Master's Thesis

A Return on Investment Model for Software Configuration Management

Lorenzo Borraci *Erasmus*

Department of Computer Science
Lund Institute of Technology
Lund University, 2005

ISSN 1650-2884
LU-CS-EX: 2005-19

Lund University
LUND INSTITUTE OF TECHNOLOGY

Department of Computer Scienze

A RETURN ON INVESTMENT MODEL FOR SOFTWARE CONFIGURATION MANAGEMENT

Candidate

Lorenzo Borracci

Supervisor

Prof. Lars Bendix

Ulf Asklund

2004-2005

Contents

List of Figures

Introduction

Software configuration management (SCM) is an important discipline in professional software development and maintenance. SCM has gained in importance as programs have become larger in size, longer-lasting in terms of life-cycle and increasingly mission and life critical. The growing need for product customization and the shorter time-to-market for new products and software updates have led the companies to adopt this discipline and to use SCM tools.
The main SCM objective is to control changes in large and complex software systems, to manage and track the numerous corrections, extensions, and revisions that are applied to a system over its lifetime.
Virtually, everyone on a software project is affected by SCM. SCM is a task that begins early in the product life cycle and is present in all the activities concerning product development, from the requirements and design all the way to the maintenance phase.
For this omnipresence in the software production cycle there are many costs and benefits associated with this discipline. Actually many of the direct benefits are known, but the exact quantification of costs and benefits deriving from the application of SCM are not immediately evident.
There are several aspects concerning SCM that must be considered. Companies can apply different levels of SCM, from a simple naming convention to a complex and integrated change process and can use many different tools, from a simple versioning tools to a complete instrument like ClearCase.

Objectives

The main objective of this study is to create a document that can clarify the costs and the benefits of SCM.
This work is addressed to product managers, to project managers and generally to the people who can take the decision of introducing SCM in the software production process. For this reason, the target of this work is to create a ROI model for calculating the payback and answering the question: ”How much?”. This ROI formula must be as complete as possible and show the economical impact of SCM in order to convince senior management to invest in it.
A few studies concerning this subject can be found in current literature, but they are either incomplete or unreliable. An example of an incomplete study is the one by J.O.Larsen and H.M.Roald's : ”Introducing ClearCase as a Process Improvement Experiment”, where the authors focus only on some aspects of the introduction of ClearCase in a company by measuring some data pre and post-implementation of the SCM tool and process. This is the only academic study in current literature. As for unreliable studies, examples can be found in the documentation deriving from SCM tool vendors, like Merant PVCS, or Rational ClearCase both of which assure impressive benefits without demonstrating them.
This study focuses on the identification of the main arguments concerning the introduction of SCM in a company and also tries to impartially demonstrate the associated costs and benefits. In order to have a wider view, this problem is analyzed from three different points of view: the four activities of SCM, the people who are involved in SCM and the different phases of the product development life cycle.
This work presents different difficulties. First of all the great variety of aspects and parameters associated to this discipline and the difficulties to identify and quantify these parameters. This problem is caused by the subjectiveness of many SCM benefits, because they depend on how much and how much the SCM activities are implemented in a corporation. This aspect makes it very difficult to have a full vision of the impacts of SCM because many costs and especially benefits are hard to quantify.
For this reason, that is the difficulty to create a formula that can exhaustively

describe the global gain deriving from the use of SCM in all the possible cases, the target of this thesis is to create not only a model but also a guide to use it and the ROI formula.
The model gathers most of the parameters associated with this discipline by organizing them in a matrix where these aspects are divided in three degrees of measurability.
The guide has been though in a way to help the potential user of the model to identify and quantify its parameters. The only parameters that are measurable are the inputs of the formula. This way the difficultly quantifiable parameters and the different context and company specific situations can be managed more efficiently by the use of a more complete and elastic model.

Thesis organization

The next chapter will discuss the analysis of the SCM discipline; following is a section that describes the model and the ROI evaluation. The method can be found in the fourth chapter. Chapter five presents the discussions and the thesis is concluded in chapter six with the conclusions of this work.

Chapter 1

Analysis

Analysis is the first step towards the solution of a previously outlined problem. Breaking the problem down in different problem areas is essential for a clear comprehension of its structure. This work is initially divided in three different studies into clarify the different aspects of SCM. The discipline is analyzed from its four main activities to have a good knowledge of it. Then the changes in the phases of the product development life cycle after the introduction of SCM are studied and finally the changes in the personnel working life are considered.

1.1 SCM

Software configuration management is the organization of the components of a software system so that they fit together in a working order, never out of synch with each other.
Wayne Babich describes SCM as "the art of identifying, organizing, and controlling modifications to the software being built by a programming team. It maximizes productivity by minimizing mistakes" [3]; IEEE[1] defines it as the discipline of organizing, controlling and managing the development and evolution of software systems.
SCM is a SEI [2] Capability Maturity Model (CMM) level 2 key process area.

[1]IEEE, Institute of Electrical and Electronic Engineers
[2]SEI, Software Engineering Institute

The Software Engineering Institute affirms that it is necessary to establish and maintain the integrity of the software project products throughout the software life cycle.
The CMM can serve as an example of how SCM brings a software quality and lowered costs improvement. Simply moving from Level 1 of the CMM to Level 2 represents significant improvements. In a SEI report (SEI 92-TR-24), data were averaged over 1233 separate projects in 261 organizations spanning 10 countries, to estimate the benefits of reaching higher maturity levels [6].

Maturity Level	Calendar Months	Effort (Work Month)	Defects Found	Defects shipped	Total Cost
1	29.8	593.5	1348	61	$5,440,000
2	18.5	143.0	328	12	$1,311,000
3	15.2	79.5	182	7	$728,000
4	12.5	42.8	97	5	$392,000
5	9.0	16.0	37	1	$146,000

Table 1.1: Observed CMM Benefits

Table 1.1 shows that going from level 1 to level 2 there is a reduction of the time to market by 38%, a reduction of the defects found by 76%. At the same time the costs at level 2 are 25% of level 1 costs.
This is an enormous benefit; but how can this discipline permit to have this earning? To answer this question it is necessary to show the implication of implementing SCM in the software development production.

1.2 Documentation

The first phase of this work is the documentation and the research of information on SCM and its related economical costs and benefits. Only few works exist on this topic, this fact makes difficult to find the information. There are two main sources:

- Academic text
- SCM vendor's documents

The first source is the more reliable one. Unfortunately there are no specific studies about this subject except for a Norwegian work [5] where the authors studied the impact of introducing ClearCase in a software company. Many theoretical papers have been studied to understand the discipline, how it can be implemented and what changes there are in the software production and in the work of the developers. In this first part of the documentation the guide lectures were the Babich's [3] and Daniels's [8] ones. These lectures permit to have a clear view of SCM from a technical and a managerial point of view.
In literature there are some more specific documents concerning the problem of finding a ROI model for SCM, these studies are available from the SCM tool vendors only, and for this reason, there are many problems of reliability. These documents are created to sell their products and they promise impressive improvement, even if they do not demonstrate them, although for example, in Merant white paper there is a supposed reduction of software and contents error by up 90% [1]. These works are considered to find the major benefits that are associated to SCM. The impact of SCM discipline on the production process is analyzed to search the costs that this discipline implies.
To understand how SCM can be applied and what changes in the software production, it has been necessary to study this discipline from three points of view: technical, production and individual. This work is summarized in the three following paragraphs.

1.3 The 4 activities of SCM

SCM is a management discipline, thus, one of the first and important activities in configuration management is the planning process. The configuration management plan is developed by the configuration management office, and it should be approved by a higher authority. It should reflect how the project is going to be developed. The plan has to fit with the organization of the company, for this reason there are different ways and degrees of SCM implementations. A standard configuration management plan foresees four main activities:

- Identification
- Control
- Auditing
- Status Accounting

These activities must be satisfied regardless of the amount of automation within the SCM process. All of them may be satisfied by a SCM tool, a tool set, or a combination of automated and manual procedures.

1.3.1 Configuration Identification

Configuration identification is the process of recording and communicating information about the requirements, the design, or the product itself. This process is accomplished by use of documentation coupled with naming and numbering schema. This documentation defines the original approved configuration (the baseline) and the approved changes to this baseline. Configuration identification addresses traceability, it permits to be able to identify all artifacts that compose a related configuration item(CI[3]).
Configuration identification divides the product in a hierarchical, top-down manner into small detailed elements (Configuration Items). The Configuration Items are created by the process of decomposition, which starts from

[3]A CI is a component with associated documentation that is designated for configuration control and treated as a single blackbox entity by the configuration management activity.

the beginning of the project to the implementation phase. The configuration manager is responsible for the implementation and the control of naming and numbering systems, once they are established.
This activity adds some direct costs especially for the configuration manager on the requirements and design phase of the project. Extensive collaboration with the rest of the staff is needed to create proper configuration identification. For example in the case of software element naming, the configuration manager requires to work closely with the software engineers in developing effective naming conventions. On a large project, configuration identification could be a wasteful job depending on the automation degree and if it is possible to reuse the already existent scheme. This added work provides some inducted benefits. The traceability addressed by configuration identification helps the tester doing V&V [4]. If the system is easily understood and if the numbers and the names convey information, the maintainers have a very easy job. These benefits provide a reduction of the time spent testing and maintaining the product.

1.3.2 Configuration Control

Configuration control is the control of changes to hardware, software, firmware and documentation. In the context of SCM, ”control” means that proposed changes to a CI are reviewed and, if approved, incorporated in the software configuration.
No product developer has the luxury to write code and than to forget it. Along with requirements and schedules, the software that has been developed changes in response to those other changes. Software is not static. These changes have an impact on budgets, schedules and associated changes to other components. For this reason the change process is a major element of configuration management (sometimes called change management). Depending on organization and product structures, the change control process may be complicated. All the activities of the change process are regulated by the configuration control board (CCB) which is a permanently established committee of representatives of the organization. Its primary mission is to ensure complete impact assess-

[4]V&V, Verification and validation

ment and analysis.
The standard CCB structure consists of:

- *Chairperson*: usually the program manager; it has to be someone who can make a decision affecting the program
- *Secretary*: usually the configuration manager
- *Members*: the CCB could be composed by some members of the staff depending on the politics of the company
- *Specialists*: they are called to a CCB meeting on a case-by-case basis

Figure 1.1 shows, with a flow diagram, a typical CCB process.

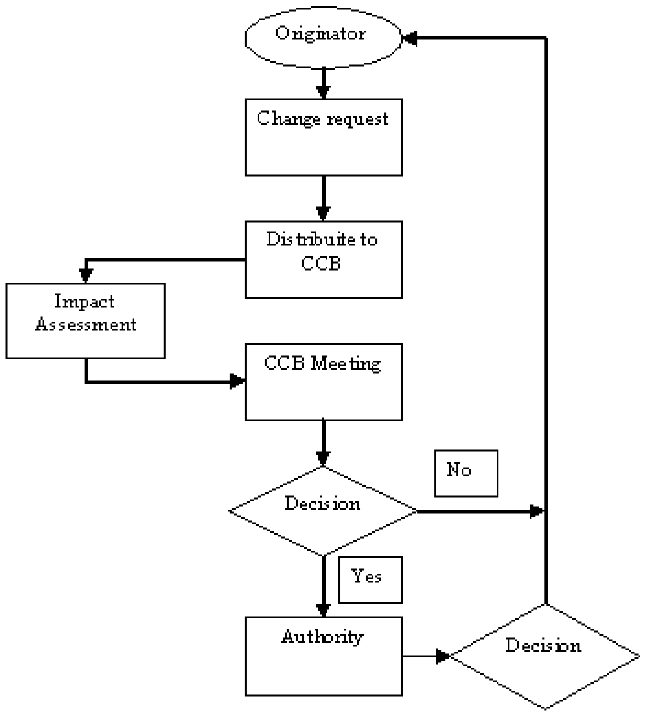

Figure 1.1: CCB Process

The procedure of configuration control adds rigidity to the software development process and some costs, to advantage of more organization and more awareness of the changes in the project. There is an increased work for all the

members of the CCB: the developer has to prepare a formal change request and has to do, when required from the configuration manager, the impact assessment; the configuration manager has to assist to the entire CCB meeting. There is also an inducted cost derived from the loss of developer's time waiting for an authorization.
Configuration control and configuration identification achieve a sense of organization and control instead of chaos; this fact provides stability to the project and an increased productivity of the staff, derived working in a stable developers' environment.
Another benefit of this procedure is the ability to trace the original product through its development and its growth, in order to trace the problems to their origin. This is a benefit especially in the maintenance phase. If a wrong change is made, a previous working version is available. This only capability has saved enormous amounts of time, as developers have tried out specific solutions that did not work in the product environment and were able to back up rapidly to a working version.
Configuration control permits to handle very complex activities like multi-version or multi-installation project. This characteristic helps developers to find and fix bugs in different variants or versions of a product.
Changes are planned, their impact is assessed, their costs are determined and their schedules are set and followed, this benefit is extremely important to finish the project on time and to stay in budget.

