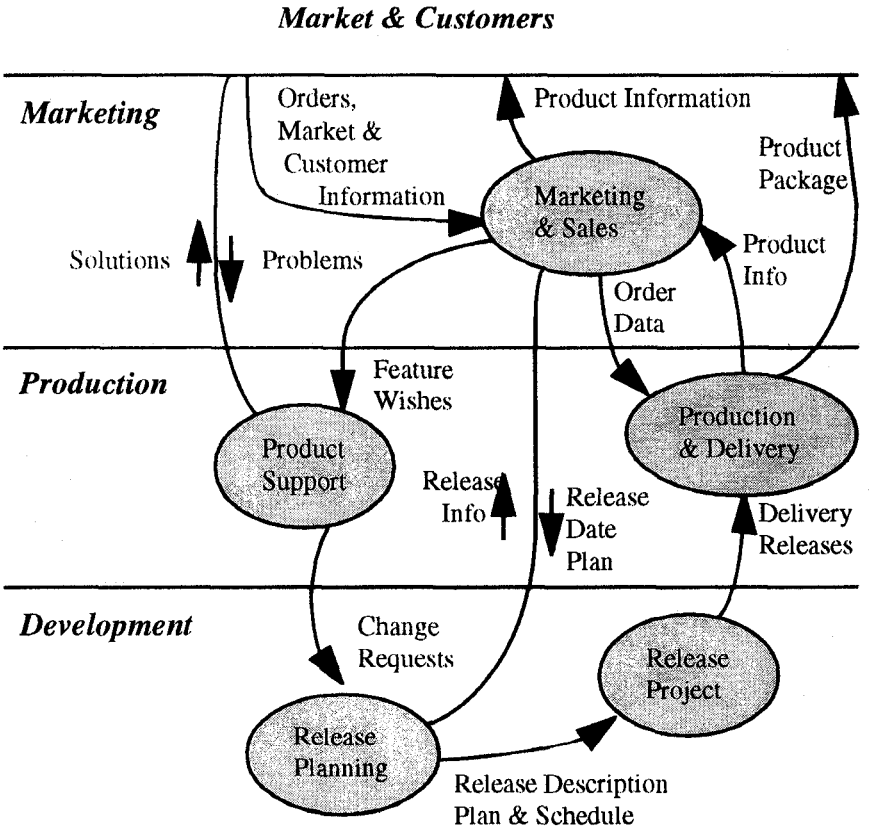

Figure 2. Software Product Management Process in TPM-Companies

The function of a company is modelled with three activity areas in the TPM-process model, which are: the *Development,* the *Production,* and the *Marketing* activities.The three activities complete each other, and have been designed to work together and to complete each other in order to reach the company objectives. The

Marketing activity collects and analyses the market and the customer information, as well as informs the market and the customers of the products and the new releases of them. The *Production* activity provides the support facilities for the customers' problems, and takes care of handling the orders and the deliveries. The *Development* activity manages the product change process by analysing all the feedback and the development ideas collected, and by planning the release projects. The general level TPM-process model is illustrated in figure 2.

The model consists of five main processes: the *Marketing & Sales*, the *Product Support*, the *Production & Delivery*, the *Release Planning*, and the *Release Project*. The *Marketing & Sales* process collects the Market Information and the Customer Feedback, and produces Feature Wishes basing on the results of the information and the feedback analysis. The Feature Wishes are then passed to the *Product Support* process where they are classified and saved together with the Customer Problem data in the Change Request format in the Change Request database. Solutions to the customers' problems are normally provided by the *Product Support* process, and sent to the customers, and the unsolved ones are saved as Change Requests. The general level Product Strategy is defined as a part of the *Marketing & Sales* process, and the Release Date Plan basing on the Strategy is delivered to the *Release Planning* process. The *Release Planning* process starts to plan a new release basing on the Release Date Schedule by analysing all the collected Change Requests.

As a result of the *Release Planning* process detailed Release Description, Plan, and Schedule are delivered to the *Release Project* process for starting the development work. The *Release Planning* process also sends Release Info of the next release to the *Marketing & Sales* process. The *Release Project* process produces a new release of a product corresponding to the Release Requirements. The accepted new Releases are delivered then as delivery Releases to the *Production & Delivery* process for archiving and managing deliveries. The deliveries base on the Customer Orders that are passed to the *Production & Delivery* process normally through the *Marketing & Sales* process. In a delivery a customer gets a Product Package containing all the product components, the manuals, the examples, etc. The *Production & Delivery* process also sends Product Information to the use of the *Marketing & Sales* process.

