Reporting framework-based software process improvement

A quantitative and qualitative review of 71 experience reports of CMM-based SPI

MASTER THESIS

Written by
Håkon Ursin Steen

for the degree of

candidatus scientarium

29th of October 2004

Simula Research Laboratory
&
Department of Informatics
University of Oslo
Norway

"All of science is the telling of stories. Even the most abstract and formalistic manual of mathematics presents its argument in a way meant to convince the reader that things are done in a correct manner. [...] How to give proof of method? How to convince us that one given method is correct? The paradox is well known: either by the same method (circular evidence), or with another (infinite regression).
[...]
Like it or not: all of science is the telling of stories. If you believe the stories or not, is maybe dependent on how well they are told." [1]

"As the beothuk people experienced ...it's not true that winners write history.
It's the other way around: those who write history win.
It doesn't matter if you're losing or winning
as long as you are backed by a decent
media department. Or a well-behaved bard or two [...]." [2]

SOCRATES: Then you have a look round, and see that none of the uninitiated are listening to us – I mean the people who think that nothing exists but what they can grasp with both hands; people who refuse to admit that actions and processes and the invisible world in general have any place in reality.
THEAETETUS: They must be tough, hard fellows, Socrates.
SOCRATES: They are, my son – very crude people. [3]

Acknowledgements

A special thanks to my supervisor, professor Magne Jørgensen, at the Simula Research Laboratory, Oslo for support and healthy critical thought to the most convincing of stories.

Other acknowledgements:

Academia
Geir Amsjø (of *Department of Informatics, University of Oslo*)
The guys and girls at the library (of *Department of Informatics, University of Oslo*)
Tone Bratteteig (of *Department of Informatics, University of Oslo*)
Tore Dybå (of *SINTEF*)
Peter Axel Nielsen (of *Department of Computer Science, Aalborg University, Denmark*)
Tian Sørhaug (of *Centre for technology, innovation and culture, University of Oslo*)

Industry
Tom Handegård, and the rest of the SEPG at Escenic (www.escenic.com)
Liv Moen, and the rest of the people at *Kongsberg Defence & Aerospace* (www.kongsberg.com)

Other influences
Joachim Carlsen (keeping the spirit of Hans Georg Gadamer and Paulus Svendsen alive)
Tanja Gruschke (my fellow student at Simula)
Hein Haraldsen (his clear-mindedness goes a long way)

Charts and statistics were produced in *SPSS 12.0.1* and *Microsoft Excel 2K*.

Abstract

Software development projects have a notoriously high failure rate. Software process improvement (SPI) frameworks have since the early 1990-ies been a suggested remedy for this. The Capability Maturity Model (CMM) is such a framework, but the actual process of *implementing* the CMM has proven difficult for many software organizations. Another problem is that documentation of the actual benefits of CMM-based SPI (CBS) is vague. In a pursuit of rectifying the situation we present a quantitative and qualitative review of 71 published case stories of CBS. With the data collected we set out to examine several issues: first, the potential for software organizations for *learning from* and *reproducing* the almost non-exclusively *positive* results of CBS reported in case stories, second, to what degree the calculations of Return On Investment (ROI) present believable numbers, and last, if CBS is something that is beneficial for the software industry as a whole. We found that, first, because case stories are largely reported by companies that are unrepresentative for the industry as a whole, the average company will have problems learning from and reproducing the results reported. Secondly, we found that calculations of ROI in general in the literature are of doubtful quality, but with a few prominent and notable exceptions which indicate that viable calculations of ROI for CBS *are* possible. Finally, we present a reasoning that indicates that CBS probably is beneficial for the software industry as a whole. Drawing on a tradition in the SPI literature of collecting "success factors" for CBS in assisting implementation, we also present a list of all explicitly reported "success" and "non-success"-factors found in the case stories.

Table of Contents

List of tables

List of figures

"Bridges are normally built on-time, on-budget, and do not fall down. On the other hand, software never comes in on-time or on-budget. In addition, it always breaks down. " [4]

1 Introduction

Software projects exhibit a high degree of failure. Even if the numbers referred in the "Chaos" reports by Standish Group [4, 5] are somewhat off target [6], the general tendency is that there is still a tremendous gap between the current software project success rate and anything that resembles an optimal situation.

Many a "silver bullet" [7] has been crafted in the pursuit of eradicating the monsters of software project failures. Various CASE[1] tools and variations on the theme "agile" application development[2] have surfaced to try to stop the evils, unfortunately without giving significant and traceable impression on the figures bi-yearly popping out of Standish Group [5]. A tidbit different from the rest of "silver bullets" is *software process improvement* (SPI).

Organization-wide frameworks for SPI take into consideration research indicating that the most important reason for software project failure is *lack of management control* of project development efforts [11]. The currently most popular SPI framework is called the *Capability Maturity Model for Software* (CMM). The idea is simple: by enabling a software organization to increase its process maturity capabilities, it will at the same time increase its potential for completing software projects at estimated time, at budget, and in accordance to specifications. Sounds nice.

But implementing the CMM has during the last decade unfortunately seemed to be no "walk in the park". Scientifically sound failure rates do (initially surprisingly) not exist (for more on this, see section 2.2.4), but we settle for the notion that failed implementations of CMM probably exceed 50% of all SPI efforts. Based on calculations on survey data, a group of

[1] CASE is an acronym for *Computer Assisted Software Engineering*. It covers a wide range of different types of software tools used by the developer to support various processes of software development [8].
[2] Including development methodologies like *Extreme Programming* [9] and other methodologies founded in iterative systems development inspired by Barry Boehm's seminal article on the "spiral" model [10].

prominent software engineering researchers daringly state that resources expended to SPI implementations are "certainly in the billions of dollars" [12] (it's here not quite clear if they account for the U.S. or the world). Anyway, assuming these two approximations are closer to truth than not, the result is: very much money spent on failed SPI initiatives.

When things go wrong, the failure rates for CMM-based SPI (CBS) is generally met by two kinds of responses:

- **Optimism:** if we only learn from others how to do it, it is possible to implement CMM in a more efficient manner than we do now (this is the typical approach, and what this thesis will concern itself with)

- **Pessimism:** CMM has internal contradictions [13]; CMM is probably the wrong kind of approach for SPI [14]; SPI is the wrong focus for our problems altogether [15]

For over ten years, experience stories have been published in the literature, documenting purportedly successful CMM implementations. Yet, with a continuing high failure rate, the case is not closed.

If we treat learning from other organizations' experiences as a *necessary* condition for improving the sobering status quo, an inquiry into what degree learning is possible, is of greatest interest. Therefore, our first research question is this:

Q1: To what extent is it possible to learn from case stories of CBS?

We set out to examine the potential fruitfulness of learning from CMM experience, and we do this by mapping characteristics of 71 individual case stories of CMM implementation in accessible literature. We map these stories into 29 different categories in what can best be described as a *combined quantitative and qualitative* article review.

Secondly: When organizations embark on software process improvement activities, their motivation is often economical - they want to reap some benefits from their investment in SPI. Financial benefits from SPI are often expressed in the relationship of *Return On Investment* (ROI). A ROI of 2.5:1 means that for every monetary unit invested, you may expect to get 2.5 monetary units in return. Case studies of CMM sometimes contain a calculation of ROI. But what is their rationale for calculating ROI? How do they rectify

collecting the potentially unlimited amount of positive and negative effects of a SPI initiative in one, single equation? This is the background for our second research question:

Q2: What is the rationale for calculating ROI in case-stories of CBS?

The "cost of optimism" is undoubtedly rather high for CBS. With a failure rate maybe even close to 70%, there are apparently a lot of companies not getting the pay-off they hoped for when initially embarking on a CBS initiative. But, is it, despite potentially high ROI's for those who succeed, worth the money *for the software community as a whole* to embark on CBS? Maybe is not only the cost of failed initiatives causing an unhealthy polarization between those who "succeed" or not, but perhaps CBS actually also causes *a weakening of the IT industry as a whole*? The final research question is thus:

Q3: Are the costs for CBS higher than the benefits for the industry as a whole?

2 Theoretical foundations

2.1 Previous CMM case study reviews

There has been no studies like this one before in the literature (as we know of), both with respect to the amount of case studies in the review, and also to the relative thoroughness we show with a high number of coding categories. For a review of related research, please see section 2.3.1.

2.2 Software process improvement

2.2.1 Quality revolution

Software Process Improvement (SPI) is a discipline inspired by classic modernistic principles for *scientific management* of organizations, as expressed by Frederick Taylor [16] and later by William Edwards Deming [17]. *Taylorism* was a necessary set of theories for organizations to be able to go through the industrial revolution experienced from the middle of 19th century, until the beginning of the 20th. Deming's theories on *statistical process control* was what would get Japanese industry back on its feet after the second world war [18].

Taylor's ideas fit perfectly for the classic modern production facilities with obviously optimizable processes of production, like steel mills and car factories. Deming's thoughts on putting processes under statistical control can be said to be a further elaboration on Taylor's ideas.

Another important thinker in the quality revolution was Phil Crosby [19]. His slogan "quality is free" played on the thought that if you cancel out a given company's current cost of *not* adhering to quality standards ("cost of non-conformance" - which includes expenses related to customer support, product service, bad reputation and loss of sales opportunity), with the cost of implementing improved systems for quality control ("cost of conformance" – for instance the cost of implementing statistical process control in a company), you will (allegedly – this belief has at least inspired the Total Quality Management-revolution, including the CMM) end up with that "quality is free".