1.3.3 Configuration Status Accounting

Configuration status accounting is a system of reports and records documenting CIs and baselines throughout the product life cycle. It means to report and to record product configuration data and can be considered as an information exchange system.
The process is showed on figure 1.2. It begins with an input function; data is stored, there is a capability to manipulate stored data, and the process terminates with the output of the data. The input is provided by the configuration control, identification and audit activities, as data into the status accounting file. The configuration status accounting system has a manipulation mecha-

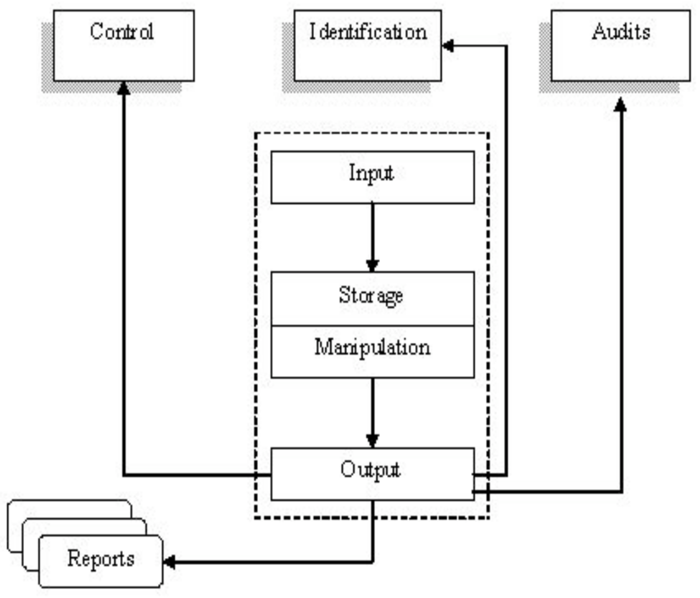

Figure 1.2: CSA as an information exchange system

nism to sort or order data as necessary. The output has two directions. One of them provides reports and archives the contents of the system. The other direction feeds back the upgrades to the three functions (Configuration control, identification and audit), reflecting the transition of the configuration item from CI_n to CI_{n+1}, already considered.
The mission of Status Accounting is to support traceability. Beginning with each defined baseline, the status accounting target is to be ale to show the product's progress to the next baseline tracing the changes.
Configuration status accounting can be made automatically with the help of a SCM tool, or manually.
Depending on the degree of automation, the accounting of status and activities work may cause some new works for the developer, as reporting and recording status of proposed changes or register configuration documentation. This loss of time can be amortized in the production phase, because the historical CIs' reports of the can be used for the documentation of the product. This way the documentation work is done before the product is finished.

Configuration status accounting and configuration identification provides the framework and knowledge base of what has been going on during a project helping staff changes.

1.3.4 Configuration Auditing

Configuration audits verify that the approved changes have been implemented. Performance data and physical configuration are compared to documentation. Configuration audits assure both manager and the customer that requirements are reflected in the product and that the product responds to these requirements.
There are two types of configuration audits:

- Functional configuration audit (FCA)
- Physical configuration audit (PCA)

The FCA verifies the functional characteristics, while the PCA verifies the form and fit of the production. The result of the FCA is a formal acknowledgment that the product performs in compliance with requirements and specifications. This performance is established by review of the test cases and test results and by tracing the documentation back to initial requirements. The PCA results in a formal acknowledgment that the product matches the physical description in its documentation.
Established the target for the audit, the first action is to perform the audit team. A chairperson is needed. Other members of the team are selected on the basis of their special interests or knowledge. Next, the audit begins with a meeting of all players of both parties; this meeting is conducted by the chairperson. After its conduct the results are published.

Configuration audits assure that the product responds to the requirements, this is a hallmark for the management and for the customer. The conduction of audits during the development assures the consistency of the work, improving the stability of the baseline. There are some added costs associated with the time spent doing the audit meeting.

1.4 SCM in the product development life cycle

SCM is an integral task, beginning in the early life cycle. Required from the beginning of the system exploration phase, the project software configuration management system must be available for the remainder of the project.
There are many possible software development models which can be used by a company. A waterfall model composed of 5 principal phases (requirements, design, coding, production and maintenance) is considered to ease the research of the costs and benefits of SCM. These costs and benefits can be easily translated in another product development philosophy, considering that these five phases can be found in small scale also in iterative model, like Agile software development.

Figure 1.3 illustrates the ”when” of SCM on our full product development life cycle.

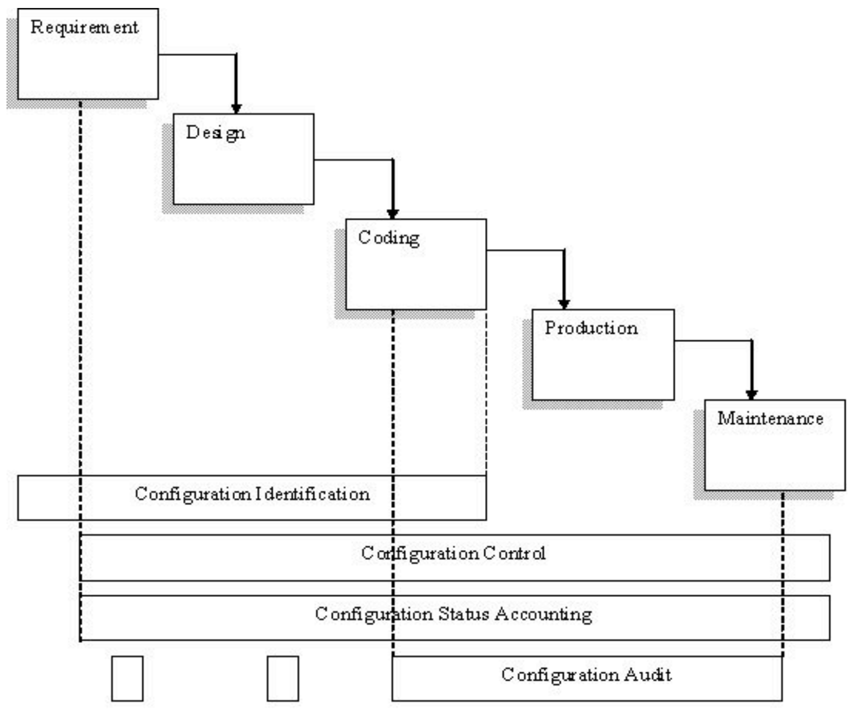

Figure 1.3: SCM in the product development life cycle

Configuration identification starts with the beginning of the project. Configuration control and status accounting are present in all the five phases of the product development life cycle. Audit are done at the end of each phase and are relevantly present during the final phase of the production.
Configuration Management helps individuals and interactions over processes and tools, to create documentation, to have more customer collaboration (CCB) and to have changes supporting the organization and track of changes. Configuration Management tool helps the continuous integration minimizing the merge conflicts. The followings paragraphs show the different benefits and costs associated with the 5 phases of the product development life cycle.

1.4.1 Requirements

The aim of the first phase of a project is to create a description of what a system should do; describing system features, properties that the system must have and limiting the development in some way, such as, defining an operating system the system must run on, or defining which programming language must be used to implement the system.
At the same time the configuration manager starts the planning process. One of the first activities is the configuration identification to divide the project in a top down manner defining configuration items. To have configuration identification on this phase helps to coordinate the workgroup with a proper division of the project in different CIs.
The requirements can be adjusted by the feedback of the change process; the CCB adds the possibility to interact directly with the customer assuring, through the CCB meetings, that the product will follow his requires.
These activities add costs for the new tasks introduced by SCM but this added work brings more benefits especially in the late phases of the product development life cycle.

1.4.2 Design

In the design phase the architecture is established. This phase starts with the requirement document delivered by the requirement phase and maps the requirements into an architecture. The architecture defines the components,

their interfaces and behaviors.
The design may include the use of existing components, this way SCM can help the developers with software reuse support as version management tool. In this phase there is still the configuration identification work, which, simultaneously with the design, defines the configuration items which compose the product. As in the requirement phase, the feedback from the CCB and the configuration status accounting add the possibility of changing the design documentation as the project goes on. Designers' workload is facilitated by the configuration identification by the formal definition of the CIs.

1.4.3 Coding

In this and in the next two phases, there are the most of the benefits derived from SCM discipline. The version control tool helps to handle variants of the product and to reuse software, eliminating repetitive developing efforts. When supported, the SCM tool helps developers' parallel work with automatic merge. Depending on the tool features the building step is eased and developers are sure to work on a correct build program, with no file missing or wrong version files link.
SCM discipline provides communication between developers. This benefit provides more knowledge in the development group, removing classical coordination errors like the double maintenance problem, the simultaneous update and the shared data problem. These benefits bring to a reduction of errors during the development as the development time decreases.
There is an added work for CCB members; programmers are not as free as before because they have to wait for the authorization before implementing the change and the change process can be complicated. There is a loss of time and money waiting because of the authorization.

1.4.4 Production

During the production there is the validation phase, the last build, the issue of the release and the documentation production. All the SCM activities provide the ability to trace the product through its development and growth, so that the problems may be traced to the origin. This is an advantage for the testing

people, who have to conduct the validation of the final product.
Automatic build procedures, when supported by the SCM tool, can save time and money; in this phase there are less benefits than in the previews one, because there is not the necessity to do frequent and quick build as development build[5]. The main advantage is to have the security of including in the build all the right version files, avoiding a loss of time for developers.
Quality assurance is facilitated with configuration identification and configuration control. The time spent auditing, is repaid with the assurance that the customer gets what he paid for improving the relationship with him.
During the early phase of the project there is an added cost associated with configuration status accounting and configuration identification documenting and ordering all the different tasks in the project. This is a great effort during the production of the documentation.

1.4.5 Maintenance

The most of the benefits of SCM are during this phase. One of the main aims of SCM is to provide the traceability among versions, releases, and product families. The value of this benefit is enormous especially during the maintenance phase of the product. A classical maintenance setback happens when a problem arises in one release or product family that impacts other client releases and products. Making a change and promoting it through the entire product software base is an incredible cost savings in time, money, and client satisfaction. A lack of linkage between project events contributes to project failures, where solving a problem either aggravates a problem in another area or fails to fix the same problem elsewhere. A traceability thread allows management to examine the chain of events that caused a project to encounter difficulty as an integral process within the auditing capability of SCM. Traceability and the improved organization given by SCM discipline are a support for the maintainers' work, with a reduction of bug fix time.

[5]A development build is a build done by developer that is never released to QA

1.5 People involved

In this section SCM is analyzed from the point of view of company roles. SCM is a discipline which is present in almost every phases of the software production; for this reason there are different people involved in it, everyone has a different goal. The SCM person this paper looks at are:

- Senior management
- Project management
- Configuration manager
- Designers
- Administrator and Technician
- Developers
- Production personnel

These categories are described in the following sections.

1.5.1 Senior management

Senior management means a distinct level of decision-makers, that must approve or disapprove SCM investment decisions. They are interested in what goes on in the development of their products, especially in the quality of the product and the service to the customer. They take care about the financial benefits and costs.
Using SCM provides competitive advantage as to be able to bring out the product earlier or to decrease the time required to respond to the end-user requests and inquiries. All these benefits are difficultly economically quantifiable, but they can be a hi-discriminating for the decision implementing SCM in the company. There are many direct costs to be considered by this company class, as the tool licenses costs, the training costs for all the company staff, the maintenance costs.

1.5.2 Project management

The activities of the configuration manager are concerned with successfully achieving a set of goals. It includes planning, scheduling and maintaining progress of the activities that comprise the project. He has to maintain the risk of failure as low as necessary over the lifetime of the project. SCM help him to have a clear view on the project and to keep the control of it. CCB and the impact analysis assure to finish on time and in budget.

1.5.3 Configuration manager

The configuration manager is a new character in the company; he is involved in every activity of a configuration management plan.
He is the responsible for development and implementation of a logical and a well defined configuration identification process for decomposition, naming and numbering conventions and baseline managing. The configuration manager is an advisor in the process of CI selection and decomposition; he is responsible for producing the listing of decomposition.
He is usually the secretary of the CCB; he prepares and coordinates the meeting and he is a member of the audit team.

1.5.4 Designers

This subject has already been analyzed in paragraph 1.4.2, where the design phase is described.

1.5.5 Administrator and technician

The administrator of the system and the technician are involved with new work associated with the use of the SCM tool. Depending on its complexity they will spend their time maintaining the system, creating new version views and managing and maintaining the repository. The complexity of these works depends on the dimension of the projects and on the size of the team working at the same project.