A successful implementation of the process model presented in this section lays high demands for VC/CM process supporting it like presented in section 1. Along with the traditional VC/CM needs the supporting system should also be capable of saving customer, delivery and bug information. The customer interaction and bug information are in most cases related to some specific versions of a product. An analysis of the feedback collected is not possible if the former product versions and their building environments can not be regenerated. All the former deliveries made should be able to be rebuilt in their original form in a company. The VC/CM system should support along with this also saving of all the delivery-, bug- and customer related data possibly needed for development and production processes.

4. Criteria for VC/CM-tool selection

After defining the problems (section 2) and the SPM process model (section 3) it became obvious that the tool support of SPM needed to be renewed in the TPM companies. In the first place this meant a selection of a suitable VC/CM-tool and a definition of a selection criteria for it. On the one hand the criteria had to be tool oriented and on the other hand SPM process oriented. In outlining the demands for a VC/CM-tool the CM-services model [4, Dart 1992] and the CM requirements [2, 3, Dart 1990, 1991] were applicated.

The services model ties together process model and user roles. It does this through the definition of services independent of user role and process. The user picks and tailors the services to suit a particular role for a particular process. Figure 3. gives an indication how the services model was used in TPM. The services selected are made out of various mechanisms that have been customised and integrated together to meet certain policies. The role used is the VC/CM-needs of the TPM-companies. These needs can straightly be led from the SPM problems presented in section 2. To analyse the needs from the point of view of tools the functionality of VC/CM-systems needs to be defined and understood first. Dart (1992) [4] has defined the CM functionality areas as follows:

- **Components:** identifies, classifies, stores and accesses the components that make up the product.
- **Structure:** represents the architecture of the product.
- **Construction:** supports the construction of the products and their artefacts.
- **Auditing:** keeps an audit trail of the product and its processes.
- **Accounting:** gathers statistics about the product and the process.
- **Controlling:** controls how and when changes are made.

- **Process:** supports the management of how the product evolves.
- **Team:** enables a project team to develop and maintain a family.

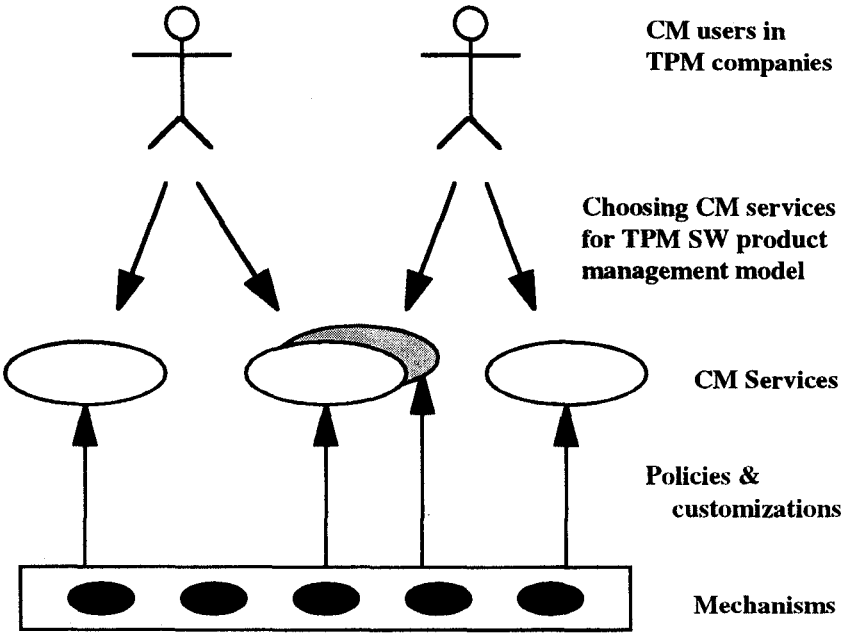

Figure 3. Applicating the CM Services Model for TPM needs [4, Dart 1992].