2.2.2 Software process improvement

In the end of the 1960-ies a new industry emerged together with the breakthrough of computers – *computer software*. Mainly the toys of large countries' defense industries and some large corporations (amongst them IBM), the industry lacked established "best of practice" methods for its development. At the time there was a chronic shortage of educated software developers (simply because the education did not exist at the time). "Best of practices" was *whatever practiced* in a given community. Each company had their own, special process for developing software. The development of suitable processes became a low priority issue. This was because successes of companies was more dependent on company alliances, and who got their ideas first out on the market, than anything else[3] [20, 21].

The "age of puberty" of the software industry began in the late 1970-ies (with the first "unbundling" of delivered software from delivered hardware). Because few "best of practice" processes were known, the 1980-ies was a decade of software organizations drawing unnecessary large amounts of resources, with more often than not, ridiculously delayed projects as the result. Many projects were never even finished. The defense industry, with its large software projects, was severely affected by this. To illustrate, an unpublished study from the U.S. Department of Defense (DOD) tells about a reality where, out of 17 large contracts committed by the department, the average development time of 28 months per project was exceeded by, on average, *20 months*. A classic 1987-report from DOD, which examines "the software crisis", emphasizes that "few fields have so large a gap between best current practice and average current practice" and concludes with that the problem is *management*, not *technology* [22]. The DOD no longer wants each and every company to have their own, more or less, random processes for developing software.

2.2.3 A brief account of the CMM

What is the Capability Maturity Model (CMM)? From the backflip [22] of the "official" hardcover book, Software Engineering Institute (SEI) describes the CMM as "a framework that demonstrates the key elements of an effective software process". It was spawned by funding from the U.S. Department of Defense, and created over a period of five years from its inception in late 1986 at the SEI of Carnegie Mellon University, Pittsburgh, Pennsylvania. Its first "official" release was in 1991.

[3] Process sceptics will of course point to that this is still the case today.

This CMM is divided into five "maturity levels", beginning at a "lowest" maturity (level 1) to a "highest" maturity (level 5) (inspired by the five stages of Crosby's quality management maturity grid [19]). Every software producing organization is by "default" at level 1 (hence it is also called the "initial" level). An organization can be *assessed* at any other given level, but to be assessed, the organization has to have implemented that level's set of *key process areas* (KPAs). For an overview of the levels and the according KPAs, please see illustration 2.1 (quoted from [23]), below. At level 1, the software process is characterized by being *ad hoc* and based on "fire-fighting", "heroics" and "working overtime". At level 2, focus is on establishing repeatable project management processes. At level 3, focus is put on implementing tailored software processes throughout the whole organization. At level 4, processes are giving quantitative results and are put under "statistical control" (see 2.2.1). Ultimately, at level 5, the quantitative data is used to *continuously optimize* the organization's processes.

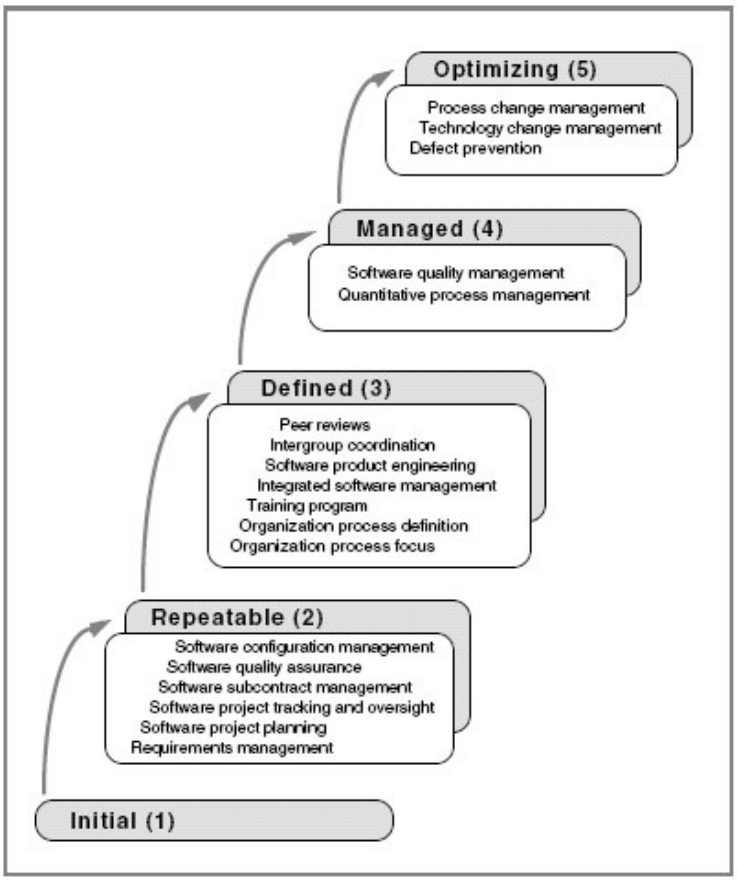


Figure 2.1: The five maturity levels of the CMM

The CMM is an "incremental" framework: any given level requires *all* levels below. For example, for an organization to fully qualify as a "level 4" organization, the KPAs "Software quality management" and "Qualitative process management" have to be fully implemented, in addition to *all* the KPAs of level 3 and level 2.

In contrast to the ISO 9001 and other frameworks similar to the CMM, there is no central certification authority for the CMM [24]. For an organization to declare that they "are" at a certain level, an *appraisal* (a "thorough" assessment) has to be done by a SEI-certified "lead assessor" (a person specially trained in performing CMM-appraisals with software companies). After a valid appraisal has been made the organization can, at its own discretion, use its level for competing for contracts amongst customers [25].

This thesis doesn't give room for further elaboration on the process details of CMM. I recommend as an introductory text the first chapters of the easily, electronically available "Capability Maturity Model for Software (Version 1.1)" [23].

2.2.4 On the alleged failure rate of software process improvement initiatives

Much of the scholarly research into software process improvement the last decade has originated in a common belief that software process improvement initiatives in organizations tend to fail. But, being skeptical: what empirical evidence is the belief of a failure rate founded on? We set out to examine the foundations for the belief that SPI tends to fail, and found more problems than we had hoped for, as will be apparent in the following sub-sections.

2.2.4.1 "The Debou reference"

In his doctoral thesis "The Role of Commitment in Software Process Improvement" [26], Pekka Abrahamson refers to an article by Christophe Debou [27] when giving a foundation for his claims of a high failure rate in software process improvement.

Debou argues that more than 2/3 of SPI initiatives fail. This he bases on 1) an *unreferenced* and *anonymous* statement that he quotes from some "SPI forum"[4] and 2) a few lines of personal musings (which have no solid empirical foundations.)

2.2.4.2 Peter-Axel Nielsen's calculations based on the Maturity Profile

The *CMM Maturity Profile* [28] is a bi-yearly document published by the SEI, and gives a handful of statistical presentations of assessment data reported to the SEI from all over the world. Based on this document[5], Ngwenyama and Nielsen [13] have calculated that CMM has a failure rate at around 70%. In an e-mail-message from Nielsen [29] to yours sincerely elaborating on the rationale for this calculation, the argument is as follows (my translation from Danish):

> CMM is a "measurement model" where success is measured by the ability of an organization to reach the next given level. In other words, an "assessment" is followed by a "re-assessment". CMM is in this way a model which should help organizations reaching a higher level. The question is, then, does the CMM have this effect?

The calculation is as follows:

> In the period of 1997-2002, 1756 organizations have been assessed (measured), but only 451 is re-assessed (re-measured). This gives a re-assessment-rate of 25.7%, in other words around 75% failed implementations of the CMM. [Of the re-assessed organizations], there is 15.5% that does not increase its level, which gives a [corrected] success rate of between 21-22%. The error rate is therefore clearly larger than 70%.

This calculated "failure rate" of above 70%, is indeed a brave initiative to calculate a failure rate, because no failure rate has been attempted calculated and published before. Unfortunately, this is not a "failure rate". This a *re-assessment-rate*, as emphasized by Magne Jørgensen [30]. The calculation contains several problems:

A) The rationale for the calculation is wrong. It is correct that it is "implicit" in CMM that it *is* a goal to reach the "next level". But it might neither be economically feasible, nor even *interesting* from a software engineering-point-of-view, as the following case excerpt from a case study from Texas Instruments clarifies (my italics):

> For example, it makes sense to strive for SEI Level 3 and beyond for fielded, 2167A flight-line support equipment software, while letting In-Circuit manufacturing test software remain at SEI 2. The important point is that SEI Level

[4] The anonymous and unreferenced statement was, quoted: "Two-thirds of the U.S. Software Process Improvement (SPI) initiatives did fail so far."
[5] Specifically; Ngwenyama and Nielsen use the 1997 and 2002 versions of the Maturity Profile.

> 5 *is not the assumed goal for all software developments projects*, especially for domains as diverse as test software [31].

This effect, that software organizations may want to "stay put" at a given level, makes it impossible to say that all the organizations that didn't re-assess at a higher failure were CMM "failures". This observation probably *reduces* the calculated failure rate.

B) The calculation does not take into account those organizations that never reached an assessment. It is reasonable to suggest that a lot of organizations that have failed never found the incentive to report their "failed" result to the SEI (just to keep the Maturity Profile a somewhat scientific document). If there is a clear understanding within an organization that one didn't reach the improvement goal – why, even if the SEI stressed that reported results are confidential, report the result?

This effect, that failures probably are underreported, probably *increases* the calculated failure rate.