1.5.6 Developers

The job of the developers is the most influenced in the company by the introduction of SCM in software production, because with SCM discipline, their work changes. For this reason they could be reluctant to introduce this new discipline in their work life. If they have never used SCM before, they need training, after having learned the discipline their fear of new procedures should be facilitated.
Developers have several tools to help them to create the program. Version control and refactoring tools decrease the time of development and the number of bugs. Working with a baseline and with the own workspace increase the stability of the project. A developer carries out their modifications undisturbed by the activities of other developers.
Depending on the implemented philosophy of the tool, the check-out may also place a reservation (a lock) into the version group, giving the developer exclusive rights to a later check-in, or can implement an optimistic and merging-based like CVS philosophy. In the first case developers, wishing to modify the same original version, can only create a variant of this version and then have to merge their changes back into the original line of development. In every implementation of a versioning tool, SCM adds the possibility for the developers to be able to have parallel work and to develop in an isolated workspace. This two benefits permit to increase the productivity of the staff (parallel work) and of the individuals (working in isolation in the own workspace).
Developers join the benefits of SCM but on the other hand there is an improvement of their work, doing new tasks like writing configuration status accounting reports and participating at CCB meeting as members or specialists.

1.5.7 Production personnel

In this category there are all the people involved in the production of the final product: release manager, builder, system tester and quality assurance people.
Release management deals with the formal aspects of releasing to the customer and the more informal aspects of releasing to the project. For a customer release he needs to carry out a configuration audit before the actual release. This means also to verify if the produced package (application, documentation, help

files, etc) is actually installable. This procedure adds more work for the production, however it is a hallmark because everything follows the requirements. Release manager takes advantage of the traceability and the documentation achieved by SCM; his work is facilitated by version control tool.
Version management and build management tool help builder decreasing time to create a running system and are a hallmark, because it makes sure that every file is included in the build.
The traceability brought by configuration management activities helps the production personnel work. Audits guarantee that everything follows the requirements and verify that the approved changes have been implemented. QA people are able to test accurate and consistent builds.

1.6 Conclusion

In these three studies it is evident that there are a multitude of parameters and aspects concerning the introduction of SCM discipline in the software production. The previews study is summarized in Figure 1.4

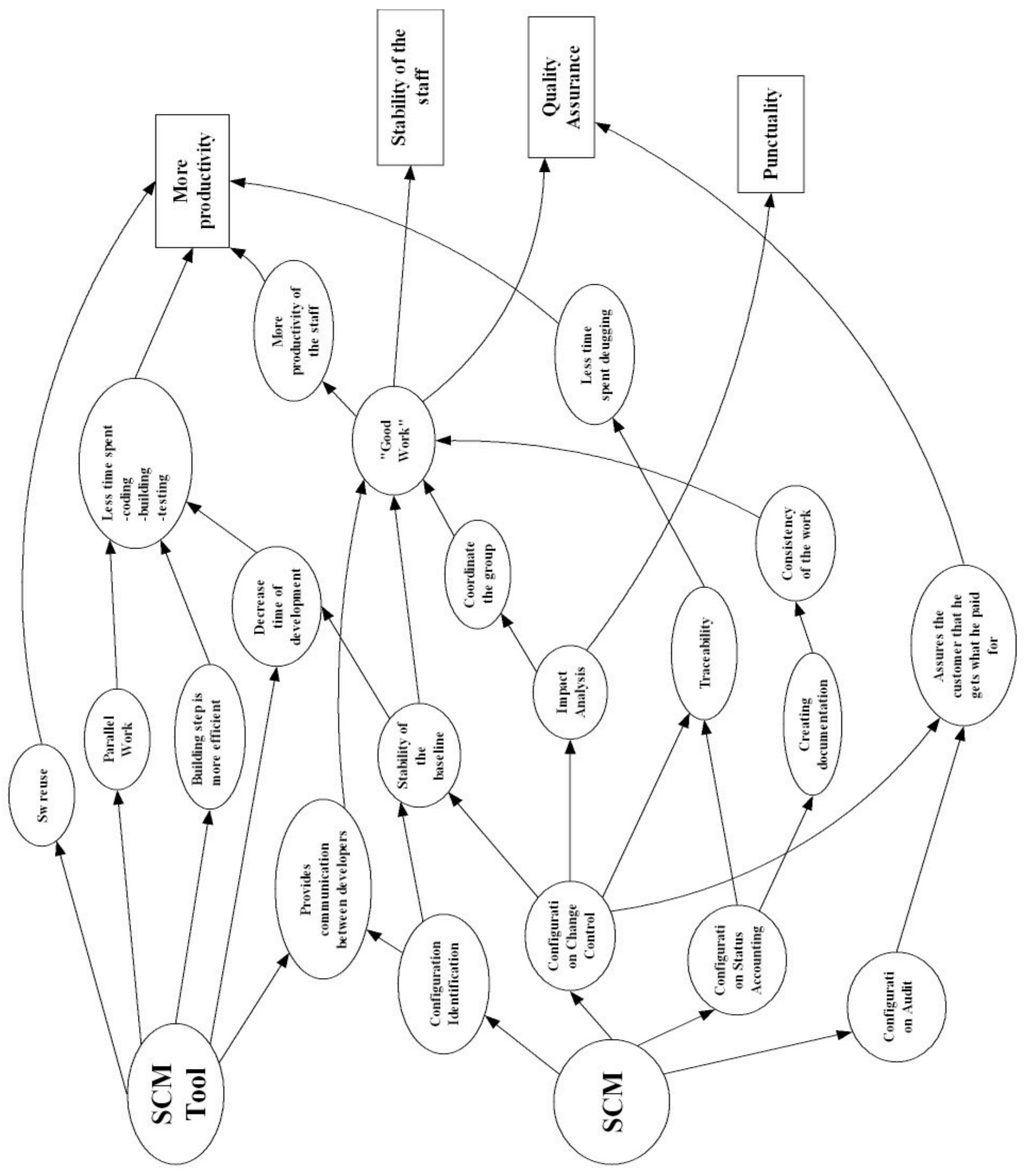
More productivity
Stability of the staff
Quality Assurance
Punctuality
More productivity of the staff
Less time spent deugging
"Good Work"
Less time spent -coding -building -testing
Consistency of the work
Coordinate the group
Decrease time of development
Traceability
Assures the customer that he gets what he paid for
Impact Analysis
Creating documentation
Parallel Work
Building step is more efficient
Stability of the baseline
Sw reuse
Provides communication between developers
Configuration Identification
Configurati on Change Control
Configurati on Status Accounting
Configurati on Audit
SCM Tool
SCM

Figure 1.4: SCM benefits

It is difficult to imagine performing any kind of configuration management without using an appropriate SCM tool, for this reason the graph is organized with two roots, which represent the two possible sources of benefits and costs:

- SCM discipline
- SCM tool

SCM discipline addresses more benefits to the top part of the company hierarchy. It addresses organization of the workgroup, assuring more visibility of the project and facilitating the management. Following the four activities of SCM selected software work products are identified, controlled and available; changes to identified software work products are controlled; affected groups and individuals are informed about the status and content of software baseline.
SCM tool benefits are associated with the developers' work. The programmers have parallel work support and can work in a stable environment; SCM tool provides communications between developers, increasing knowledge of the entire project. Automatic merge tool speeds the production of new code.
The related costs depend on the tool GUI[6] because it is used throughout the project and not only by developers, an intuitive, and easy-to-use graphical user interface is very important. This aspect is significant also to learn the costs, a tool with a high learning curve adds many costs for its introduction in the software development process. Maintenance costs also weight on the budget of the company.

The different benefits of SCM are divided in to four categories

1. Stability of the staff
2. More productivity
3. Quality assurance
4. Punctuality

[6]GUI, Graphical User Interface

Affected groups and individuals are informed of the status and content of the software baseline mostly by the status accounting activities. SCM eliminates confusion (chaos, double maintenance, shared data problem, simultaneous update), providing coordination and communication in to the group. All this aspects provide stability to the group.
There are many aspects concerning the increased productivity of the company. Selected work products are identified, controlled and available providing maintenance support. SCM helps to manage versions and variants of software, helping software reuse.
Quality assurance is mostly guaranteed with change process and with configuration audit. With the CCB changes to identified software work products are controlled and audits assure that only the approved changes have been implemented.
With SCM discipline all the production activities are planned, providing visibility for the project manager assuring that the project will respect the deadline.
The 4 benefits categories are examined in paragraph 3.2.
SCM offers tools which help the developers, and adds order and discipline to the group, it is a discipline that can be applied on different levels and different tools can be used; for this reason benefits are not univocally defined and many parameters are uncertain. The main idea of this work is to create a model which can calculate the ROI with objective and quantifiable data and a guide of the costs and benefits customizable from the different situation. For this reason three different degrees of measurability are defined.

1.6.1 Measurability

Two aspects of the parameter are considered to define the degree of measurability: the objectivity of the benefit/cost and how this benefit/cost is quantifiable. Following this criteria the parameters are divided in four regions within a matrix (Figure 1.5).

As shown in Figure 1.5, those three categories of parameters are:

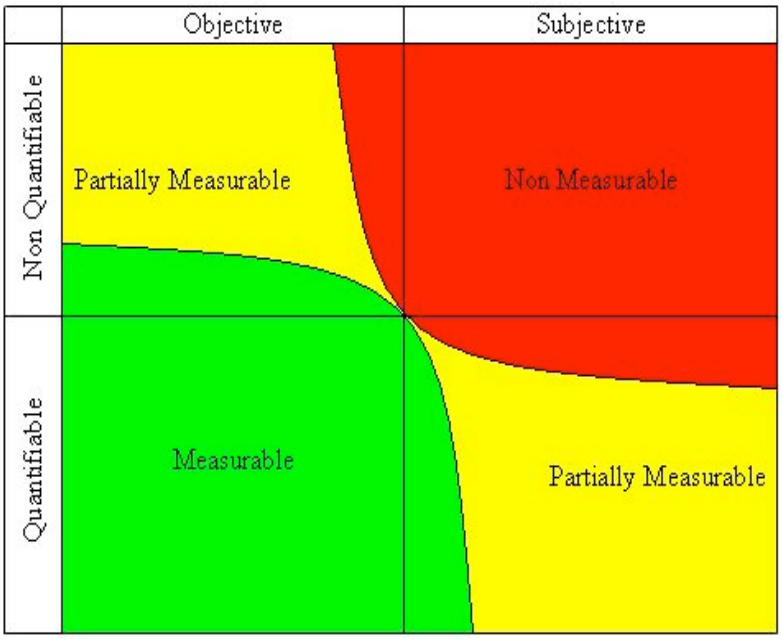

Figure 1.5: Measurability

- *Measurable* - the quantifiable data that can be objectively measured. Also those parameters that are near the border of the rectangle enter this category. This is the case of the cost associated with the added work for the configuration manager that is a parameter which depends on subjective aspects like the dimension of the project and the salary of the configuration manager. The gains associated with these parameters are constant with the context where SCM is applied. These data will be the inputs of the ROI formula.
- *Partially measurable* - these data are difficultly measurable, because they depend on the context where they are measured, or they are difficultly economically quantifiable. In this case the earning of SCM implementa-

tion is dependant on the contest of the company where SCM is applied; the gain is too variable to be considered in a generic formula.

- *Not measurable* - In this area all the parameters that are considered too much dependant on the context and intrinsically economically unquantifiable can be found.

This way the parameters can be divided in: those that can directly enter the formula for the ROI evaluation; the parameters that have to be elaborated because of their dependence on the context where they are applied; and those that can be relevant for the decision of using SCM even if they can not be used in a formula because they are intrinsically economically not quantifiable.

Chapter 2

The Model

In this chapter the results of the analysis of the SCM economical impact are presented. The costs and benefits discovered during the three previews studies are summarized in a matrix. Then a detailed explanation of its entries follow. The chapter is concluded with the ROI evaluation.

2.1 The model

SCM is a discipline intrinsically dependant on the context of application. The possible benefits and the respective earning factor could vary depending on the different case of application. For this reason it is impossible to create a complete formula applicable to all the possible cases.
SCM is a subjective discipline and for this reason this work focusses on the creation of a subjective ROI model which value has to be calculated by the company itself, knowing the context where SCM is applied and how the discipline is implemented. It has to be the most complete model including all the aspects of SCM discipline, not only the most known and quantifiable.
The model considers most of the SCM aspects. Considering all of them, this work fits perfect for those companies which introduce for the first time SCM discipline. The model can be also used considering a particular set of parameters, permitting to use it also from a partial to a full SCM implementation.
All the possible costs and benefits related to SCM are included in a matrix where they are divided, according to the classification described at the end of

the preview chapter. Both costs and benefits are divided in three degrees of measurability[1]. The matrix entries are described in detail in paragraph 2.3 and, when reasonable, it is indicated the possible metrics to measure them.
As it can be seen in the matrix in next paragraph, the parameters have an opposite distribution in the matrix; the most part of the costs are perfectly quantifiable and there are a lot of benefits that are partially or not measurable. This aspect is a hallmark for the person who has to decide about SCM introduction, because the model assures to consider even the worst case.
Considering the first column parameters it is shown a possible way to evaluate when the return on investment is achieved. This ROI evaluation is based on the estimation of the time that has to elapse in order that total benefits equals total costs. It does not consider the discount rate because, according to the previous studies [5] and [1], the return on the investment should come in few months. For this reason it is plausible that the contribution of the investment discount rate is irrelevant for the ROI evaluation.
According to the purpose to create an elastic model, this study includes a method to use it to facilitate the comprehension of the parameters involved in the goals of a SCM system; this part is described in the following chapter.