The demands towards the tools were outlined by analysing the practical VC/CM-experiences of the companies through the functionality areas. As a result of this a set of CM services required was defined. The set of services is strongly tool oriented because it is meant to be applicated straightly for tool evaluation. The services are called here the tool evaluation criteria, and they are presented in table 2.

CM services for TPM needs / tool view	
Components	1. creation of a new project 2. creation of a new file 3. making project specific variants of components
Structure	4. taking a commonly usable component into use in a project 5. using 3rd party components (libraries)
Construction	6. handling files through using menus (VC++) 7. version management of development tools
Auditing	8. listing the revision history of a single file 9. listing the revision history of the whole project 10. listing the revision history of the whole project between two releases of the project
Accounting	11. listing which files are locked in a project
Controlling	12. parallel use of the same files by more than one developer 13. locking of file revisions against further changes (before 'promotion')
Process	14. definition of a 'promotion' hierarchy (draft, proposal, accepted, published, obsolete) 15. automation of actions when 'promoting'
Team	16. defining different 'roles' in the development process

Table 2. CM services for software product management model of the TPM project

In defining the CM-services the TPM project group found several SPM service needs in TPM companies which are outside VC and CM. These needs had to be left out of the set of services because the primary goal was to select a VC/CM-tool. Including the outside features to the criteria might have interfered the results of the tool evaluation. However, the necessary product management features will be implemented later during TPM project to the SPM support system of the companies. Some additional tool support will be needed for the SPM features, which had also to be noticed on a general level in the tool selection. The general level knowledge concerning this was collected mainly through benchmarking in other companies.

5. Evaluating the long list of VC/CM tools

The CM-services defined in the previous section were applied as selection criteria for VC/CM-tool selection in the TPM-project.

The long list	Notes
1. Aide-De-Camp	- requires UNIX server - change set model as a primary CM-model
2. CCC/Harvest	- requires UNIX server - change set model as a primary CM-model
3. CCC/Manager	+ quite good functionality for TPM needs - interferse with the way a developer would normally work
4. ClearCase	+ second best functionality for TPM needs - requires windows NT or UNIX server - Windows client /Attache) only a subset
5. CMZ	- poor functionality for TPM needs - a bit academic to be an industrial strength tool
6. Continuus/CM	+ the best functionality for TPM needs - requires UNIX server - the most expensive
7. ExcoConf	+ covers much of the functionality required by TPM + good overall impression - no clear picture of the capabilities (project concept?) - product literature hard to understand
8. MKS Source Integrity	+ cheap - no version tree, slow reports, file based
9. PVCS	+ large installed base + TPM-companies' earlier experiences of PVCS + offers a large area of functionality for TPM needs - expensive because of a lot of separate add-on tools - GUI just a recent feature - file oriented - slow - standard report format is not very good
10. RCE-Revision Control Engine	- no project concept, no support for process management - poor functionality for TPM needs
11. VCS-UX	- long transaction model as a primary CM-model - the model feels a bit "strange" - product literature hard to understand
12. Visual Source Safe	+ cheap - no support for process management - strange branching - poor functionality for TPM needs

Table 3. The tools of TPM "Long List" and notes made of them

The use of the criteria demands installation and test use of all the tools that are to be tested. Because the selection of the VC/CM tools on the market is numerous at the moment, it was not possible to test all the tools in TPM. A pre-evaluation was found necessary in order to find a set of four tools for the final selection where the selected tools were to be examined in detail. The pre-evaluation was based on the literature, brochures, benchmarking, interviewing, internet and the experiences of the TPM companies. Preliminary filtering was done by using platform availability and pricing as criteria. Through this method a "Long List" of twelve tools was gathered: Aide-De-Camp, CCC/Harvest, CCC/Manager, ClearCase, CMZ, Continuus/CM, ExcoConf, MKS Source Integrity, PVCS, RCE-Revision Control Engine, VCS-UX and Visual Source Safe.

The twelve tools were examined closer in order to reduce the number of the final candidates. The selection criteria used for this phase was:

- availability of the tool on platforms used by the TPM companies
- functionality required by the TPM project objectives
- pricing
- the CM-model behind the tool

In estimation of the CM-model the four CM-models [5, Feiler, 1991] were applied as a base. The four models could be characterised as follows. The *checkout/checkin model* offers version management of individual components. The *composition model* focuses on improving the construction of system configurations through selection of alternative versions. *The long transaction model* emphasises the evolution of systems as a series of configuration versions and the co-ordination of concurrent team activity. The *change set model* promotes a view of configuration management focused on logical changes. Typically one or two of the models are supported by a CM system/tool. Each of the tools selected to the long list were then examined on a general level. The notes made each of them are presented in table 3.