2.2.4.3 A preliminary conclusion

From our research of the Debou and Nielsen articles, *there exists no empirical valid calculation of a CMM failure rate*. (That said, we still believe that there are scientifically sound reasons to account for a high CMM failure rate. The reasoning for this is found in section 6.2.)

2.3 The case study as unit for scientific inquiry

2.3.1 Earlier research

We have found only one article that is systematically evaluating CMM case studies for both qualitative and quantitative content, as we do in this study, and that is "An Analysis of Some 'Core Studies' of Software Process Improvement" by Rainer and Hall [32]. The authors analyze 14 scopes of improvement in 11 different organizations. They draw interesting conclusions regarding effects of organizational stability on software process improvement initiatives, and process change expertise, but both are out-of-scope for this thesis. Rainer and Hall make no assumptions regarding ROI for CBS, nor any explicit assumptions regarding success factors.

2.3.2 A question of formality: "Case study" or "experience report"?

The epistemological status of case studies is unclear, at best. It is generally accounted for that it is possible to learn from case studies, and this is perfectly why they are scientifically interesting. But it depends on the context the case study is done within.

A general methodology and a standardized protocol for committing a case study will also help to favor the generalizability of case studies. Kitchenham has earlier [33] designed a 17 point checklist[6] for designing and administering case studies so that one can be able to draw more valid conclusions from them. Unfortunately, almost none of the case studies in this review are even close to adhering to the rather strict regime for case studies Kitchenham sketches in her paper (this is also true for the studies conducted mainly by university researchers, like [34], [35] and [36], which could be thought to have been stretching towards a more "scientific" validity). There is one notable exception in the review with regards to formality. But this article with its meticulous scientific form is (ironically) not published in a *software engineering* journal, but in *Management Science* [37].

Thus to use the term *case study* on CMM case studies, is questionable from a scientific software engineering point-of-view. CMM case studies vary in form; from the methodologically very sound (the above mentioned *Management Science* article), to stories with methodology no better than tales of deeds relating to big fish told around a crackling campfire (like [38] – an *EuroSPI* article, by the way).

The term *experience reports* would be a more "scientifically" correct categorization of the cases in this review. Though, we will in this thesis *not* make an effort to make a systematic the distinction between experience reports and case studies when referring to the cases. This is basically because none of the cases in the study qualifies to be called a case study, by Kitchenham's standards.

2.3.3 Learning from CMM experience reports

So, with case studies with formality more or less relating to stories of catching big fish, how can an organization learn from other companies' experience with CMM, and make it relevant to its own situation? As Kitchenham puts it: "If it worked for someone else, how do you

[6] The checklist is too long (almost a whole page) to include a copy of it here. Please refer to Kitchenham's article.

know it will work for you?" [33]. A certain degree of similarity of *context* between the case study and that of the (potentially) learning organization, seems to be key.

Kitchenham *et al.* regards "experimental context as extremely important for software engineering research" [39]. Lets assume, presumably without pushing the envelope too much, that a SPI initiative in an organization can be viewed as a single "experiment", documented in an experience report, and authored by (more often than not - see 4.2.3), an in-house "researcher". Depending on the "experimental context" of the SPI experience report, it then can be assumed that the case study is of greater or lesser value to a given audience.

But how shall we measure context? What *context variables* are of importance? Kitchenham *et al.* have made no attempt to single out the "necessary" and the "optional" context variables in their article; they only mention a few examples of variables that probably are more important than others. These variables are "the industry in which products are used", "the nature of the software development organization", "the skills and experience of software staff", "the type of software products used" and "the software processes being used". Though, they emphasize that software engineering is in dire need of a *taxonomy of context*.

Partly inspired by Kitchenham, and partly influenced by what we think are important contextual variables for CMM case studies, we have developed a *tool* for measuring the context-"quality" of an experience report. The tool checks for how many of 12 selected characteristics of a CMM case story are present in a given case story, and gives weighted scores based on these characteristics. The tool is described in section 4.3, and it has to be mentioned already here that none of the case stories, when exposed to the tool, came anywhere close to being a "perfect" CMM case study.

3 Methodology

3.1 Introduction

This chapter deals with *how* the research of this thesis was committed. This includes preconditions for selection of case stories, an outline of the importance of both qualitative and quantitative methodology, and a description of issues related to the different methodologies.

3.2 The selection of case stories

An essential ingredient in any quantitative article review is having an explicit protocol for criterions of inclusion and exclusion of cases based on "quality" [40] and other parameters like availability, and place of publish. During the review, a whole lot of articles were rejected.

3.2.1 Scope

Our scope for selection was ***documented case studies of CMM-based software process improvement efforts*** in software organizations. In assembling articles we defined a protocol for selection. First, we selected a collection of *candidate* articles. Secondly, we selected a set of *valid* articles, which was a subset of the candidate articles.

3.2.2 Selection of candidate articles

The systematic selection of candidate articles was committed in two phases:

1. A search with the words "capability maturity model" was made at the INSPEC [41] article search engine at the 1st of December 2003. The search returned 439 hits. The abstracts of all matching articles were meticulously read. With the slightest suspicion of the article documenting any kind of CMM experience, the article became a "candidate" article, and ordered through the library services at the University of Oslo (if the article wasn't available online through campus subscription services).

2. Because the conference proceedings of the annual conference EuroSPI are not in INSPEC, all accepted submissions to EuroSPI dating from 1998 to 2003 (inclusive) were scanned for candidate articles.

3. Because the journal *Crosstalk* [42] has devoted itself thoroughly to CMM (it is a DOD software engineering journal), all its articles from 1992 to 2003 (inclusive) have been scanned for relevant case studies.

3.2.3 Protocol for selection of valid articles

The protocol for selection of articles evolved into the following:

1. **The primary effort of an article should be to elaborate a CMM experience**.
   Even if an article has some information that could be coded, if the delivery of the experience is second to some other issue in the article, the article is excluded. This excludes articles like [43] and [44].

2. **There is a demand for a certain substance in the elaboration on the experience**.
   Small articles describing that some company has progressed from level *x* to level *y*, are therefore removed as a candidates (for an example, see the brief notes of assessments in Crosstalk, like [45]). This also excludes, though rarely, good articles which are apparently "safe" candidates (like [46]).

3. **Scope is paramount**.
   Articles that has a scope of wildly different organizational departments, and where it is not clear which results that can be attributed to which department, the article is excluded. This unfortunately excludes interesting articles like [47] and [48]. This also excludes a number of articles that intentionally derive and synthesize experiences across scopes, for example [49].

4. **Only articles based on classic literary and textual expression is used.**
   In examining the reference list for the article [50], I found a list of case studies from an hitherto unknown conference. Here the case studies consisted mostly of *PowerPoint*-slides. Although the slides consisted of depictions of apparently viable CMM alternatives, it is inescapable that something valuable is missed when you don't attend the original presentation of the slides. Adding to this case, some recent research performed by professor Edward Tufte of Yale University controversially concludes that the way *PowerPoint* slides in general over-simplifies and obscures information, actually was a contributing factor to the tragic Columbia accident [51].

5. **Only articles with English language are used.**
   This is to ensure that other researchers can review and verify the coding of the articles (see Appendix B) without needing assistance from a translator.

6. **If an article is published several times, the original publication is used where possible.**
   When an article is published several places, the first place (chronologically) of publish is used, if possible.

3.3 On the interplay of qualitative and quantitative methodology

If one could for any reason imagine the body of CMM case stories as a an unploughed farmer's field, and the researcher's unforgiving job is to make an account of the fertility of that field - by turning the field upside down, *quantitative* analysis is probably the equivalent of ploughing that field with a fixed-size plough (the *coding categories*) and looking at what the plough might reveal. On the other hand, *qualitative* analysis would be the equivalent of meticulously sampling more or less arbitrary spots on the field with a spade, giving a handful of detailed accounts - and hoping the results would be somewhat representative for the whole field. The strengths and limitations for both quantitative and qualitative methodology hopefully become apparent from this example, but are illustrated in table 3.1.

	Quantitative	*Qualitative*
Strengths	- Get a statistical overview of a large field. - Easily uncovers critical variations to the field.	- Get a thorough, open-minded and detailed analysis of one (or a few) very small part(s) of the field.
Limitations	- Variations in the field that isn't covered by "the design of the plough" are simply omitted.	- Does not get the whole picture. Parts of the field containing critical variations might remain undiscovered.

Table 3.1: Strengths and limitations of quantitative and qualitative article reviews

The problem with CMM case studies is that they are not very good for quantitative ploughing, because they are of a textual and therefore *qualitative* nature (liable to a different interpretation by different readers). Yes, it is reasonable to assume before embarking on a study like this that some quantifiable elements must be found in "all" of the articles. For instance, "Return on Investment", "starting and ending CMM level", and maybe some condensed experiences about the initiative, useful to others ("success factors")[7]. And, by some measure of methodological doubt: Who knows what else might be hiding in there.

Making a quantitative "plough", a scientific coding tool that did a reasonable job of discovering (what later to be found) important characteristics of the field, proved then to be a typical exploratory process of "trial-and-error". The problem with this is that for every error another coding category had to be added to the tool, and the field has to be re-ploughed with the new plough, yet another time. This is no joke: during the development of the tool, a considerable amount of effort was spent re-reading and re-ploughing the first 10 or 15 case stories for additional information.

[7] The apparently random musings in this paragraph quite accurately depicts the expectations in the beginning of the study.