[1]Measurable, partially measurable and not measurable.

2.2 The matrix

	Measurable	**Partially Measurable**	**Not Measurable**
c o s t s	• Tool costs • Training costs • Added work associated with new SCM tasks for the: -○ Configuration manager -○ Admin and Technician -○ Developer	• Change process may be more complicated(loss of time and money waiting for authorization, average CCB_{time} • Little loss of time for doing the status accounting reports(depending on the tool and the level of automation)	• Fear of new procedures
b e n e f i t s	• Decrease of the externally reported defects (defect report arrival rate) • Less time per bug fix/ Ability to trace the original product through its development • Save time with automated software builds (building time) • Manage version, parallel work, automatic merge • Traceability addresses less time for V&V and testing	• Decrease of number of staff changes / helps to integrate new employers (less cost for teaching to a new employer) • Allow us to handle very complex activities (handle variant of a product) • Reusing existing code and reducing repetitive development efforts • Gain factor fixing bugs in different variant of a program (number of variant of a program) • Help the maintenance • Changes are planned, their impact is assessed (finishing the project earlier, Cost/day) • Reducing the number of errors (given from double maintenance, shared data problem, simultaneous update, working in chaos)	• Employers are happier • Provides for communications and coordination in the group • Pleasure of working in stability with a baseline and the own workspace • Home working and distributed developing • Be able to bring out the product earlier (before other companies) • Decrease the time required to respond to end-user requests and inquiries • Assures the customer that he gets what he paid for • Audits at the end of each phase assures the consistency of the work • Provides visibility of the project (good for project manager) • Achieves a sense of organization and control instead chaos
	Quantitative	<——————————————->	**Qualitative**

Table 2.1: The matrix

Depending on the level of SCM that will be implemented and the tool that will be used, the manager checks in the three columns the different benefits and costs, that he could have introducing SCM, helped by a list with the descriptions of all the entries of the matrix. For example he could be interested on the saving time using an automated software builds; in this case he fills up a form where there are the main parameters associated with the benefits using an automated builds tool, like the average personnel cost, the number of developers and builder people, the number of modules, the average time spent building.
This is the procedure for a measurable data; this kind of measure goes in input to the ROI formula. For the parameters in the partially measurable column the manager has to consider how the SCM rules are applied in the company and what is the contest of application. For example, looking for the increased costs for the change process. The time spent waiting for a decision of the CCB is strictly dependant on the organization of the CCB and the availability of the members. For what concerns the not measurable parameters, in this case it is not possible to have a valid evaluation independently from the observation. There are many benefits that are extremely subjective, in this case the manager could take a decision considering the costs/benefits that he has in his particular situation. If the manager knows that in his development team there is a lack of method, the introduction of SCM could be a very important benefit to achieve organization and control on the group and to provide for communications and coordination.

2.3 Parameters of the model

2.3.1 Costs

The different costs are mostly measurable and concrete parameters of the model. This is an important feature, because the most part of the costs are objectively considered. This way it is possible to consider the worst case of SCM introduction (the most part of costs are perfectly measurable and there are a lot of benefits that are partially or not measurable). This is a hallmark for who has to pay.

To be able to apply SCM discipline and install SCM tool, it is necessary to have a particular hardware, as a network with a server. In this study it has been supposed that these hardware units are usually present in most of the companies of a certain dimension, even if they are not using SCM. For this reason the contribution of hardware costs are not considered in the total investment concerning the SCM introduction.

2.3.1.1 Tool costs

The first costs that has to be considered introducing a SCM tool are the licenses and maintenance costs. They strictly depend on the tool used for the unitary costs and the number of licenses. An open source tool like CVS, which is free, or an expensive and complete tool like ClearCase which adds considerable costs for the maintenance of the system can be used. This choice is taken considering the particular necessity of the company. For this field there are a lot of vendors model that support the user to calculate the cost of the tool licenses.
Another variable is the number of users. Usually the system allows sharing licenses, respecting the necessity of the developer's staff and knowing the number of programmers it is possible to calculate the necessary number of licenses. The maintenance cost could be a percentage of the licensing price or could be variable depending on the contract with the tool vendor.

- Type of SCM tool and unitary cost per license
- Number of users

2.3.1.2 Training costs

The training costs for the system administrator and the developers represent a large part of the total costs. The tool by itself can not be expected to improve the product quality, but if accompanied by a formal change management process improvements might be expected. Unfortunately this process is not free. It depends mostly on the learning curve of the tool and on how the project is large and complicated (number of modules, number of people working in parallel, number of versions and variants of the program, etc). For example,

ClearCase has a much larger command set than CVS, and therefore it has a much higher learning curve; in this case the teaching cost to use ClearCase are more expensive than VSS.
In addition to the tool training costs there are also costs associated with the training of the group to the new SCM procedures. The configuration manager has to learn about all the SCM activities, in particular how to conduct a CCB meeting, how to conduct audits and how to do configuration management. The developer has to learn how to conduct audit, how to conduct status accounting and how behave when there is a change request.
Associated to those costs, it is necessary to consider also the annual costs for training the new employers for the SCM tool.

- Learning curve of the SCM tool
- Type of project
- Annual changes in the staff

2.3.1.3 Added work associated with new SCM tasks

Configuration manager The configuration manager is a partially new figure in a company where SCM is introduced, because every company has to manage the changes and the versions of a program, the more formal activities require a new character who has the responsibility of implementing SCM in the best way. He is the responsible for development and implementation of a logical and a well defined configuration identification process for decomposition, naming and numbering conventions and baseline managing. The configuration manager is an advisor in the process of CI selection and decomposition; he is responsible for the production of the list of decomposition. He is usually the secretary of the CCB; he prepares and coordinates the meeting. He is also a member of the audit team.

Administrator and technician Sometimes the tool requires dead full-time administration. These figures are responsible for the correct working of the SCM tool; for example they are responsible to maintain the system (the SCM tool), to create new views, to manage and maintain the repository. This fact depends on the dimension of the project and on the number of views (number of developers, number of versions and modules of the program and the number of the company products) and on the type of SCM tool used.

Developer The work of the developer is facilitated by the SCM tool, at the same time he has also some new tasks. The main activities he has to perform are the status accounting tasks (comments and status accounting reports) and the CCB tasks (doing impact analysis).

2.3.1.4 Change process may be more complicated (cost derived by loss of time)

The essential issue of the Change Process is the communication, it assures that the changes will be implemented to the real product after an authorized decision, no matter how complicated the process becomes. For this reason change process may add some costs. There would be a loss of time and money waiting for the authorization of the CCB chairperson. This parameter is quite difficult to estimate because of the aleatory nature of the duration of a decision and the variable number of change requests in a project.

2.3.1.5 Little loss of time doing the status accounting reports

The mission of Status Accounting is to support traceability. Beginning with each defined baseline, the target of status accounting is to be able to show the product's progress to the next baseline by tracing the changes. Developers lose little time writing comments and reports during the development of the program. This is a difficult parameter to measure, but it's reasonable to consider that this cost is not as relevant as the others. Time spent doing status accounting depends also on the assistance given from the SCM tool.

2.3.1.6 Fear of the new procedures

This is a not quantifiable cost. It is an impediment to the introduction of SCM discipline, even if training should help to decrease the fear of the new procedures.

2.3.2 Benefits

There are many aspects concerning the benefits of using SCM discipline in software development. These aspects are described in the following sections using the division (measurable, partially measurable and not measurable) as showed in the matrix (table 2.1).

2.3.3 Measurable benefits

In this category there are the more objective benefits, which can be associated to a constant economical gain factor. These parameters go in input to the ROI formula, to estimate how long does it takes to have the return on investment.

2.3.3.1 Decrease of the externally reported defects

This is the main parameter for the ROI estimation. A decrease in the defect report arrival rate provides less time to develop and maintain the product, which means that there is a saving time on the project and a reduction on the salary paid. Developers are less disturbed by the continuous requests for bug fixing, thus they can work in a better way and more efficiently.
Larsen and Roald's study points out that using SCM in the production of software product brings a decrease of the externally reported defects of 36% [5]. This is achieved also because there is an increment of the internally reported defects. SCM helps in to find and to fix the bugs before the production and the last release of the product.
This parameter is used to evaluate the return on investment, multiplying it with the average cost to correct one bug obtaining the total save.

- Defect report arrival rate (both external and internal)

2.3.3.2 Decrease of bug-fix time. Ability to trace the original product through its development

The documentation achieved by configuration status accounting, the version control tool and the configuration change control (with the CCB) bring to traceability. This property renders in less time per bug fix, easing and speeding the maintenance phase. The Norwegian study of Larsen and Roald estimates that the average time spent to process and fix urgent defects goes down by 6% [5]. The total saving time can be quantified knowing the average time spent per bug fix and the number of reported defects.

- Bug fix time
- Average fixing time

2.3.3.3 Save time with automated software builds

This parameter strictly depends on the utilized tool and on the dimension of the project. An automated build procedure can save time and money. The decrease of workload associated with the build task is estimated from the Merant study between 12% and 40% depending on the type of build[2]

- Development build time (reproducible build, incremental build, reproducible build)
- Release build time

2.3.3.4 Manage version, parallel work and automatic merge

This feature can help the programmers in their work, especially in big groups of development. The degree of help depends on the characteristic of the used tool: how the check-in check-out process is managed, if an automatic merge tool is supported, if there is the possibility to lock the files, what are the methods of programming, etc.
With an automatic merge tool it is possible to save much time especially if there are many changes in the program development and if there is high parallel work.
Another characteristic of the tool that is necessary to consider is how it manages the branches. The key advantage of branching is parallel development and the resulting efficiencies and responsiveness. Using a version management

[2]The possible types of build are divided in two main categories: development build and release build. They are described in paragraph 3.2.3.

tool there is a time save in creating, tracking and recreating versions.
The degree of parallel work that it is possible to handle assures more productivity, because there are more developers' group working at the same time and because there is an increase of the productivity for human resources.

- Number of people working at the same project or module
- Type of tool

2.3.3.5 Traceability addresses less time for V&V and testing

All four activities of SCM add benefits to the V&V phase. Configuration identification and change control defines the original approved configuration and the approved changes to this baseline, configuration audits assure that the standards have been met and configuration status accounting permit logging and tracking of incident reports. The documentation produced during the development achieves traceability and with the help of the version control tool the testers' work is easier than before.

- Time spent testing and validating the program

2.3.4 Partially measurable benefits

2.3.4.1 Decrease of number of staff changes and helps to integrate new employers

In every companies, people get promoted, take other jobs, and leave. When this happens in the middle of a development project, not just the technology knowledge goes out the door. The long-learned knowledge of how things are done is also gone. When a replacement person is brought on board, they may know the technology, but without a documented SCM process, they will have no real idea how to do product development. SCM provides the framework and knowledge base of what has been going on before in the project. A new staff member has one place to go to learn how the organization development process goes on.
A documented SCM process facilitates the insertion of a new employer in the company. This should get proportional benefit with the average cost for

introducing a new employer.
Another benefit that is achieved by SCM is the decrease of the number of staff changes. This is difficultly quantifiable for a ROI calculation because there are many different and independent reasons for staff changes. But it should be logic thinking that if employers are happier they would not leave they job.

- Number of annual staff changes
- Average cost for introducing a new employer in the company (cost for training and initial scant productivity)

2.3.4.2 Allow us to handle very complex activities like handling variant of a product / Gain factor fixing bugs in different variant of a program

The purpose of a professional software development project is to build a family of computer programs. SCM controls all the members of a program family, keeping multiple versions identical in ways they are supposed to be identical and different in ways they are supposed to be different. All the efforts to manage many different variants of a program are minimized by the tool and the knowledge achieved by all the documentation. If a bug is found in a module which is present in some variants of a program family it will be fixed once for all the different variant. More variants exist and more benefits the company obtain.

- Number of variant of a program

2.3.4.3 Reusing existing code and reducing repetitive development efforts

This benefit is a direct consequence of the possibility, achieved by SCM discipline and SCM tool, managing different version of a program. It is a difficultly quantifiable benefit because it is impossible to know how much code will be reused in a project.

2.3.4.4 Help the maintenance

All the SCM activities add order and produce documentation to a project, achieving traceability. This property helps the developers to find the problems earlier and decrease the bugs-fix time.
The version control tool helps the developers to manage the different versions of the program; the possibility to store, recreate and register the historical development of an item (document or source code) is a fundamental characteristic of a version control system. The tool keeps track of relevant information about the different versions. These characteristics, with the ordered environment assured by SCM, help maintainers to assist the customer and to maintain the program.

- Time spent debugging externally reported defects
- Time spent maintaining system

2.3.4.5 Changes are planned, their impact is assessed

The plan with configuration identification and the impact analysis of the CCB permit that all the activities are done with the project manager overview. Changes are planned and their impact is assessed. This is a guarantee of finishing the project on time.