Four tools were selected to the "Short List" of TPM through analysing the notes presented in table 3. Up to the improvement plan these tools were to be installed for trial use in the TPM companies, and evaluated through experimenting there. The tools selected to the Short List are: ClearCase, ExcoConf, PVCS and SourceSafe.

6. Evaluating the short list of VC/CM tools

Each tool of the TPM short list except ClearCase was evaluated up to the detailed testing plan. The results of the evaluations of ExcoConf, PVCS and Visual SourceSafe are presented in table 4 where the support provided for each of the CM services used as the criteria in the tool selection are marked with one, two or three stars (one star is the lowest support and three stars the highest). The results base on the test use done in TPM companies, as well as the criteria defined for TPM tool evaluation in section 4.

The evaluation of ClearCase VC/CM tool was prevented because some serious problems were encountered in installation and configuration of the tool. The relatively hard platform demands of ClearCase were complied, but problems were encountered with the network configuration. The Attache part of the ClearCase suite was not able to authenticate users, probably because it was incompatible with the domain controller which was running OS/2 Lan Server Program 3.0. ClearCase tool was initially developed and targeted for the UNIX platform, and probably for this reason the administrative demands of it appear to be relatively complicated in relation to other types of platforms.

Considering the experiences having with ClearCase at this point of the project, and also because of the lack of experienced system administration resources in the TPM companies, it was decided to abandon further investigation of this tool in the TPM project. In addition to the criteria based on the evaluation of table 4 the test use of the three tools produced also a general level comments of each of the tools. These descriptions give an overview of the tools and their suitability for TPM-project needs. The descriptions are as follows:

ExcoConf implements a hierarchical folder structure, where directories are stored as folders which can contain documents and other folders. ExcoConf can manage entire directory trees. Each document or folder can exist in either of the two states: 'released' or 'work'. The released versions of the documents are stored under the folder's 'release' directory, and the 'work' versions are stored in 'work' directories. For multiple users, it is possible to specifically lock files so that all users can not access all files. The file saving system of ExcoConf uses a lot of disk space. ExcoConf supports branching, but it relies on third-party merge and branch tools (a user must provide his own merge and branch). Also, there is no special make tool; only a pre-processor to process the make files so that

logical file names can be used instead of physical ones. ExcoConf is available on VMS, UNIX and Windows 3.1, and the Windows 3.1 version runs also under NT and Windows 95. ExcoConf is fairly easy to install and maintain. The graphical user interface is a bit awkward. It supports file drag-and-drop and file associations, but it must be kept open practically all the time, and it takes a lot of room in a display.

CM service	ExcoConf	PVCS	Visual Source Safe
1. creation of a new project	* *	*	* * *
2. creation of a new file	* *	* *	* *
3. making project specific variants of components	*	*	* *
4. taking a commonly usable component into use in a project	* *	* *	* * *
5. using 3rd party components (libraries)	*	*	*
6. handling files through using menus (VC++)	*	* * *	* *
7. version management of development tools	*	*	*
8. listing the revision history of a single file	*	* * *	* * *
9. listing the revision history of the whole project	*	* *	* * *
10. listing the revision history of the whole project between two releases	*	* *	* *
11. listing which files are locked in a project	*	* *	* *
12. parallel use of the same files by more than one developer	*	* * *	* *
13. locking of file revisions against further changes (before 'promotion')	*	* * *	*
14. definition of a 'promotion' hierarchy (draft, proposal, accepted, published, obsolete)	*	* * *	*
15. automation of actions when 'promoting'	*	* *	*
16. defining different 'roles' in the development process	*	* * *	*

Table 4. The results of the TPM tool evaluation

PVCS implements a hierarchical folder structure, where directories are stored as folders. A problem with PVCS folder structure is that it is just a one level hierarchy, and the creation of a hierarchy of more than one level has to be done by manual creation and linking of working directories. PVCS does not support a creation of separate personal working directories for different members of a project team. This can only be managed by creating different projects for different users, but this causes a mix up with varying settings and policies of the users. PVCS contains facilities for branching but only one revision can be in a given promotion status at a time. Problems occur with this when bug fixes are done to a recent release and features belonging to the next release have already been added to the "newest" version. The new features are not wanted to deliver to the customer, only the bug fixes. The merging facilities of PVCS worked inaccurately in the TPM test use. Automatic merge did not work at all and the manual one was very complex to use.