Not as if the field was by any standards prepared to be ploughed. Without self-pity: it was the farmer's nightmare. Every case story had to be read, line-by-line, word-by-word. No shortcuts. The process of moving the plough was thus more a process of *manually* preparing the area to be ploughed (*qualitatively* reading the case studies word-by-word – turning the whole field literally "upside down") *before* it could be quantitatively ploughed (coding the relevant sprouts of information). Ploughing was the easy part. Reading the case stories was what took all the time.

What we eventually can harvest from this part quantitative, part qualitative field, is a matter of *qualitative* interpretation. As will be showed later in this chapter, proving or disproving hypotheses using significance-levels and *t*-tests is more or less futile with the material at hand. Thus we end up with a scientific process in several stages, which uses alternating forms of methodology. There was initially a qualitative phase (reading), secondly a quantitative phase (coding), and eventually a qualitative phase again, bringing it all together. See figure 3.1, which also indicates that there is a fourth stage for analysis which compares the material not influenced by coding to the qualitative interpretations of the statistical summaries of the quantitative coding.

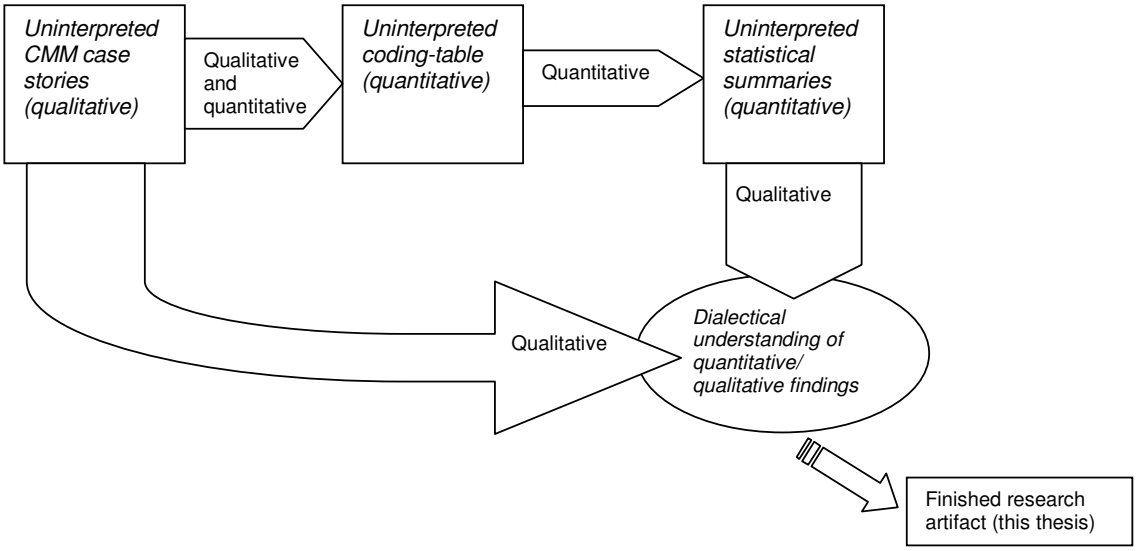


Figure 3.1: A pathway of research using both quantitative and qualitative methodology

It is amusing to deal with analogy, but metaphors (more often than not) just go "so far". I hope the analogy to ploughing a field has emphasized an understanding of that that there has been a very fundamental amount of qualitative, textual research involved in this kind of article review. This explains the corresponding amount of methodology devoted to qualitative aspects of the analysis of this review in the following section.

3.4 Qualitative methodology

3.4.1 An outline of hermeneutics

The reading of texts – any text – are in traditional humanistic sciences[8] believed to be prone to a circular and individual process of *hermeneutics* [52]. Hermeneutics is typically not referenced in software engineering. I will here try to run a case for the inclusion of hermeneutic theory in software engineering. The interesting point is that it follows from hermeneutics that every understanding of a text is a *personal* and *unique* understanding of that text. Hermeneutics argues this from the following two points:

1. A reader can *only* understand any given text based on his or hers prior *personal*, cognitive experiences in life (in hermeneutics this is called *prejudice*, or more easily: "horizon of understanding"). The prior personal experiences are thus (by definition) unique.

[8] Like the disciplines of *history* and *literature science*.

2. Any given *text* has originally been produced in a historical context different to that of the reader - in a different "horizon of understanding". It then follows that the reader is caught by his or hers *own* context and thus *cannot reach into the original context of the text*.

A further proposition of hermeneutics is that as a reader increases his or hers understanding of a text, both the text and the reader changes. Behind these seemingly supernatural statements, there is a logical explanation. The text changes, because it is now not any longer an unintelligible text, and the reader changes because the reader now has expanded his or hers personal "horizon of understanding".

But the change is not necessarily for the better. And here comes the scientific problem which *hermeneutics* brings to court: since every understanding of a text is a *personal* understanding of the text, it might not be the understanding *usually acquired within a field*. Hermeneuticians argue that the only "real" understanding of a text can be made by reading several related texts and learning about the historical context within the text was produced – immediate or a long time ago. Even so, it does not make sense to talk about a "full" understanding of a text, only a "personal" understanding.

There are a few aspects of hermeneutics immediately relevant to software engineering. The first is that it immediately relates to the question of *the validity of the results* presented in this thesis. A general approach to increase validity when coding research reviews is using multiple reviewers, as mentioned in [53]. But in the situation when material is to be coded based on qualitative assumptions (for instance: to what *degree* SEI has been involved in the authoring of an article), using multiple reviewers might not ensure a "correct" interpretation; it only ensures an interpretation that the reviewers might find probable and agree on (and that SEI, for instance, as owners of the "original context", might disagree heartily with). Here we feel that only the discipline of *hermeneutics* makes sense – a reviewers choice (for instance for what I have chosen to write in coding category #SEI – see Appendix B) has to be complemented with an account of that reviewer's *personal horizon of understanding* (this is why the section following next paragraph contains a tentative account of the development of my own "horizon of understanding").

The second aspect of hermeneutics is that it is close to the notion of *context* that Kitchenham uses [33]. Context can be said to be an important concept for two reasons; *learning* (that there is not too large a difference between the reader of the text and the author of the text) and *generalizability* (that it can be expected that somewhat similar results can be expected to be reproduced in the context of that which is described in the case study). The principles of hermeneutics pave the way for a qualitative reasoning on *why* context is important for learning. As will be demonstrated in the following paragraph, differences in "horizon of understanding" between the author and a reader, significantly influences what can be learnt from a text by a given reader. This notion is largely omitted in software engineering research, but is a very important point, especially when committing reviews of qualitative case studies (like this one): you may have the best case study in the world, but if a reader in a software company fails to understand, or worse, *misunderstands* a reasoning due to interpretational subtleties, this can in its worst consequence spell disasters for the SPI-initiative for that software company.

3.4.2 A personal account of an expanding horizon of CMM

A nice illustration of some aspects of the challenges of hermeneutics come from this authors own reading of CMM case stories. As mentioned above, a tentative account of this author's "horizon of understanding" relating to CMM can improve the understanding of certain results I have arrived at. Therefore, here goes.

The first case story I read was the account of the *Space Shuttle Onboard Software* in the original CMM (hardcover) book [22], during a software process improvement course at the University of Oslo in the autumn of 2002. But, even after reading that case story faithfully, at that point grasping *whatever* it meant that an organization could be at CMM level 5, completely eluded my mind.

During the spring of 2003 I went for several visits to one of the top assessed CMM companies in Norway, *Kongsberg Defence & Aerospace* (KDA) (they currently have a scope of 200 engineers recently[9] assessed at a "clean" CMM level 3 [54]). Joining process engineer Liv Moen for half a dozen CMM-assessments of various projects within KDA was an invaluable experience to understand more of what CMM meant *to the practitioners*. Coming back from KDA, I started reading CMM case stories with "refreshed understanding". Especially the

[9] June 2004.

mentioned *Space Shuttle Onboard Software* case study made a lot more sense (even if I still didn't quite "get it" [10]). My horizon was expanded. Previously unintelligible details concerning assessments suddenly became comprehensible.

In the spring of 2004 I ventured into another "horizon-expanding" position. I then became the leader of a group responsible for implementing several of the CMM level 2 KPAs[11] at a small (20 software engineers) internet publishing company in Oslo. There I really got "down and dirty" with the details of performing a CMM implementation. That experience *widely* opened my eyes for what tremendous amount of human effort that really is put into CBS. Whilst a concluding remark in case story like "Finally, it should be understood that what we went through is extremely stressful" [56] might for a person without a large enough horizon of understanding of CMM seem pompous at best, and self-indulgent at worst, for me, now, such a remark does *not* question the credibility of the author (and thus the case study) due to its colorfulness, but rather appears as a sincere and apt account of what I now find to be a very believable situation.

I hope to briefly have illustrated that the hermeneutical process of horizon-expansion that I have been through these last years is of paramount importance for how I qualitatively interpret the case stories in this study. I believe the hermeneutical perspective is fruitful to illustrate the differences in result and perspective we would get if different persons wrote this review; a SEI-employee, a professor of software engineering from Australia, or a student of software engineering from Norway.