- Having the product before other companies
- Save development money (Cost/day)
- No forfeit for belated end project

2.3.4.6 Reducing the number of errors (given from double maintenance, shared data problem, simultaneous update, working in chaos)

On any team project, a certain degree of confusion is inevitable. The goal of configuration management is to minimize the confusion so that the productivity increases. All the classical problems of working in group (like double maintenance, shared data problem, simultaneous update) are a setback for the developers' work. Use of SCM increases productivity for human resources. This parameter depends also on the size of the developer's group and on how many programmers are working simultaneously on the same module.

- Degree of confusion (number of developers, how the project is divided)
- Number of classical CM problem

2.3.5 Not measurable benefits

2.3.5.1 Employers are happier

This is a really subjective and qualitative measure of the benefit of SCM. This entry of the model is an index for the productivity for human resources and for the rate of changes in the staff; if the personnel is happy working in its environment it won't change its job.

2.3.5.2 SCM provides for communications and coordination in the group

A better coordination of the group improves the productivity, minimizing the waiting time and the mistakes. Knowing how many people are involved in the production and how this people are organized permits to estimate the possible number of communication channels and the degree of assistance that SCM offers.

2.3.5.3 Pleasure of working in stability with a baseline and the own workspace

This is probably the aim of SCM. Babich says: "It is not possible to overemphasize the importance of stability to successful software development. Nothing is more frustrating to programmers than not being in control of changes affecting the program they are trying to get working" [3].
SCM activities permit to control all the changes in a project, for this reason developers can work in a safe environment, where only the approved changes are implemented. The items in the baseline are a stable base for development.

All programmers want the database to be stable when they want it to be stable and changeable when they want to make a change. This conflict is solved with the workspace, which permits to work in isolation allowing the developers to change the program any time. The benefits depend on the used tool, and on the dimension of the project (how many people are involved, in how many modules the project is divided).

2.3.5.4 Home working and distributed development

The tool can permit home working and distributed development. This is a not quantifiable benefit, but it could be a really important one; for examples the possibility to develop software in a country where the production costs are cheaper then in the country where the software company is based, permits higher earnings.

2.3.5.5 Be able to bring out the work earlier, before other companies

All the different benefits of using SCM bring a shorter time for developing a software product. This is a direct economic benefit saving time and money in the software production. A more subjective is achieved if a product comes sooner on the market. This benefit is not measurable because the earning of coming first in the market is not quantifiable; the company should became the market leader if it produce a program before the others; but it can go bankrupt if it comes late than the others.

2.3.5.6 Decrease the time required to respond to end-user requests and inquiries

This is a consequence of the better organization of the group and the decrease of the time per bug fix; it is a difficult quantifiable benefit, which improves the relationship with the customer. The pleasure of the customer is none economically quantifiable parameter.

2.3.5.7 SCM activities (in particular configuration audit and configuration control) assure the customer that he gets what he paid for

All the four SCM activities bring traceability and the consciousness of how the project is going on; the continuous interactions with the customer through the CCB permit to know and satisfy the customers requests. All these hallmarks add costs in the production.

2.3.5.8 Audits at the end of each phase of the development life cycle assure the consistency of the work

Configuration audits assure that developers are working on a stable and approved version of the product. This procedure adds some costs and in the successive phases of the software production there is a return on investment because of the improved productivity of the developers working in a stable environment.

2.3.5.9 Achieves a sense of organization and control instead chaos

The plan achieves a sense of organization and control instead chaos, helping developers to work in a ”good way”. This achieves more quality and productivity of the staff.

2.3.5.10 Provides visibility of the project

Having a plan and doing status accounting and reviews provides visibility of the project, helping the work of the project manager.

2.4 ROI evaluation

The return on investment (ROI) is a calculation used to determine whether a proposed investment is wise, and when the investor could be repaid. The ROI formula is supposed to calculate how long does it take for the company to have the return on an investment in SCM field.
This payback is evaluated making the intersection between the trend of costs and benefits in the time; this is calculated solving the system composed by the functions *TotCost* and *TotBenefit* in the months after SCM introduction.

$$\begin{cases} TotCost = IC + \sum(\frac{Cost}{Month})_i \\ TotBenefit = \sum(\frac{Benefit}{Month})_i \end{cases}$$

TotCost represents the total costs function of the investment, it is the sum of the initial investment (IC) as the cost for the first training period and the licenses of the tool, and the day to day costs for maintaining the SCM system.
TotBenefit is the sum of the earning associated with SCM. The function should have the trend showed in Figure 2.1

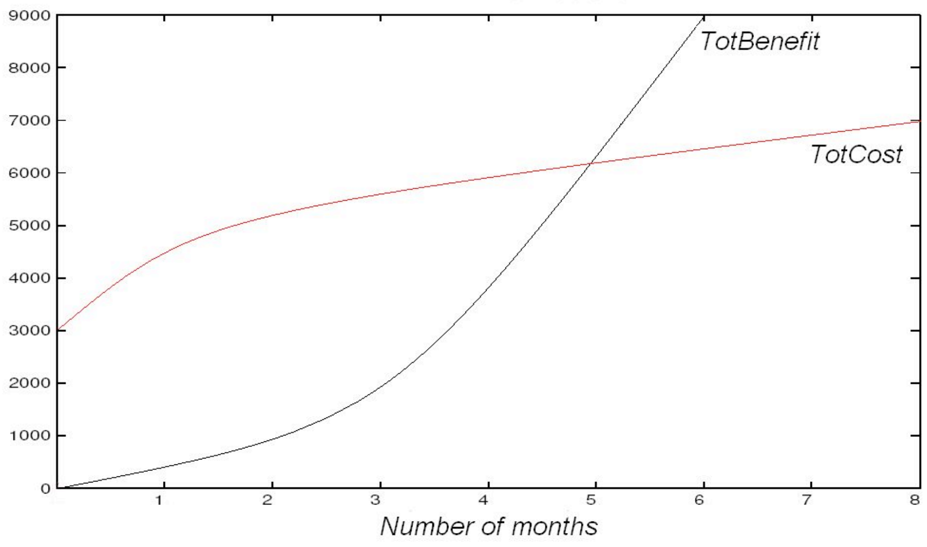

Figure 2.1: Cost and benefit trend

TotCost should start from an initial value, the initial investment, and then it rises with a decreasing growth rate to a linear trend, because in the first period there are added costs for the introduction of the system. *TotBenefit* starts from the origin and has an increasing growth rate as the staff people learn how to do the new tasks and use the new tool.
The inputs of those functions are the entries of the first column of the matrix. These parameters are used because they do not depend on the context of the company and on how SCM is implemented. This way it is possible to calculate the ROI in a general formula. With the help of the model is possible to customize this calculation with the entries of the other two columns.
According to the description of matrix parameters in the previous paragraph (par 2.3), total cost is divided into three parts: Tool costs, Training costs, Costs for added work associated with new SCM tasks. Total benefit is similarly divided into four parts: Benefits on bug fix, Save time with automated build, Manage versions Automatic merge Parallel work and save time on testing. The functions become:

$$TotCost = ToolCost + TrainingCosts + SCMTaskCosts$$

$$TotBenefit = BenefitOnBugFix + AutomatedBuilding + ManageVersion + SaveTestTime$$

As described in paragraph 2.3, all these seven entries of the formula depend on different aspects.

2.4.1 Costs

2.4.1.1 Tool costs, licenses and maintenance cost

These entries of the formula are perfectly quantifiable. The formula is composed of the total costs of the tool licenses and the cost of the vendor support.

$$ToolCost = NdevLicDshare + MantCost$$

Where:

N_{dev}	Number of tool users
Lic	Cost per license
D_{share}	Degree of share license
$MantCost$	Maintenance costs

2.4.1.2 Training costs

The training costs strictly depend on the utilized tool and on how SCM is implemented. The formula for its calculation is below presented.

$$TrainingCost = (C_{adm} + C_{teach})h_{adm} + (N_{dev}C_{dev} + C_{teach})h_{dev}$$

Where:

C_{adm}	Cost per hour of the administrator
C_{teach}	Cost per hour of the teacher
h_{adm}	Average hour to teach to the administrator
C_{dev}	Cost per hour of the developer
h_{dev}	Average hour to teach to developer

In these entries of the cost formula it could be also included the training costs for new employers. This is a recurrent cost because every company has changes in its personnel. In this case it is possible to quantify it, if the rate of staff changes and the average cost for learning SCM discipline and tool per person are known.

2.4.1.3 Cost for added work associated with new SCM tasks

The configuration manager spends most of his time doing configuration identification, preparing and assisting to the CCB meetings and conducting audits. The total time spent in configuration identification is divided in decomposition activities, baseline management and the creation of the baseline. The time spent doing the CCB secretary can be evaluated by the formula $N_{chan}T_{CCB}$. Where N_{chan} is the number of changes in the project per month and T_{CCB}

represents the average time per CCB decision. The time spent doing audit can be estimated considering the average time spent doing an audit (T_{audit}) and the average number of audit per month (N_{audit}) by the formula $N_{audit}T_{audit}$.
There are other costs associated with new SCM tasks, as shown in the second chapter. These costs can be quantified with the analysis of the particular SCM implementation case.
The administrator and the technician spend their time maintaining the system, creating new version views and managing and maintaining the repository, this work mostly depends on the complexity of the SCM tool.
The main developers' added workloads associated with SCM are status accounting, like writing comments and reports, and doing impact assessment for CCB meetings. This staff category does not add relevant costs associated with the new SCM tasks and it can be omitted in the formula.

2.4.2 Benefits

Benefits are calculated in hour units but they can be easily translated in money units multiplying the number of hours with the cost per hour associated with the worker category.

2.4.2.1 Benefit on bug fix

Using the number of internal and external defects per month and the average time per bug fix before the introduction of SCM and knowing also the decrease rate of the number of bug fix and of bug fix time permits to estimate the total save time associated with the bug fix.

$$BenefitOnBugFix = (\alpha_1\alpha_2N_{err}T_{BugFix})_{ext} + (\alpha_1\alpha_2N_{err}T_{BugFix})_{int}$$

Where:

N_{err}	Average number of defects per month
T_{bug}	Average time spent per bug fix
α_1	Rate of decrease of the number of bug fix
α_2	Rate of decrease of the time per bug fix

2.4.2.2 Save time with automated software builds

Considering the average time spent every month on the various type of build in a project ($Tbuild_i$) and the respective rate of workload decrease (β_i), it is possible to calculate the payback using an automated build with the formula:

$$AutomatedBuilding = \sum \beta_i Tbuild_i$$

2.4.2.3 Manage version, parallel work and automatic merge

Version management Considering the time spent per month creating, tracking and recreating versions ($Tver_i$) and the respective rate of workload decrease (γ_i), it is possible to calculate the payback using version management tool with the formula:

$$B_1 = \sum \gamma_i Tver_i$$

Automatic merge The improvement achieved by an automatic merge tool depends on the dimension of the diff[3] and from the average number of merge done in a month (number of version or number of branches). Knowing the average time spent merging files (T_{merge}) and the average number of merge per month (N_{merge}), the total benefit can be calculated.

$$B_2 = \gamma_4 T_{merge} N_{merge}$$

Where γ_4 is the rate of decrease workload.

[3]Diff, The difference between the program files

Parallel work SCM permits to have parallel work eliminating the classical coordination problems, like double maintenance, shared data problem, simultaneous update, the reduction of lose time instead this errors is represented by B_3 .

$$B_3 = N_{CoordProbl}T_{lose}$$

Where T_{lose} is the average wasted time lose with one of the classical problem of coordination and $N_{CoordProbl}$ is the average number of those errors per month.

2.4.2.4 Traceability addresses less time for V&V and testing

Knowing the average time per month for doing test (T_{test}) and the percentage of decrease of this time (δ) the last term of the formula can be calculated.

2.4.3 ROI Algorithm

During the first period of SCM introduction, the gain factor associated with the entries of the model can be variable; because, during this time, the staff has to learn the new processes and understand how the SCM tool works. For this reason the ROI should be calculated considering monthly costs and benefits; this calculation can be done with an iterative algorithm in which the changes of the gain factor during this period can be considered.

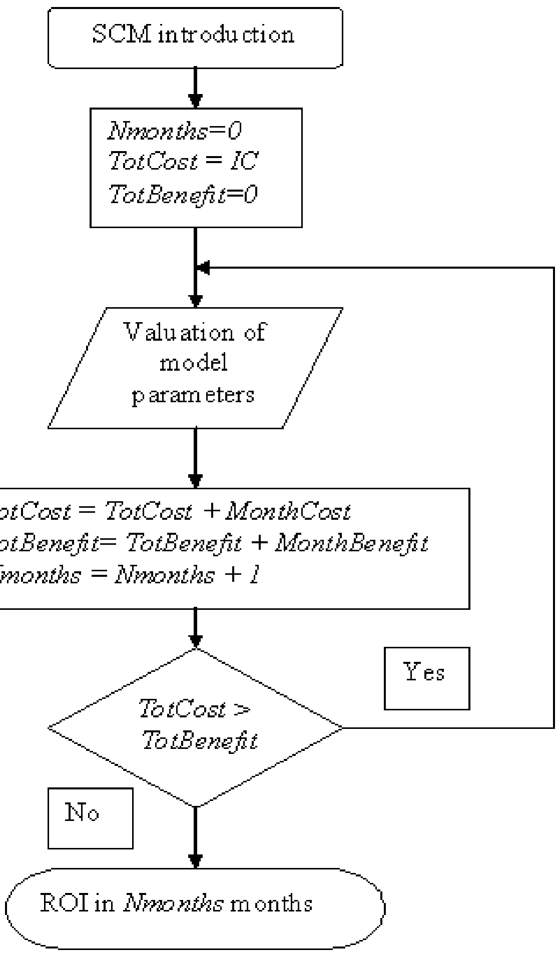

Figure 2.2: ROI Algorithm

Chapter 3

The method

SCM is a set of activities designed to control changes identifying the work products that are likely to change, establishing relationship among them, defining mechanism to manage different versions of these work products, controlling the changes imposed, and auditing and reporting the changes made. SCM is involved in most of the activities of a project.
There are different aspects about costs and benefits of SCM, often subjective, which have to be considered to calculate the earning applying this discipline in a corporation. The presence of this multitude of subjective aspects, associated with a SCM system, made necessary the elaboration of a method which can guide the user selecting and quantifying the related costs and benefits.
GQM (Goal Question Metrics) is used to show where the organization is along the path to gather the benefits of SCM.