Visual SourceSafe proved to be the most easy-to-use of the VC/CM tools of TPM short list. The GUI is well designed, the project view logical and loading time of a project is fast. A major shortage of Visual SourceSafe is that it does not contain a promotion hierarchy feature. Another disadvantage of VSS is that it does not provide the user with a proper diagrammatic view of the revision history of a file. The multiple check-out feature which enables many users to check out the same file affected quite useful in the test use. VSS does not revision files like other VC/CM tools do, instead it assigns version

numbers to each revision of a file. This might cause confusions in normal use.

7. Conclusion

The conclusion of the TPM evaluation team was to recommend the selection of PVCS VC/CM-tool for supporting the SPM-processes in the TPM companies. PVCS collected the highest number of stars in the tool evaluation, illustrated in table 4 in section 6. Considering this result as well as the verbal statements of the TPM evaluation team of each of the tools, the TPM project group found the following reasons for making the selection. ***ClearCase*** was not recommended because of the difficulties in setting it up and getting it into working condition in the platforms of TPM companies. The price of ClearCase and its purchase and maintenance costs were also found to be high. The relatively high expertise required to support the tool was also found unsatisfactory by the TPM project group. ***ExcoConf*** was not recommended because the VC-model it uses was experienced to be unfamiliar to the TPM team, intermediate versions of source files are not stored by it, it requires a lot of disk space for the archives and there are inconsistencies and bugs in the user interface. ***Visual SourceSafe*** was not recommended because the functional limitations discovered in it were found to be fatal. Although many problems were found with the ***PVCS*** tool as well, it was recommended to be selected for TPM because it carries a large installed base of tools and it also has large coverage of functionality required by the TPM companies.

On a general level the remarks made in the TPM project by this far have showed that the development of version control and configuration management processes have to be regarded as a part of developing the whole product management process in a company. In the next phase the TPM project will concentrate on adding product management facilities on top of the already implemented VC/CM-processes. In practice this means design and implementation of customer interaction and bug data management features to the processes. The long term objective of TPM companies is to raise the standard of their software product management to a level that covers also the marketing processes. This lays high demands, for example, on the release planning and scheduling of the products.

The concepts of release project, release planning and release approval need to be defined and formalised, and as the most important part, they need to be implemented to software product management processes of the companies.

On this stage of improvement the software product management reaches a state where it is a part of business process of a company. However, the most important finding of the TPM project by far is the old truth: there is no such thing as a perfect process. That is why the software product management process improvement will be added as a regular and steady part to the business processes of the TPM companies.

References

[1] Auer, A. 1996. Seminar Presentation Material of the Final Presentation of LEIVO-project 14.2.1996 in Helsinki. Organised by VTT Electronics.

[2] Dart, S. 1990. Spectrum of Functionality in Configuration Management Systems. Technical Report. CMU/SEI-90-TR-11. ESD-90-TR-212.

[3] Dart, S. 1991. Concepts in Configuration Management Systems. Proceedings of the 3rd International Workshop on Software Configuration Management, June 1991, pp. 1 - 18.

[4] Dart, S. 1992. The Past, Present, and Future of Configuration Management. Technical Report. CMU/SEI-92-TR-8. ESC-TR-92--8.

[5] Feiler, P. H. 1991. Configuration Management Models in Commercial Environments. Technical Report. CMU/SEI-91-TR-7. ESD-9-TR-7.

[6] Kilpi, T. 1996. Evaluating the Maturity of Software Product Management: A Case Study in Three Companies. Proceedings of the Ninth Australian Software Engineering Conference ASWEC '96 July 14 to July 18, 1996. IREE Society.

[7] Pr²imer. 1996. http://www.ele.vtt.fi/projects/primer/primer.htm. VTT Electronics/Technical Research Centre of Finland.

[8] Trillium. 1994. Trillium: Model for Telecom Product Development & Support Process Capability. Bell/Canada. Release 3.0, December. 118p.