3.5 Quantitative methodology

3.5.1 Introduction

The quantitative aspects of this thesis are largely dependent on the data gathered with the coding scheme. Initially the coding scheme was created with special attention to evidence of the variables "reported ROI" and "SEI involvement". A working hypothesis at the time was that there could be some kind of positive relationship between *SEI involvement* and *reported ROI*. We hoped for unveiling effects of bias (CMM is SEI's invention, so they quite possibly

[10] I can luckily say I wasn't the only one. Even within the established NASA CMM software process improvement-community, the notion of CMM level 5-organizations, and how they operated, seemed mysterious to the general community, at least until 2002, when Computer Sciences Corporation tried to settle the score at the NASA Goddard Software Engineering Workshop with their article "What Is a Level 5?" [55]

[11] More specifically: Software Project Planning (SPP), Software Project Tracking and Oversight (SPTO), and Requirements Management (RM).

don't want their own darling to look bad) or a possible "halo"-effect of having the SEI present, but our aspirations came to a halt when we found out that the number of cases reporting were ROI was present were very few, it was in fact only present in 9 out of 71 cases (see section 4.2.10).

In fact, even if our sample size is high (71 cases), most case studies have serious shortcomings in one field or another (for more on this, see the "quality tool" in section 4.3). This significantly lowers the chance of doing any "classic" correlational analysis of two variables.

Nevertheless, a coding scheme evolved, initially through persistent cycles of trial-and-error as described in section 3.1, but after the first 15 case stories or so, the coding scheme "plough" became a fixed tool, and didn't go through more changes.

3.5.2 Low-down of the coding scheme

I will here elaborate on all the coding categories. Table 3.1 illustrates the category, what it was coded as (number of variables and type of variable), whether or not it was considered a "problematic" category (especially relating to issues of interpretation), and any elaboration on the "problem" and/or additional comments are given under "elaboration".

Category	*Coded as...*	*Proble-matic?*	*Elaboration*
Title	1 Text	No	n/a
Author(s)	1 Text	No	n/a
Journal	1 Text	No	n/a
Publisher	1 Text	No	n/a
Year	1 Scalar	No	n/a
Volume	1 Text	No	n/a
Issue	1 Text	No	n/a
Pages	1 Text	No	n/a

Affiliation of author(s)	2 Ordinal	Yes	The affiliation of the author is not always stated explicitly. When in doubt, affiliation is coded as "unknown". Two variables are used to allow for a simple coding of heterogeneous affiliations of authors.
Success	1 Dichot-omous	No	Whether or not the case study proclaims to describe a successful or unsuccessful initiative.
Country	1 Ordinal	Yes	Country of scope is not always known (coded as "unknown"). In addition, in two cases ([57] and [58]), the scope is invariably linked to two countries in each of the cases. In order to minimize number of variables, the two cases were coded as separate values for (see 10.3).
Industrial sector	2 Ordinal	Yes	Industrial sector is not always known (coded as "unknown"). The important sectors *defense* and *aerospace* are coded in unique categories. This because we want to be able to distinguish between scopes of improvement that are "only" in aerospace (for example NASA [59]), and companies that are "only" in defense (for example Oklahoma City Air Logistics Center [60]) (even if most such companies usually are into both defense *and* aerospace).
Company and scope	1 Ordinal	Yes	This is the ultimate "drill-down" of unique scopes for process improvement in the organizations. Here both the name of the company and the specification of the scope are combined in one variable. Unknown scopes are coded as "unknown".

Developers in scope	1 Ordinal	Yes	The number of developers reported to be within the scope is coded in an ordinal, range-type variable ("1-4", "5-9", "10-24" etc.) Hopefully without cutting back on explanatory power, this range variable solves the problem when number of developers is actually given as a range, for example due to varying number of employees over several years (as in [55]).
Return On Investment (ROI)	1 Scalar	Yes	The ROI number is only coded when an explicit ROI is given in the article. A ROI of 7:1 is understood as "for every 1 dollar invested, 7 dollars are received in return". *Exceptions:* In [61], a ROI of "2-3 to 1" is reported. This case is coded with a ROI of 2.5:1. In [62] a ROI of 677% is reported. This is coded as a ROI of 6.77:1 (due to other numbers in the case it is possible to verify that this is what they meant to be the actual ROI).
Info on investment	1 Dichot-omous	No	This is only coded with a "yes" when the article contains some information about the investment done in relation to the improvement effort. *Important:* This variable is not coded for articles that are coded for ROI.
Info on other effects	1 Dichot-omous	No	This is only coded with a "yes" when the article contains some quantifiable information of one sort or another about the effects returned from the improvement effort. *Important:* This variable is not coded for articles that are coded for ROI.
Number of enablers	1 Ordinal	Yes	Number of CMM "enablers" or "success factors" reported in the article. An "enabler" is counted as an experience understood and reported by the author as directly contributing to the success of the improvement initiative. For more on this, see section 5.1.2.

Number of disablers	1 Ordinal	Yes	Number of CMM "disablers" reported in the article. A "disabler" is counted as an experience understood and reported by the author as directly contributing to the failure of the improvement initiative. For more on this, see section 5.1.6.
Methodology	1 Ordinal	No	Where reported in the article, this variable contains information about the methodology used for implementing the improvement initiative.
Parallel standards	1 Ordinal	No	Where reported in the article, this variable contains information about any parallel standards (like ISO 9001 [63]) present in the organization prior to, or along with, the improvement initiative.
Starting level	1 Ordinal	Yes	Where starting level is not given, and there is no reason to believe in other respects from the interpretation of the text that the case described is not an effort beginning at level 1, the level is assumed to be 1. If it can not safely be assumed to be level 1, it is coded as "unknown". If the author emphasizes that they are partially fulfilling the demands for a certain level, for instance level 2, the level is coded with a decimal, like 1.5.
Ending level	1 Ordinal	Yes	If ending level is omitted, it is coded as "unknown". If the authors emphasize that they are partially fulfilling the demands for a certain level, for instance level 3, the level is coded with a decimal, like 2.5.
Starting year	1 Ordinal	No	If given, the starting year of the initiative.
Ending year	1 Ordinal	No	If given, the ending year of the initiative.
Number of months	1 Ordinal	No	If an explicit number of months were given, these are coded here. *Note:* No calculations are done to derive number of months from starting and ending year.

Sponsor	1 Ordinal	No	If the article mentions an explicit sponsor of the article, it is coded.
SEI involvement	1 Ordinal	Yes	Codes whether SEI is involved or not. Coded with one of four values: "no indication", "suspected", "highly probable" and "explicitly mentioned". Subject to personal bias.
Advice from level 1 to 2?	1 Dichot-omous	Yes	Codes whether or not the article includes explicit advice for going from level 1 to level 2. This is coded as true for every article that has an ending level of 2, and has enablers.

Table 3.2: The coding scheme

3.5.3 Descriptive statistics

In section 4.2 we describe the most relevant of the variables. Different kinds of charts are used, combined with tables where explicit numbers are of importance. After we have presented the data, we make comments on the data. If we deem it appropriate, the data is contrasted with other data from other sources to make them more intelligible.

3.5.4 Representativeness of results

As pointed out in the introduction to this chapter, there is a rather high sample size (71 cases) involved in this article review. Prospects for representativeness should be good, had this been a typical "random sample" used in statistics. But it's not. As mentioned in the definition of protocol (see section 3.2), we have actually tried to include *the whole universe* of known CMM case studies adhering to the protocol in our review.

The question we should ask is whether the "universe" of case studies of CBS efforts (71 cases) is representative for the "universe" of all CBS efforts (several thousand cases, according to [28]). Unfortunately, it's not, and to make it worse, it's not anywhere close. As we will later point to, the defense and aerospace industry is grossly overrepresented in the case studies (see section 4.2.4), U.S. companies are by any account overrepresented (see section 4.2.7), small companies are vastly underrepresented (see section 4.2.9) and last, but not least, the microscopic failure rate in the case studies of 2.8% (see section 4.2.6) is by any

account (even if we don't have any "hard" evidence, as pointed out in section 6.2) most likely attributed to the assumption that those who are "successful" in CBS are more likely to write case studies, than those that are unsuccessful. Exactly the lacking representativeness of CMM case studies is thought to be a fundamental problem when embarking on CBS.

4 Quantitative analysis

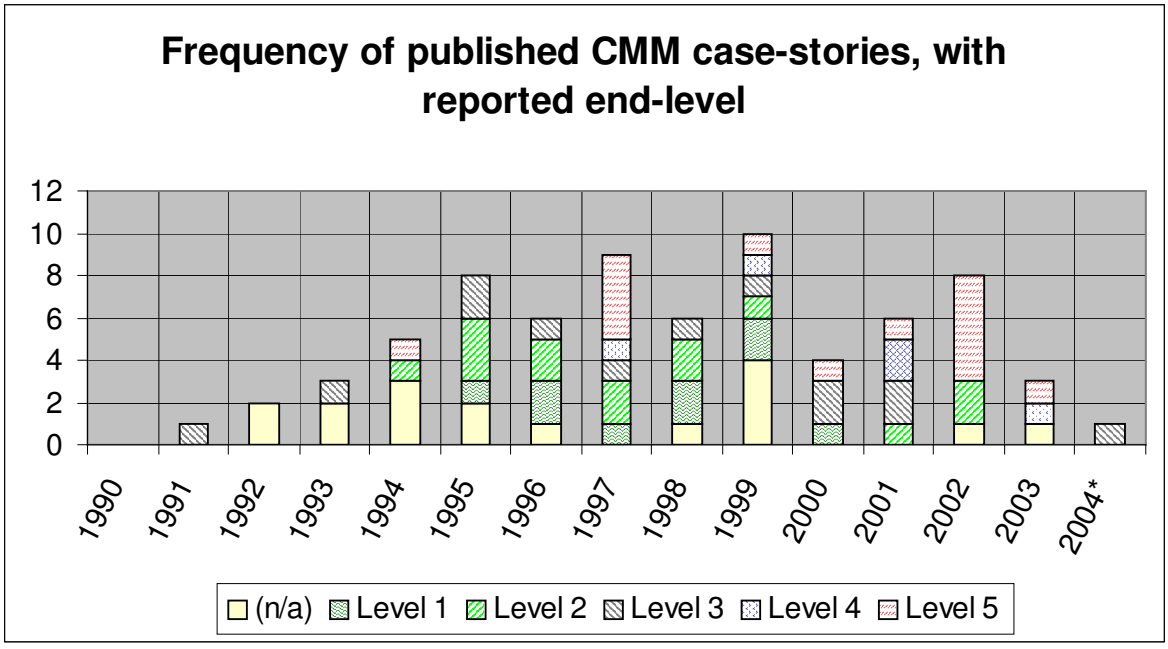
4.1 Introduction

In 4.2, we do a presentation of selected dimensions of the quantitative data gathered in the coding scheme. The data is commented as appropriate. In 4.3, we develop and present the results of a self-made algorithm for a "quality tool" applied to the case studies.