3.1 GQM Analysis

The GQM method provides a measurement plan, that deals with the particular set of problems and the set of rules for obtained data interpretation. The interpretation makes clear if the project goals were reached [10].
The Goal Question Metric approach is based upon the assumption that, to make measures in a purposeful way, an organization must first specify the goals for itself and its projects. Then it must trace those goals to the data that are intended to define those goals operationally. Finally provide a framework for

interpreting the data with respect to the stated goals.
GQM defines a measurement model on three levels:

Conceptual level (*goal*): This level comprehend the major goals of the object of study.

Operational level (*question*): A set of questions is used to define models of the object of study and then focuses on that object to characterize the assessment or achievement of a specific goal.

Quantitative level (*metric*): A set of metrics, based on the models, is associated with every question in order to answer it in a measurable way.

A GQM model is a hierarchical structure (Figure 3.1: GQM Analysis).

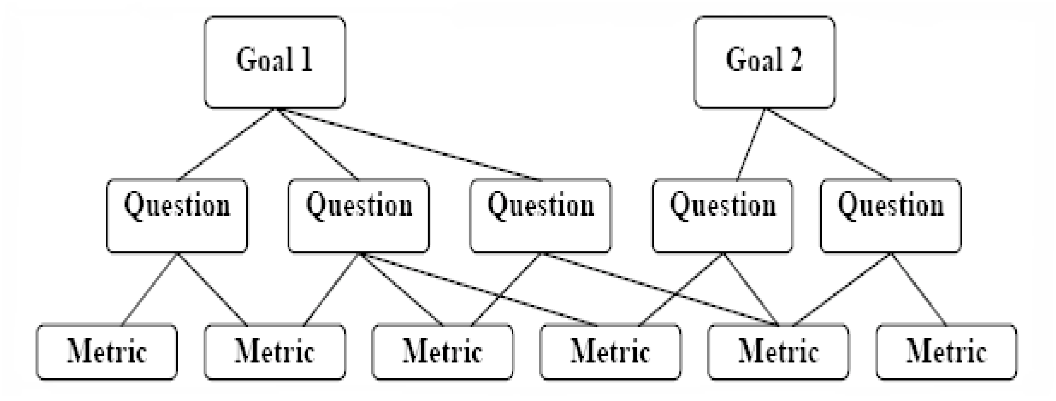

Figure 3.1: GQM Analysis

The main idea is that measurement activities should always be preceded identifying clear goals for them. To determine whether a particular goal has been reached, it is necessary to make questions, whose answers will show if the goals have been obtained. Then, from each question, the attributes that must be measured, in order to answer that question, are generated.

3.2 GQM application

3.2.1 SCM Goals

SCM purposes are divided in 4 goals

Stability of the staff: Affected groups and individuals are informed of the status and content of software baseline (Status Accounting). SCM eliminates confusion (chaos, double maintenance, shared data problem, simultaneous update, etc) providing coordination and communication in the group.

More productivity: Selected work products are identified, controlled and available providing maintenance support. SCM helps to manage versions and variants of a software, helping software reuse.

Quality assurance: Changes to identified software work products are controlled (Change Control).

Punctuality: SCM activities are planned, providing visibility to the project manager.

3.2.2 Goal 1: Stability of the Staff

One of the first benefits of SCM is to achieve a sense of order in the developers group, given improving the communication and the coordination between people working on the project.
Numerous research studies show that 15% to 20% of the day to day effort of software developers is spent dealing with configuration and change management activities on new development projects. On maintenance projects, this number increase to 25% to 40%. Applying SCM discipline and using an appropriate tool, it is possible to reduce time necessary to track, manage and communicate these configuration and content management activities.
SCM is necessary to coordinate big group of developers; more people are involved, and more time is spent communicating among staff members. The efforts spent in a team work grow up with the number of possible channels of communication, which increase as a quadratic function [2].

$$\frac{n(n-1)}{2}$$

On a three person team there are three possible communication channels but on a four person team, there are six possible channels.

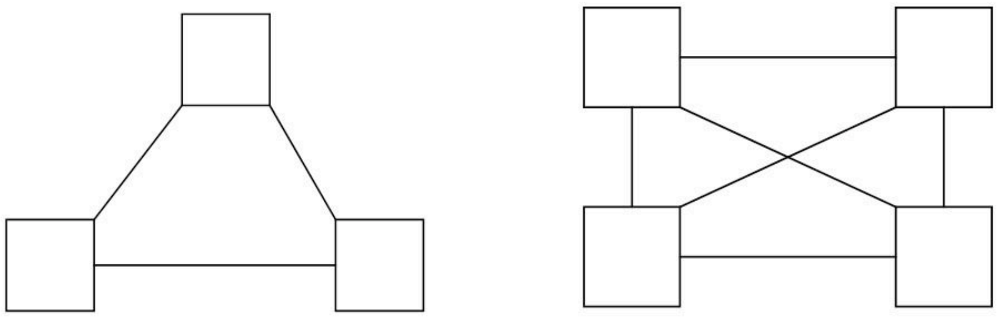

Figure 3.2: Channel of communication

Programmers lose productivity mostly for two reasons: they have to spent time coordinating their work with the one of the others; and even more important, they lose productivity because of mistakes that are made failing to coordinate. The typical problems that bring mistakes in work group are the double maintenance, the shared data problem, and the simultaneous update.
SCM overcomes these problems with the baseline and the concept of the workspace from the developers' point of view, and with the plan, the CCB and other management procedure from a managerial point of view.
There are different tools to help developers in their work. Those tools allow different degree of SCM assistance. From a simple version control tool to an automated merging and building tool, from parallel work to distributed development support.
All these instruments and procedures add pleasure to developers; in fact, nothing is more frustrating to programmers than not being in control of changes affecting the program they are trying to make it work. The use of a stable baseline and the own workspace, the security of not loosing time and programming in a planned and ordered project achieve a sense of confidence and tranquillity

in the developers' job.
There will be also some initial costs for the developers which have to learn the new procedure of work and, at the same time, developers see their freedom decreasing. This is a little price to enjoy the benefits of the discipline. The biggest cost is to learn how to use the new SCM tool, this cost depends on the learning curve of it.
Applying SCM to a corporation adds stability and knowledge of what is going on in the project and achieves another important benefit: decrease of number of staff changes and help to integrate new employers. In fact in every company, people get promoted, take other jobs, and leave. When this situation happens in the middle of a development project, not just the technology knowledge is lost. The long-learned knowledge of how things are done is also gone. In this case, when a replacement person is employed, without a documented SCM process, he will have no real idea how to do product development. SCM provides the framework and knowledge base of what has gone on before in the project. Using SCM a new staff member will easily understand the "how" of the organization's development process.

3.2.2.1 Questions and Metrics

Question	**Metrics**
How many people work in the company?	1. Number of employers
Who are the affected individuals and groups?	2. Affected individuals
Are developers satisfied of the working environment?	3. Degree of developer's satisfaction
How much work is spent dealing with configuration and change management activities?	4. Percentage of the developers' daily work spent dealing with configuration and change management activities
How many times classical CM problems occur?	5. Number of problem 6. Type of the problem
How many changes there are in the staff per year?	7. Number of staff changes 8. Cost for introducing a new employer in the company
What are the supported environments for developers?	9. Tool characteristic

3.2.2.2 Parameters from the model

- **Employers are happier (3)**: This is a really subjective and qualitative measure of the benefit of SCM; the entry of the model is an index for the rate of changes in the staff and for the productivity for human resources.

- **SCM provides for communications and coordination in the group (1, 2, 10, 11)**: A better coordination of the group brings a better productivity, minimizing the waiting time and the mistakes. Knowing how many people are involved in the production and how this people are organized, it is possible to know the possible number of communication channels and the degree of assistance that SCM offers.

- **Pleasure of working in stability with a baseline and own workspace (4, 5, 6, 9)**: SCM activities permit to control all the changes on a project, for this reason developers can work in a safe environment, where only the approved changes are implemented. The items in the baseline

are a stable base for development.
All the programmers want the database to be stable when they want it to be stable and changeable when they want to make a change. This conflict is solved with the workspace, which permits to work in isolation allowing the developers to change the program any time they want. The benefits depend on the used tool, and on the dimension of the project (how many people are involved, in how many modules the project is divided). One index of the necessity to use SCM could be the metrics 4, 5, 6.

- **Decrease of number of staff changes and help to integrate new employers (7, 8)**: SCM brings two important benefits: decrease of the number of staff changes and help the integration of new employers. The first entry is difficultly quantifiable for a ROI calculation because there are many different and independent reasons for staff changes. For what concerns the second entry a documented SCM process facilitates the insertion of a new employer in the company. It is reasonable to assume that there will be proportional benefits according to the average cost introducing a new employer.

3.2.3 Goal 2: More productivity

There are mostly three principal fields through SCM increases the productivity:

- Decreasing development time
- Helping and speeding the maintenance works
- Managing the different version and easing the software reuse

The first entries is achieved by decreasing the coding time through the SCM tools which permit, with automatic merge and branching tool, parallel work minimizing the risk of conflicts and also decreasing the testing time through traceability and documentation. With the help of an automatic building tool it is possible to get benefits decreasing the release time.
SCM increases productivity for human resources throughout the organization which builds and/or uses software applications and their associated contents

(there is an increase of the human resource usage from 22% to 56%) [5].
The second mode to increase the productivity is achieved with traceability, which helps the maintenance staff to find and fix the bugs.
The less time spent on maintenance effort (33%) mostly correspond to the fewer defects reports [5]. Usually if a defect is internally reported it takes less effort to fix it, then if it comes from the customer. Using SCM decreases the number of external defect reports, increasing the internal one; this way there is a decrease of the fixing time.
The third way to increase the productivity is achieved by SCM tool which permits to create and manage different variants of a program. SCM permits to manage multiple versions program so maintenance efforts are minimized and unnecessary duplication of work is eliminated.
There is another important benefit. Using SCM permits to decrease the "time-to-market" needed to develop a program. This is a difficult quantifiable benefit, although this is a really important one. Think about coming first on the market with a new product. Different articles estimate that improvement in a range of value between 5% [5] and 38% [1].

3.2.3.1 Questions and Metrics

Question	**Metrics**
How many developers are working in each project?	10. Number of developers
In how many modules a project is usually divided?	11. Number of modules per project
How do you manage the merge conflict?	12. Automatic/Semi-automatic/Manual
How many reported defects do you register for a project?	13. Number of internally/externally reported defects
How much is the average time to find a bug? And to fix it?	14. Time per bug fix
How many different variant of a	15. Number of variant of a program
program do you manage?	(number of customers)
How is divided the time in the product development life cycle?	16. Percentage of time for Design/ Coding/Production/Maintenance
Question about the tool	9. Tool characteristic

3.2.3.2 Parameters from the model

- **Decrease of the externally reported defects (13)**: This is the main parameter for the ROI estimation. A decrease in the defect report arrival rate provides less time to develop and maintain the product, which means that there is a saving time on the project and a reduction on the salary paid. Developers are less disturbed from the continuous requests for bug fixing, thus they can work in a better way and more efficiently.

- **Ability to trace the original product through its development (14)**: The documentation achieved by configuration status accounting, the version control tool and the configuration change control (with the CCB) bring to traceability. This property permit to spend less time bug fixing. Knowing the average time spent per bug fixing and the number of reported defects it is possible to quantify the total saving time.

- **Reducing the number of errors. Errors given from double maintenance, shared data problem, simultaneous update, working in chaos (10, 11, 5, 6)**: All the classical problems that are generated working in group are a setback for the developers' work. Use of SCM increases productivity for human resources. This parameter depends also on the size of the developer's group and on how many programmers are working simultaneously on the same module.

- **Save time with an automated software builds (9, 11, 16)**: This parameter strictly depends on the utilized tool and on the dimension of the project. An automated build procedure can save time and money. It is reasonable to estimate a decrease of workload associated with the build tasks between 12% and 40%, depending on the type of build [6]. The possible types of build are:

  – Development Build: It is a build done by a developer that is never released to QA. It is a frequent build; for this reason a save time in this category represents a relevant benefit. Automatic build tool provides a support for *reproducible build*, which is a build that is reproducible at any future point in time and *incremental build*, which is a build where only the changed component are rebuilt.

  – <u>Release Build</u>: It is a formal build that is deliverable to QA and may eventually be seen by the customer. This kind of build is not a frequent task in development life cycle and has to be done with high quality assurance. For this reason, saving time, in this case, is not a relevant benefit for the whole project; in this case automatic build tool assures that all the right version modules are linked, with an obvious advantage for QA.

- **Parallel work, automatic merge (12)**: This feature can help the programmers with their work. The degree of help depends on the characteristic of the used tool (how the check-in check-out process is managed, if an automatic merge tool is supported, if there is the possibility to lock the files, what are the methods of programming, etc).

- **Gain factor fixing bugs in different variant of a program (15)**: The purpose of a professional software development project is to build a family of computer programs. SCM controls all the members of a program family. All the efforts to manage many different variants of a program are minimized by the tool and the knowledge achieved through the documentation. If a bug is found in a module, which is present in some variants of a program family it will be fixed one time for all the different variants. The more number of variants that are present in a project, the more benefits that can be obtained.

- **Be able to bring out the work earlier, before other companies**: All the different benefits of using SCM make the production faster. This is a direct economic benefit saving time and money. There is also another more subjective benefit, achieved coming first with a product in the market; this benefit could be vital for companies which work in a high competitive market.

- **Decrease the time required to respond to end-user requests and inquiries (10, 14)**: This is a consequence of the better organization of the group. The decrease of the time required to respond to end-user requests is a difficult quantifiable benefit, that facilitates the relationship with the customer.

3.2.4 Goal 3: Quality Assurance

The traceability achieved by configuration management activities (in particular configuration control and configuration audit) helps testers' work. Audits are a guarantee that everything follows the requirements and verify that approved changes have been implemented.
Using SCM tool and following SCM process improve the quality of developers' work. Software development, in fact, is a people-intensive activity, and quality must be considered at every person-to-tool interface. Ensuring a high-quality work environment makes the production processes safe.
Configuration identification and configuration control assure the stability of the baseline and audits guarantee the consistency of the work though its life cycle. All these characteristics of SCM add quality to developers' work.
All these benefits are difficultly quantifiable for ROI estimation.

3.2.4.1 Questions and Metrics

Question	**Metrics**
How is implemented Configuration Audit? (When configuration audit are done, number of audits)	17. Average Time spent for an audit
How is the quality of the product in the test phase?	13. Number of defects in the code •Test phase (internal defects) •From the customer (external defects)
How much time is spent to test the product?	18. Percentage of time spent to test the product
What are the supported environments for developers to help the QA?	9. Tool characteristics

3.2.4.2 Parameters from the model

- **SCM activities (in particular configuration audit and configuration control) assure the customer that he gets what he paid for. (17)**: All the four SCM activities bring traceability and show how the project is going on. The continuous interactions with the customer through the CCB permit to know and satisfy his will. All this hallmarks add costs in the production.
- **Audits at the end of each phase of the development life cycle assure the consistency of the work. (17)**: Configuration audits assure that developers are working on a stable and approved version of the product. This procedure adds some costs, even if in the successive phase there is a return on investment, because of the improved productivity of developers, who work in a stable environment.
- **More quality addresses decrease of the externally reported defects (13)**: A Norwegian study points out that using SCM in the production of software product brings a decrease of the externally reported defects of 36% [5]. This is achieved also because there is an increment of the internally reported defects. SCM helps to find and fix the bugs before the production and the last release of the product.
- **Traceability addresses less time for V&V and testing (18)**: The documentation produced during the development achieves traceability and, with the help of the version control tool, the tester work is easier than what it used to be.

3.2.5 Goal 4: Punctuality

Everything in management has to be planned, otherwise chaos would reign. SCM is a control discipline, thus, one of the first and important activities in configuration management is the planning process. All the performed SCM activities, the schedule of the activities, the assigned responsibilities and the resources required, should be included in SCM plan.
The CM plan is something that is developed by the configuration management

office. It should reflect the program management global plan, clarifying how the project will be done. Making and following a plan is crucial for a successful SCM implementation.
A plan helps to work in an ordered ambient, where the project manager can check how the job is going on.

3.2.5.1 Questions and Metrics

Question	**Metrics**
Who is responsible to implement the SCM?	19. Number of people involved in the creation and implementation of the plan
What are the planned activities?	20. Type of planned activities 21. Number of planned activities
Are the plans being followed?	22. Number of conducted activities 23. Rate of finished in time project

3.2.5.2 Parameters from the model

- **Changes are planned, their impact is assessed (19, 21, 23)**: The plan with configuration identification and the CCB impact analysis permit that all the activities are done with the project manager overview. Changes are planned and their impact is assessed. This guarantees that the project will be finished on time.
  - Having the product before other companies
  - Save development money (Cost/day)
  - No forfeit for belated end project
- **Achieves a sense of organization and control instead chaos (3, 20, 22**: The plan achieves a sense of organization and control instead chaos, helping developers to work in a ”good way”. This achieves more quality and productivity of the staff.
- **Provides visibility of the project (10, 20, 21, 22)** Having a plan and doing status accounting and reviews provides visibility of the project

helping the work of project manager.

3.2.6 Costs

The costs depend on the tool used for implementing SCM and on the degree of SCM implementation. The following metrics are associated to the first term:

- **Licenses and maintenance cost (1, 9)**: Licenses and maintenance cost strictly depend on the used tool. An open source tool like CVS, which is free, can be used or an expensive and complete tool like ClearCase, which adds costs because of the maintenance of the system too, can be adopted. Another variable is the number of users and the necessity of the company.

- **Teaching (1, 7, 8 for the costs of teaching for new employers)**: The costs to train the system administrator and the developers are a big part of the total costs. This depends mostly on the learning curve of the tool and on how many developers will use it.

For the second category of costs there are mainly:

- **More work for the configuration manager (19, 20, 21)** In some case the configuration manager is a new character in the company, in other cases his work is done by different people from the staff. The implementation of SCM usually adds some work with its formal procedures.

- **Time spent doing audits (17)** Audits are really important for the quality assurance and to facilitates the work of the developers. Depending on how they are implemented and how many times they are done in the development life cycle, configuration audit adds work for the developers.

3.2.7 Tool questions

Core artifact function	**Goal**
Does tool support File versioning	2
Does tool support Directory Versioning	2
Does tool support baselines (tags)	1
Does tool naturally support file renaming/moving	1
Does tool have mature and easy to understand GUI	1
Does tool have flexible reporting facilities (queries/script)	1
Does tool manage transactions atomically (as a whole)	1,2
Does tool allow user to change views based on paths and baselines	1
Does tool allow you to identify who is working on a file	1
Does tool allow concurrent file access	2
Does tool allow file locking	1
Parallel Engineering	**Goal**
Does tool support parallel development/engineering (branching)	2
Does tool support basic merging (file, directory and hierarchy)	1,2
Does tool have intelligent merging (conflict resolution, merge memory)	1,2
Core Activity Management	**Goal**
Does tool support Change Reports	1,3
Does tool support Defect Tracking	3
Does tool support Job/Work Tracking (e.g. through change sets)	3
Does tool have life cycle tracking (relating objects to promotion/releases)	3
Can tool control process at role (user/group level)	1,3
Can you define quality gates/process event-triggers	3
Architectural Factors	**Goal**
Does tool support remote user	1
Is local and network Performance satisfactory	1

Transparency/Tool Acceptance	**Goal**
Is tool easily used from client perspective	1
Can users to work transparently (i.e. acceptable daily operations)	1
Does tool easily support working off-line	1
Total Cost of Ownership	
Cost of Licensing	
Cost of Hardware	
Cost of First Year Support	
Cost of Future Support	
Cost of Training	

Chapter 4

Discussion

In this chapter the results of this thesis are discussed. The first step is a self-evaluation, analyzing the qualities and the weakness of this model, aiming to the future works to solve these problems. In paragraph 4.2 a sort of validation has been done, comparing this study to other already existing works and reporting the feedback taken from a real company. Finally the chapter ends with some hints for the future works.

4.1 Self-evaluation

The objective is to create a complete model, concentrating the efforts on the technical and theoretical aspects, not including the empirical observation. The collection and analysis of data is delegated to future works.

This study starts with a research of what SCM means, what changes in the product life cycle and what changes in the working life of people involved in software production. This documentation permits to have a wide view of the discipline and to understand most of the implications concerning the implementation of SCM in a company.

From this study it is possible to understand that there are a lot of new tasks concerning SCM, from the creation of the configuration management plan to a simple change process, from the use of a simple version control tool to a complete implementation of the four activities. For this reason SCM could be performed in many different ways. For each implemented tasks there is a cost

and a relative benefit. Generally the costs are concentrated in the first part of the production; these costs bring about delayed benefits in the latest part of the production. For example the most of the benefits of SCM are associated with the maintenance phase.
One important thing to take advantage of SCM discipline is how it is implemented. The integration of all different tasks is very important for the traceability. In extreme cases a bad integration of SCM activities causes the loss of the benefit of traceability. For this reason it is important to have a unique tool or some hi-integrated different tools to benefit all advantages of SCM.
Along with the possibility of implement different degrees of SCM discipline, there are many possible SCM tools to use. From a simple versioning tool like Microsoft Visual Source Safe to a complete instrument like Rational ClearCase. During this study it became evident that SCM is a discipline intrinsically dependant on the context of application. The possible benefits and the respective gain factors could vary depending on the different case of studies. For example the benefit achieved help in reusing existing code could provide a high profit rate in company, where this type of development is extensively used, while this benefit is not present in another context. For this reason it is impossible to create a general ROI formula which could fit all the possible cases.
Understanding that the ROI value has to be calculated by the company itself, knowing the context where SCM is applied and how the discipline is implemented, the original target of this work (ROI formula for SCM) moved to a more general product, which could include all the aspects, not only the most known and quantifiable part of them, creating the model and the method.
The difference between this model and the other is that the second ones must consider only the direct and obvious costs and benefits to create an explicit model which can be generally applied to a company. These models give a ROI value, using standard fixed rate, without considering the specific case. This work represents also a guide for the costs and benefits of SCM (the model) and a method to find the right metrics and weighs up these costs and benefits. The model, with the three column matrix division, permits to consider all the possible costs and benefits of SCM, including those subjective ones which can't be included in a general formula. The potential user can quickly understand all

the aspects connected to SCM. The three way classifications[1] helps the user to focus on the aspects to which he has to pay attention for a right interpretation. The measurable parameters of the model can be directly used in the formula, partially measurable parameters can be included in the formula after having considered the context where SCM is applied; and not measurable parameters column can be considered as a check list, where the user can consider all the possible costs and benefit, which cannot be quantified but are nevertheless important for the final decision of applying the discipline or not.
The use of the model and the method can be seen as a guide to create the own formula, helping to find the right benefit and consider all the costs, assuring to have a more true prediction of the return on investment.
The flexibility of the model permits to apply it in both total and partial SCM implementation. The first objective, in fact, was to create a model to calculate the return on investment in a company, which from a zero degree of SCM implementation wants to apply all its activities, passing to a full SCM process. There is the possibility that the company has already partially implemented SCM in its production process or, as previously argued, to have a partial implementation of this discipline; for these reasons it is important for a ROI model to be applied in all the possible cases.
This work does not provide the user of an explicit formula to estimate the ROI, because that formula could be too imprecise, but it provides a guide to find and quantify better the costs and benefits of SCM discipline. Understanding them, a possible user could better fill up the module and create, with the help of the model, the proper formula which better fits his problem.
The most obvious weakness of this model is the lack of background statistics which can confirm the results. It has not been possible to collect data to give an esteem of model parameters value [2] because of the limited time. This could be the subject of a possible future work. A possible growth of this work could be to add in the future an indication of the range of the gain factors value in the model and extend the method to choose the better value of those parameters.

[1]The three column division; measurable, partially measurable and not measurable.

[2]All the rate of improvement of SCM as the rate of decrease of the number of bug fix and of bug fix time,α_1 and α_2

The direct analysis of SCM implementation takes quite a long time. It must be found a set of companies which have just introduced SCM in their production cycle, it is necessary to follow the companies from the early phases of the implementation. Then, to register substantial changes in the production, some month have to be elapsed. In Larsen and Roald's study, for example, their project went on for 22 month long to derive their conclusions [5]. The limited time of this work is not enough to deal with this job. For this reason it was not possible to do a direct validation of the work. Only a feedback from a local company has been gotten to see the applicability of the model. The results are showed in paragraph 4.2.4.