4.2 Descriptive statistics

4.2.1 A historical review of case studies

In figure 4.1 we illustrate the distribution of CMM case studies over the years. Included in the figure is the reported ending level for the case study (if any).



** = not included in this review due to the protocol of selection*

Figure 4.1: The rise and fall of CMM case stories

As mentioned in section 3.2.2, the initial search for selection of articles was committed on the 1st of December 2003. As of the 22nd of June 2004, there are no registered CMM case stories for 2004 in INSPEC's databases. Somewhat compensating for the delay from publication to enrolment in INSPEC, we conducted complementing manual searches in *IEEE Xplore* [64] and *ACM Digital Library* [65]. Only one case study [66] (reporting an ending level of 3) has

been found for 2004[12], and that had been published in *IEEE Software*. *This case study is not included in the review.*

4.2.1.1 Comment

The popularity of CMM case-stories seems statistically to be the fluke of a decade. The seminal[13] *Software Process Improvement at Hughes-Aircraft* in 1991 from the hands of Watts Humphrey [68], sets the stage with tempting prospects for CMM profitability, obviously inspiring the software community. The future looks bright, and the "golden era" of CMM case stories is literally launched when both the article describing the process efforts at the *NASA Space Shuttle Onboard Software* project [59] and a printed, hardcover version of the CMM v.1.1 [22] was released (which incidentally also contained a version of the *NASA* article). The NASA Space Shuttle case story was the first article describing a supposedly true CMM level 5 organization, indicating that this kind of "software nirvana" actually was possible to reach.

From 1995 a purportedly puzzling phenomena emerges: case stories are published by organizations that didn't reach any level at all (they are still level 1-organisations at the time of writing the case story). And not only that – the organizations almost exclusively claim their SPI efforts to be *successes* (see section 4.2.6). Why is this happening? There are several, more or less plausible, explanations emerging: a) authors are more or less desperate to publish ("getting to that Hawaii conference"), b) it's a bit harder getting to level 2 than initially planned for, and/or c) the *publishers* see a certain value (not discussing which) in case-stories coming from companies that in CMM-terms "haven't done much" yet.

The "golden era", with its wide variety of case-stories, lasts until 2002. The frequency of published papers suddenly drops markedly in 2003 and only one published paper is recorded for 2004 so far (as of July 22nd).

4.2.1.2 Why the decline in case stories?

The decline of interest in the SW-CMM has several possible reasons. We present three of them here:

[12] Not found in INSPEC as of July 22nd 2004.
[13] *ISI Web of Knowledge* contains a database of citations for articles from selected journals. In a search conducted the 26th of June 2004, the "Hughes Aircraft"-case story had been recorded cited in 30 other articles carried by *ISI* [67].

4.2.1.2.1 "Sunsetting" of the CMM

In 2001 SEI announced the successor of CMM, CMM-I [69], which consists of some changes and additions to the SW-CMM, as well as integration of a host of other maturity models as well (Systems Engineering CMM [SE-CMM], Integrated Product and Process Development [IPPD], amongst others). SEI decided in December 2001 to slowly cease supporting (they call it "sunsetting") the SW-CMM [70], making CMM-I the new "kid" on the block.

Since CMM-I is the proposed successor for CMM, we will devote it some space. It is healthy to be skeptical about CMM-I. SEI's forceful approach to pull the plug on the SW-CMM may currently just prove a hard thing to do, for the following proposed reasons:

- The business driver (ROI) for CMM-I is totally in the blue. If the ROI-evidence on the SW-CMM may at first hand seem meager and circumstantial (see section 6.2.2), for the CMM-I there hardly exists research *at all*.

- After two years of CMM-I, very few organizations have been assessed at a CMM-I level [71] compared to the number of organizations already assessed with SW-CMM [28].

- The CMM-I is considerably larger and more complex (counting number of pages and key practices) than the SW-CMM, and the question remains whether an even more complex maturity model is the solution for an industry that is already supposedly struggling with existing frameworks like the CMM.

- Introducing a SPICE-inspired *continuous* approach to process implementation, the CMM-I lacks the (suspected) appeal of a clear and "leveled" approach to process improvement, which the SW-CMM offers.

That said, CMM-I has potentially very useful elements that fills obvious gaps in the SW-CMM, for instance the very basic "Requirements Developments" process area. Maybe it might just be these kinds of "mini-recipes" from this massive framework that will let (at least elements of) CMM-I gain foothold in SPI-curious organizations worldwide? Anyway, further discussion of this question is out of scope for this thesis.

"Continuous improvement" (in one interpretation or another) is the *sine qua non* of software process improvement, and thus it could be thought that the SW-CMM for organizations actually will be *morphing* into CMM-I in their commitment to improve. Time will show.

4.2.1.2.2 Saturation of case studies

Another explanation for the decline of CMM case studies is that there simply is believed to be a "saturation of case studies". It can be supposed that there is a point where the community (represented by the editors of journals and conferences) feels that the educative value of "yet another CMM case story" has an increasingly lowered value for its readership.

4.2.1.2.3 The agile "flare"

It's no secret that fashions and the shifting lures of the *Zeitgeist* are a fundamental element of the software engineering community. Still pursuing the "silver bullet" around the next corner (Brooks [7] has told us that it doesn't exist, no one listens), practitioners and researchers alike are prone to jump on any given bandwagon as long as it serves their professional and/or financial interest. Charting the terrain of "agile" methodologies (like XP [9] and SCRUM [72]) has been given lots of attention both in industry and academia the last few years, on the cost of process-related issues. Software process improvement's "own" journal, *Software Process: Improvement and Practice* [73], rose and flourished during the golden era of CMM case stories, but was (symbolically) discontinued late 2002.

4.2.1.3 Increased importance: the consequence of the decline of published case stories

As mentioned, the maturity profile of SEI indicates a steady increase of organizations assessing themselves as continually higher levels. If this trend will continue or plateau, remains to be seen. But since the number of case stories is becoming fewer and fewer relative to the organizations embarking on their CMM journey, their importance, and what knowledge that can be derived from them, steadily increases in value.

4.2.2 Prevalence of SEI involvement in case studies

4.2.2.1 Presentation of data

	Freq.	*Percent*
No indication of SEI involvement whatsoever	53	75%
Suspected SEI involvement - but only non-conclusive evidence	5	7%
SEI involvement by a high degree of likelihood	2	3%
SEI involvement - explicit mention in article	11	15%
Total	*71*	*100%*

Table 4.1: Prevalence of SEI involvement in case studies

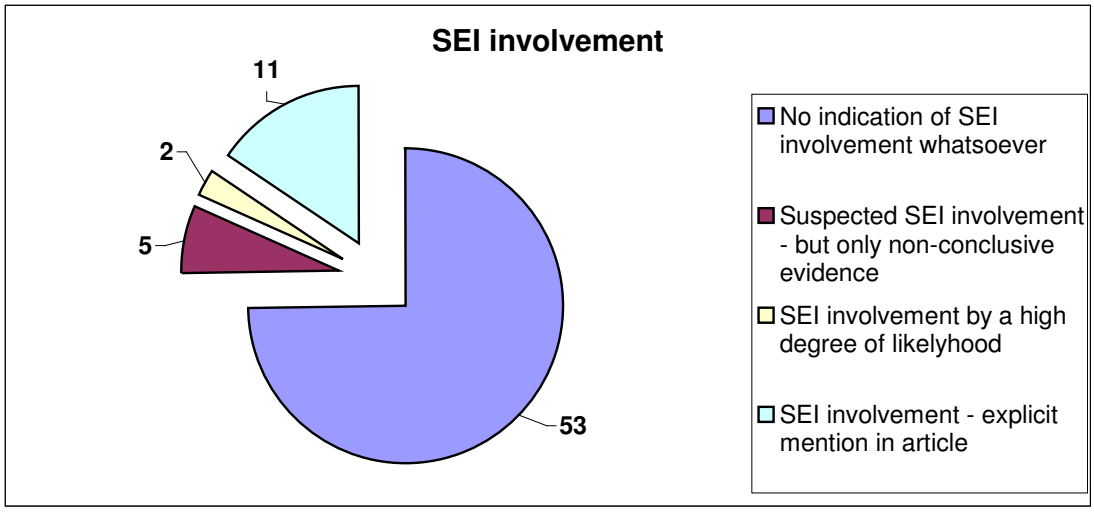


Figure 4.2: Pie chart of SEI involvement in case studies

4.2.2.2 Comment

The *Software Engineering Institute* (SEI) is the cradle of CMM (for a short historical account, see 2.2.3). It would be interesting to see how many of the CMM case studies that have links to the SEI. As we can see, these articles are in the minority. 75% (53) of the articles have no indication of SEI involvement whatsoever. Only 15% (11) of the articles have an explicit connection to SEI.