4.2 Validation

The subjectivity of the problem and the duration of this thesis make it difficult to have a full validation. For this reason the only way to validate this work is to compare the results with the already existent works and to have a feedback from companies interested in introducing more SCM assistance in their production to test the applicability and to check its completeness. For this reason the thesis is matched up to other works to show the improvements of the model/method. In particular it is compared with academic works as Larsen and Roald's "Introducing ClearCase as a Process Improvement Experiment" and commercial works as Merant white paper "The Business Case for Software Configuration Management". Finally a validation of the applicability of the model is done with a local company, which is considering this work for a future project.

4.2.1 Comparison with academic works

The unique academic document which deals about the problem of calculating a ROI model for SCM is Larsen and Roald's "Introducing ClearCase as a Process Improvement Experiment".
The aim of their project was to install ClearCase, as a configuration management system, and measure and evaluate the impact of the new system on

processes and products. They introduce ClearCase as a PIE[3] within Sysdeco GIS Company. The project has three phases, ”preparation”, where they measure the initial process; ”experiment”, which consist of system introduction and training; and ”evaluation”, where they measure the improvement results and report them. To be able to quantify improvements results they use a GQM-like process to establish candidate metrics, resulting in 16 possible metrics. The number of metrics used in the final assessment is reduced to 5, due to either insignificant data sets or changes that made data incomparable. The project goes on for 22 months; the goal of the experiment shows significant results as a reduction of the defects reports per month by 36%, a reduction of urgent defect fixing time of 6% and the effort spent on maintenance has been reduced by 33% [5]. Larsen and Roald’s article deals with SCM and its application in a company, but there are many differences from this work.
The aim of this thesis is to create a complete model which can be generally applied in all SCM introduction cases. This work focusses on the technical and theoretical aspects not including the empirical observation. This work is essentially concentrated in the first phase of Norwegian’s work, the process modelling, as they focus on measurements and quantifying improvements results. This has been possible because they are observing a particular implementation of SCM (the only installation of ClearCase in only one company) not considering all the aspects relative to whole discipline. As that work is considering a particular case, there is a limitation on using their improvement results. Those results give a confirmation of the presence of some entries in the model such as: training costs, decrease of the externally reported defects, the reduction of bug fix time, decrease of maintenance efforts. The values of Larsen and Roald’s experiment could be used as an indication of the improvements dimension but they can’t be used as certain and general results because they depend on the context of their experiment. For example the number of employees working in the project and how the SCM discipline is implemented. This is a validation of the philosophy of the model which expects an interpretation from the users to fit the value of the improvement rate to their particular case.

[3]PIE, Process Improvement Experiment: it is an experiment that should measure and report on the effect of one specific process improvement step in one company

4.2.2 Comparison with vendor white papers

There are many documents of the majors SCM tool vendor companies available on the net, which demonstrate with a ROI calculation the need to buy their product. These documents represent explicit model for the return on investment, but they are a mere advertisement to sell better their products; for this reason they are a particular case of SCM implementation and they are not reliable at all. Vendors' white papers usually promise incredible improvements on the production without demonstrating them. Merant white paper starts with a magnification of the benefits as "We will show how SCM can reduce developer workload by 76 percent, increase product quality by 80 percent and shave 38 percent off time to market." [6]; but as the text goes on there is no demonstration of these benefits.
These documents are used to compare the model/method approach with their approach of ROI calculation. In fact their ROI formula is feasible in the real world and uses inputs which can be reached from the statistics of a company. This way it is possible to have a validation of the completeness of the method approach. With the method proposed in this thesis the user is guided to establish candidate metrics and measurements. These metrics have to be observable to the user to be an input for the model. As vendor's models are feasible, the input of their model can be compared to the one suggested from the method. Observing the method it is possible to see that it includes all the topics which Merant model deals with, as version management, problem tracking and build management task.

4.2.3 Technical validation

The article "Impact of Software Research on the Practice of Software Configuration Management" has been used to have a technical validation of this work. This document discusses the evolution of SCM technology from the early days of software development till nowadays, tracing the critical ideas of the field from the early inception to their maturation in commercially and freely available SCM systems. This paper traces all the application of SCM in the software industry.
This document was published only at the end of this study, for this reason it

was not possible to use it for the documentation. The article is used as a check list assuring that all the critical aspects of SCM has been included validating the completeness of the model.
This paper traces the history of SCM through the functionality of the SCM tool. They partition the areas of SCM tool functionality into three major support areas: product, tool, and process. With the description of the evolution of these three areas they represent approximately how the field as a whole evolved, first only supporting file management, then integrating sophisticated tool support, and finally incorporating advanced process control. Considering SCM tool applications they mostly focus on the benefits of the developers. With the three points of view analysis it was possible to have a wider view considering both developer's and manager's benefits. The aim of this study is to analyze the theoretical aspects of SCM discipline, and for this reason a comparison with this article, which contains all the possible SCM tool implementations, is important to understand if all its aspects have been included.
The authors' analysis is compared with the results to have a feedback from a "practical" point of view. As it has been already said, they divide tool functionalities in three support areas: product, tool and process.
In the first category they focus on the core functionality of SCM; the management of the many files that constitute a software system. It is presented the basic versioning technique, merge support and the different access type for artifacts in the workspace. In this thesis developer's benefits using a version and merge support tool are showed in paragraph 1.4.3, including those benefits as entries in the model. The different kind of artifacts selection in a workspace are not analyzed because the objective of this work is not to analyze the detail of a specific SCM implementation. The need to support developers, who work in a large project, is shown.
In the tool support area the authors focus on the workspace control and on the building support. The document shows, with the evolution of SCM tool, the different type of workspaces, from the first implementation of SCCS and RCS with no workspace support to the hierarchical workspace of CVS and the views of ClearCase. The benefits of using a workspace, allowing developers to work in isolation, are showed not considering the implication of using different kind of workspaces because the objective of this thesis is to create a model

independent from the particular SCM system implementation. These benefits are included in the entries of the model such as ”Reducing the number of errors (given from double maintenance, shared data problem, simultaneous update, working in chaos)”, ”Pleasure of working in stability with a baseline and the own workspace”. Both the articles and this thesis demonstrate the benefit of having an integrated build tool in the SCM system.
In the last category the authors argue about the process support which addresses to control project quality, cost and delay. As shown in this work, these are the main benefits of the increased organization of SCM discipline. The document shows how the task of SCM systems evolved from managing just files, to managing people collaborating to the development and maintenance of software. This is the most significant part of this thesis which concerns with the managerial benefits of implementing SCM.
The document ends with a discussion about the areas in which the research was successful. One of this important implementation is the distributed and remote development; an entry of the model shows this benefit, but this aspect is not analyzed in depth because this is an intrinsically not quantifiable benefit, because it depends on the politics of the company.

4.2.4 Practical validation

During the development of the model it has been possible to have a feedback from a real company, who is interested to change the SCM system and wants to calculate the economical impact of this change.
They evaluated the possibility to use the model in a project that they are going to start soon to calculate the return on investment changing from Visual Source Safe to ClearCase. This work was submitted them to know, as a customer of a ROI model, if it includes the right things; and, as a company that already has CM in place, what is their opinion about its completeness, correctness and if they have experience with the size of the parameters for costs and benefits used in the model. They have expressed a good opinion about both points of view. The company has not used the model yet, but they are planning a project to calculate the economical impact changing SCM tool and this way changing the degree of SCM integration where they are going to use it. They

have appreciated the flexibility of the model/method structure which could be facilitate to fit to their particular case. This fact represents a validation of the applicability of the model.

4.3 Future works

Some possible future works are already presented in paragraph 4.1. One of them could be the estimation of gain factors value. This work could be developed following a set of companies which have just implemented SCM discipline in their production and collecting the information with the metrics suggested in the method and analyzing the historical data. This is an analysis that takes a long time to be done, because to collect enough data for the estimation and to see considerable changes in the production, it is necessary to wait time of years order.
Another possible future work could be the application of the model and the method to a set of companies, which intend to implement a SCM system in their production. This could be useful for a validation of the efficacy of the method and the completeness of the model. The objective of this work could be to test the applicability of the model/method, estimating model parameters with a customize study for each company. This is a feasible work for a possible extension and validation of this study.
With the feedback from the real world from interviews or from direct application of the model in companies a possible future work could be a refinement of the model. Its entries could be decomposed in more sub-parameters or some parameters could migrate from a column to another. The objective of this work could be to promote the entries from not measurable column to the measurable one giving a new method to quantify these entries.

Conclusion

This thesis has been motivated by the need of a tool which can ease the comprehension of possible costs and benefits deriving from the introduction of SCM in a software development company. The original target of this work was to create a ROI formula for SCM in order to directly quantify the gains that derive from applying this discipline in the software production industry. During this first study it became clear that the original perimeter had to be extended to include all SCM aspects, not only the well known and quantifiable ones. For this reason a model has been created, in which both costs and benefits are organized in three degrees of measurability: measurable, partially measurable and not measurable. These three degrees allow us to organize the parameters in three groups. The first group includes the parameters that are quantifiable and non quantifiable, depending on the context where SCM is applied, these parameters are usable in a generic ROI evaluation. The second group includes those parameters that need an interpretation because of their dependence to the particular SCM implementation. Finally the last group includes those parameters that are not economically quantifiable.
The result is a matrix where all the aspects of SCM introduction have been summarized, focusing on its completeness.
The presence of many subjective aspects associated with a SCM system clarifies the necessity of method which could guide the user in selecting and quantifying the related costs and benefits. This method is based on a GQM analysis which, through the definition of SCM goals and a list of related questions, helps the user to collect the information needed for the ROI model and helps him to be aware of SCM costs and benefits.
A ROI evaluation algorithm has been included, which uses only the measurable parameters. By using the method and by interpreting the particular company

case, the user will be able to customize it with the entries of the second column of the matrix. The flexibility and the completeness of the model are the most important improvements of this work in comparison to all previous studies which consider only the direct and obvious costs and benefits. Another focal point of this study is to create an explicit model that can be generally applied to all the SCM systems. The three column matrix division model gives visibility on all the possible costs and benefits of SCM, including the subjective ones that can not be included in a general formula.
Compared to the few works that have been done on this subject, this model is more complete in what concerns the cost and benefit parameters.
Unfortunately it has not been possible to validate these results with background statistics because of the long time that would have been necessary to follow a set of companies and collect the resulting data; this validation could be the subject of a future study: testing the applicability and completeness of the model and estimating the value of model parameters.

Glossary

CCB: Configuration Control Board, it is a permanently established committee of representatives of the organization. Its primary mission is to ensure complete impact assessment and analysis.

Check in: It is the update of the repository. It consist on a merge of the work, done in developer's workspace, with the repository files

Check out: It is the copy of some files from the repository to developer's workspace

CI: Configuration Item, it is a component of the project with associated documentation

ClearCase: It is a complete SCM tool

CMM: Capability Maturity Model is a model for judging the maturity of the software processes of an organization

CSA: Configuration Status Accounting

CVS: Concurrent Versions System, it is a version control tool

DIFF: The difference between the program files

Double maintenance problem: The problem of keeping multiple identical copies of the same software

FCA: Functional Configuration Audit verifies the functional characteristics of the production

GQM: Goal Question Metrics

GUI: Graphical User Interface

IEEE: Institute of Electrical and Electronic Engineers

PCA: Physical Configuration Audit verifies the form and fit of the production

PIE: Process Improvement Experiment: it is an experiment that should measure and report on the effect of one specific process improvement step in one company

QA: Quality Assurance

RCS: Revision Control System, it is a version control tool

ROI: Return On Investment

SCM: Software Configuration Management

SEI: Software Engineering Institute

Shared data problem: It happens when many people are simultaneously accessing and modifying the same data. The changes made by a programmer may interfere with the others work

Simultaneous update: It happens when two or more programmers are updating the same software and one person's updates is overwritten by the others

V&V: Verification and validation

VSS: Visual SourceSafe, it is a version control tool

Bibliography

[1] Merant: Assessing Return on Investment for Enterprise Change Management Systems.

[2] F.P.Brooks: The Mythical Man-Month.

[3] W.A.Babich: Software Configuration Management - Coordinatio for Team Productivity.

[4] R.Futrell: How You Can Benefit from Software Configuration Management.

[5] J.O.Larsen, H.M.Roald:Introducing ClearCase as a Process Improvement Experiment.

[6] Merant: The Business Case for Software Configuration Management.

[7] D.B.Leblang: The CM Challenge: Configuration Management that Works.

[8] M.A.Daniels: Principles of Configuration Management.

[9] L.Bendix, U.Asklund: Software Configuration Management Practices for eXtreme Programming Teams.

[10] Basili: Software Modeling and Measurement: The Goal Question Metric

[11] Robert Bogue: The Many Faces of SCM.

[12] Chao Zheng: Design Metrics Plan for SCM Practice.

[13] Winston W. Royce, Managing the Development of Large Software Systems.

[14] Spectrum SCM: A Guaranteed Way to Reduce Costs while Improving Quality and Productivity.

[15] J.Estumblier,D.Leblang,A.Van Der Hoek,R.Conradi,W tichy: Impact of Software Research on the Practice of Software Configuration Management.

[16] Comp.software.config-mgmt

[17] http://www.seicmu.edu/legacy/scm/

[18] http://www.cmcrossroads.com/

[19] http://irb.cs.uni-magdeburg.de