4.2.3 Affiliation of case study authors

4.2.3.1 Presentation of data

Note on coding: "Author affiliation" is a variable that can have multiple (two) responses because there can be multiple authors, and when this is the case, a typical scenario is that one of the authors are from the same company whilst the other author is an external consultant, researcher or whatever. Because of this, number of coded responses is higher (n=86) than the number of cases (n=71) for this variable.

	Responses	*Percent*
From same company	56	65%
External – formerly employed in company	4	5%
External to company (unspecified)	3	3%
University or other external research facility employee(s)	9	10%
Journal/magazine editorial staff	1	1%
External consultant(s)	7	8%
Employees of SEI or Carnegie Mellon University	3	3%
(unknown affiliation)	3	3%
Total	*86*	*100%*

Table 4.2: Affiliation of case study authors

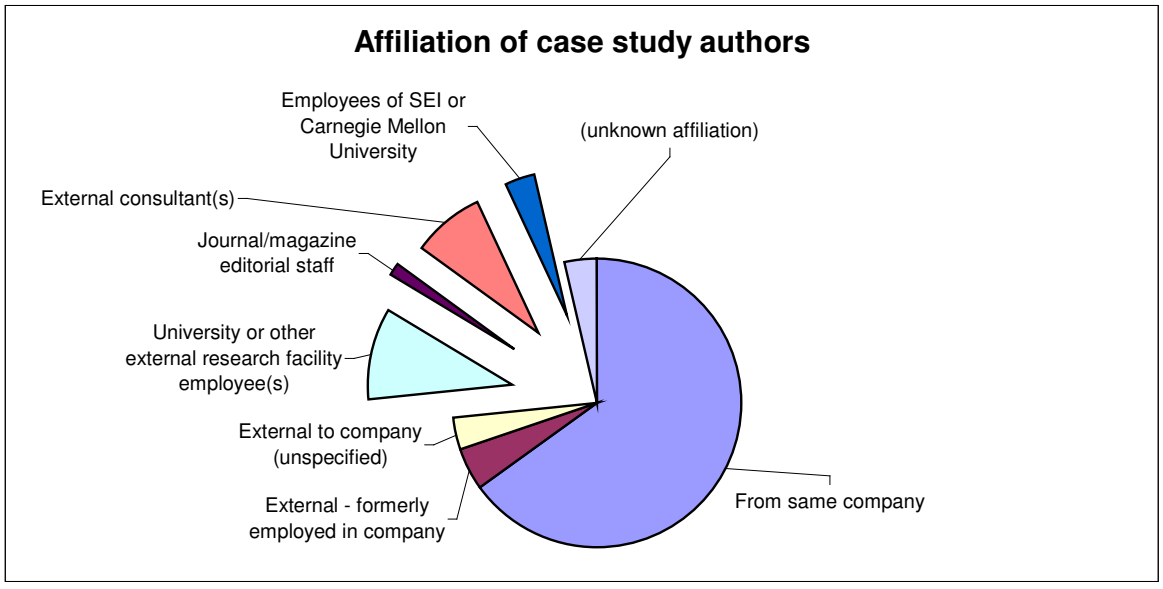


Figure 4.3: "Writing their own stories" – pie chart of case study author affiliation

4.2.3.2 Comment

We see that 65% of the authors to the case stories are from within the same company as where the improvement initiative is present. But we can with all likelihood add to that number. It is probable that the authors of "unknown affiliation", even though their affiliation is not stated, have a high degree of affiliation to the company of improvement. It is also likely that the group of authors that are "unspecified" external to the company have some degree of affiliation to the company of improvement. Then, in total, as many as 75% of the authors are likely to have some or another in-house connection to the company.

4.2.4 Industry sector affiliation

4.2.4.1 Presentation of data

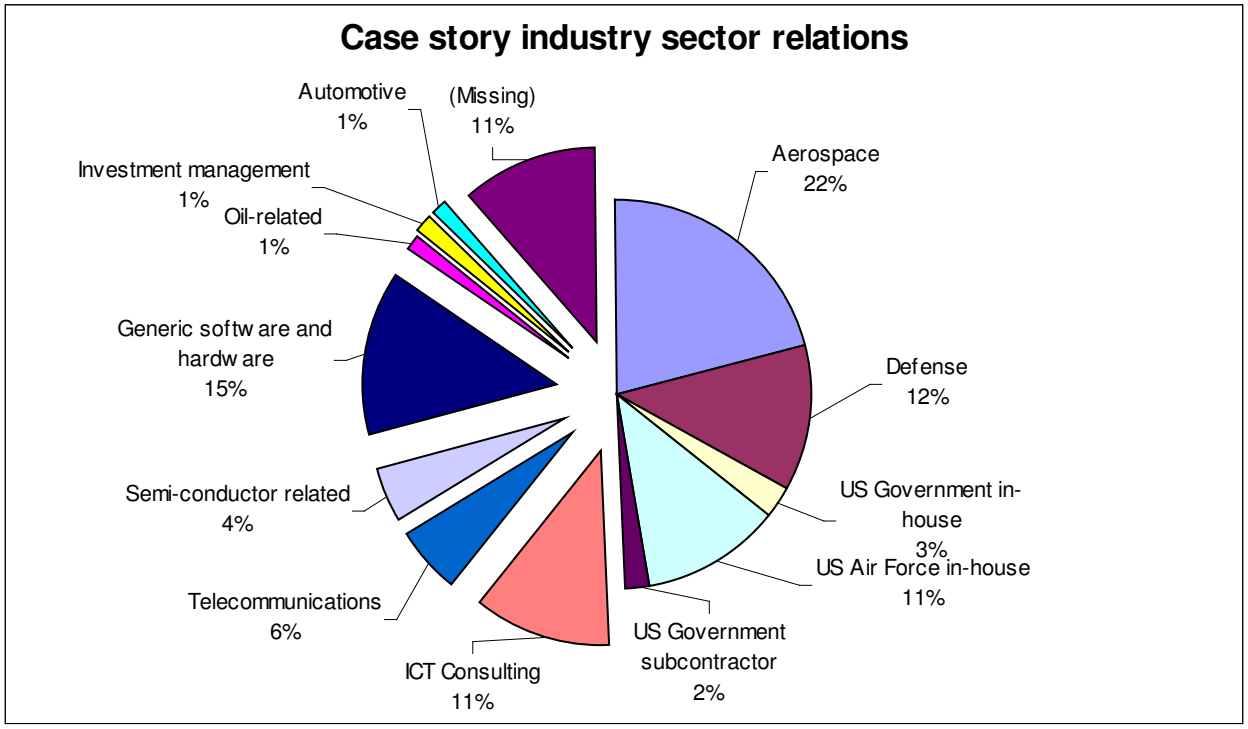


Figure 4.4: Pie chart of industry sector affiliation

4.2.4.2 Comment

The pie chart shows that accounts of CMM in the study have a fifty-fifty likelihood of coming from in-house departments of, or companies being subcontractors to, U.S. Government and Defense. Comparing these numbers to SEI's most recent Maturity Profile for the CMM [28], we see a gross over-representation for these industries in the case stories (see figure 4.5):

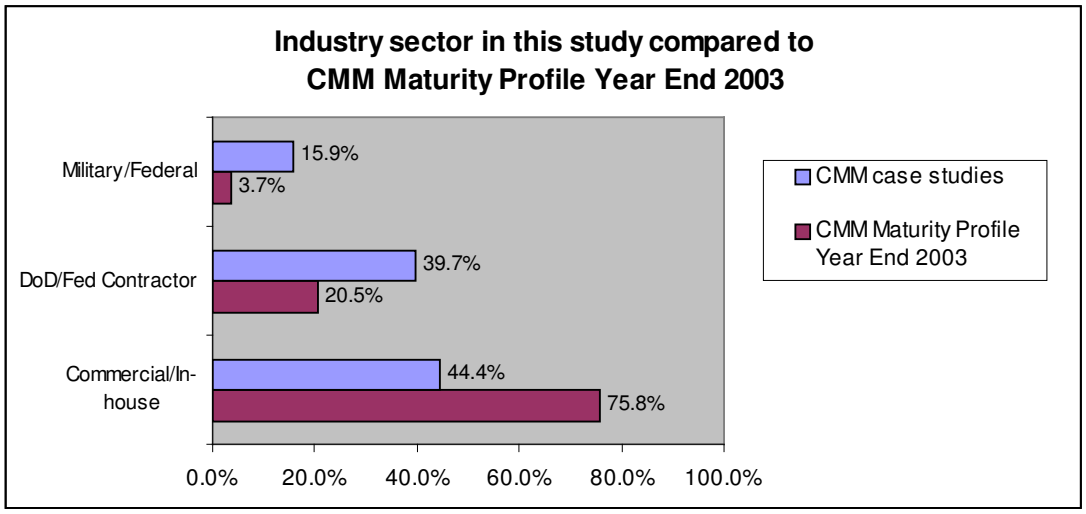


Figure 4.5: Industry sector in study compared to CMM Maturity Profile

To be able to create figure 4.5, we did the following groupings: *Military/Federal* is comprised of our categories *U.S. Government in-house*, *U.S. Air Force in-house*. *DoD/Fed Contractor* is comprised of *U.S. Government subcontractor* and *Aerospace* and *Defense*. *Commercial/In-house* is the rest of the groupings.

The maturity profile represents self-reported information from companies or the companies' assessors (no randomized sample, neither a total view of the population), so it's not of scientific validity, but provides a picture of something that probably could be representative.

4.2.5 Publishing location for case study

4.2.5.1 Presentation of data

Place of publish	*Freq.*
Conference proceedings	27
Crosstalk	20
IEEE Software	11
Company-linked technical journal	4
Software Process Improvement and Practice	2
IEEE Transactions on Software Engineering	1
IEEE Computer	1
Communications of the ACM	1
Software Quality Journal	1
Management Science	1
Software Process Newsletter	1
American Programmer	1
Total	71

Table 4.3: Publishing location for case study

4.2.5.2 Comment

As we see, most of the case studies come from conference proceedings. Of the journals, *Crosstalk* hovers above the rest with no less than 20 case studies. As almost all of the case studies can be traced to defense and aerospace companies (after all, *Crosstalk* is an U.S. Air Force journal), the results found in section 4.2.4 above would be somewhat more representative if *Crosstalk* was omitted (it's probably not considered a "scientific" journal, but does – in general – not provide with significantly lesser-quality case studies than the other sources (see Appendix B)). After *Crosstalk, IEEE Software* has devoted considerable space to CMM case studies. On fourth place are various studies published in company-proprietary journals, whilst the only journal dedicated to SPI (*Software Process Improvement and Practice*), surprises with as few as only two case studies.

When it comes to accessibility, it should be noted that it's paradoxically only the *defense*-related *Crosstalk* that is open to the public. The rest of the journals and conference proceedings require subscription or some other means of special access. This is a problem, because the only free case studies the least resourceful software companies (which obviously aren't in defense) get to read on the internet, are the case studies from the defense industry.

This might result in (with an asymmetry in information) *a)* the possible impression that CBS is only or mostly done in the defense sector, and *b)* learning from a wrong context.

4.2.6 Success and failure for published case studies

4.2.6.1 Presentation of data

Success?	*Freq.*	*Percent*
Success	69	97.2%
Non-success	2	2.8%
Total	*71*	*100%*

Table 4.4: Reported success and failure

4.2.6.2 Comment

What we can derive from these numbers is that, for sure, there is a large incentive to write about (allegedly) successful attempts on CBS. Accounting for *what went wrong* when failing an initiative, does not seem to be rather interesting. We assume that the failure rate indicated here, 2.8%, is by no means representative for the universe of attempted CBS. But it could be argued that "only reading good news" of CBS (97.2% of the cases) might somehow influence the number of organizations that attempt CBS in a positive way – given that the failure rate is "in the blue" (see section 2.2.4).

4.2.7 Country of scope for improvement

4.2.7.1 Presentation of data

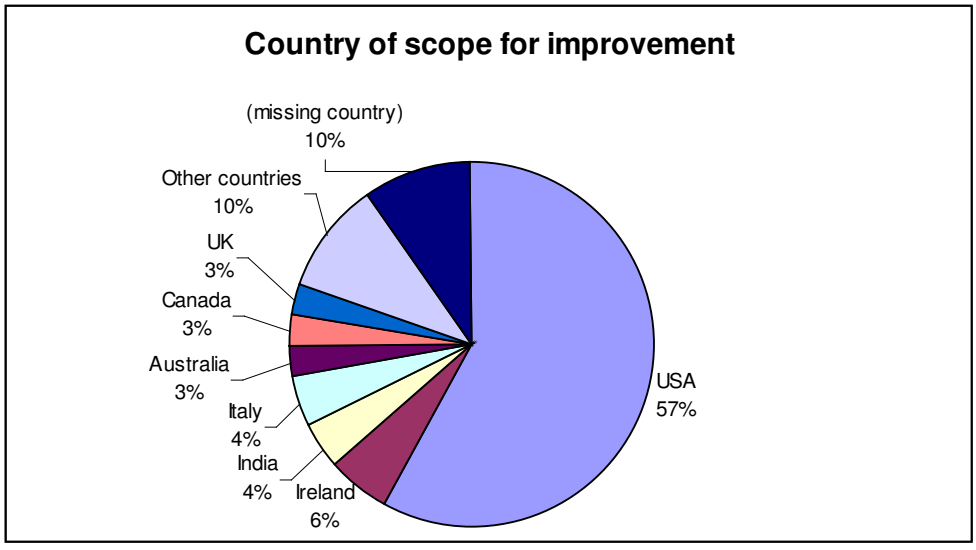


Figure 4.6: Country of scope of improvement

Country	*Freq.*	*Percent*
USA	41	58%
Ireland	4	6%
India	3	4%
Italy	3	4%
Australia	2	3%
Canada	2	3%
UK	2	3%
Other countries (Brazil, Croatia, Germany, Japan, The Netherlands, USA & India, USA & Israel)	7	10%
Total	*64*	*90%*
(Missing country)	7	10%
N	71	100%

Table 4.5: Country of scope of improvement

4.2.7.2 Comment

As we can see, writing case studies of CBS is largely an American sport. Scopes for improvement in U.S. companies are responsible for at least 58% of the case studies in this review – maybe even more, accounting for that some of the case studies with "missing" country probably are of U.S. origin as well.

4.2.8 Featured company scopes

4.2.8.1 Presentation of data

Name of company and scope	*Frequency*	*References*
Ogden Air Logistics Center SED	4	[74] [75] [76] [77]
Boeing Defense and Space Group - Space Transportation System	4	[78] [79] [80] [81]
Oklahoma City Air Logistics Center SED	3	[60] [82] [83]
SEAS / Computer Sciences Corporation	2	[84] [55]
Motorola Cork	2	[85] [86]
Motorola Cellular Department	1	[87]
Hughes Aircraft SED	1	[68]
Atos Origin India	1	[38]
S3 Silicon and Software Systems	1	[88]
Motorola Government Electronics Division	1	[62]
Raytheon Electronic Systems	1	[89]
Sodalia developer department	1	[90]
Raytheon Software systems laboratory	1	[91]
Sacramento Air Logistics Center SED	1	[56]
North American Aerospace Defense - System Support Facility	1	[92]
U.S. Department of Agriculture - National Finance Center	1	[93]
General Dynamixs Decision System SED	1	[94]

Naval Air Systems Command - AV-8B Joint System Support Activ	1	[95]
Boeing Military Aircraft and Missiles - Seattle Site	1	[96]
Harris Corporation - Government Communications Systems Div	1	[97]
Unisys - Australian Center for Unisys Software (ACUS)	1	[98]
Thomas Jefferson National Accelerator - Controls Group	1	[99]
COLSA Corporation - Software development departments	1	[100]
CELEPAR - All software employees	1	[101]
Texas Instruments Test Engineering Department ...	1	[31]
Philips Electronics - Philips TV division	1	[102]
Corning Incorporated - Information Services Division	1	[103]
Advanced Information Services - (organization-wide)	1	[104]
Litton PRC - Systems Integration Unit (and more)	1	[105]
Unisys GDG - New Shipborn Aircraft Program	1	[106]
NASA - Space Shuttle Onboard Software Project	1	[59]
Oerlikon Aerospace - Laser-guided missile-air defence system	1	[107]
AIM Management Group Inc - Telecomm. and Systems dep.	1	[108]
Bull Information Systems - (organization-wide)	1	[109]
Texas Instruments - Defense Systems and Electronics Group	1	[110]
NEC – Communication Systems	1	[111]
Bombardier Aerospace Group - Short Brothers ICAD project	1	[112]
Ericsson Nikola Tesla - Software Design Centre	1	[113]
Tektronix, Inc. - Product instrumentation group ...	1	[114]
Method Park Software AG - Software development department	1	[115]
Hewlett Packard - "One HP division"	1	[116]
Silicon and Software Systems - Software department	1	[117]
Total	52	
Missing scope and/or company	19	[61] [118] [58] [119] [35] [120] [36] [121] [122] [57] [123] [124] [125] [126] [127] [37] [128] [34] [129]
Grand total	*71*	

Table 4.6: Featured scopes of improvement

4.2.8.2 Comment

Most notable with this table is probably that some of the scopes for improvement have been mentioned in more than one case study. Ogden Air Logistics Center (incidentally the same military base as the one *Crosstalk* [42] is published from) tops the statistics together with the Space Transportation System scope of Boeing. Another air-force base is at third place.

There are both positive and negative sides to the fact that a scope is talked about several times in the review. On the positive side, readers potentially get to know a scope better, and have a better understanding of what kinds of experiences that can be applicable to their own situation. On the negative side, one scope being mentioned several times apparently skews the statistical results of the review somewhat.

Still, we have chosen *not* to work the rest of the statistics in this chapter by "filtering" case studies, so that we apparently end up with a 1:1-relationship with scopes. The rationale for this is threefold. First, we would still have no ways of checking for "duplicates" in the case studies that are *lacking* scope and/or company. Secondly, if we imagine that the typical reader and user of case stories more or less "by chance" stumbles into case studies (more than the reader makes a systematic review of *all* case studies in the literature), "any" case study that the reader could run into "should" be represented in this review. Thirdly, as pointed out in section 3.5.4, we have no illusions of acquiring any more representativeness for this review than it being an account of all case studies of CBS adhering to our protocol. Our unit for study is *case studies*, not improvement scopes.

4.2.9 Number of developers in scope

4.2.9.1 Presentation of data

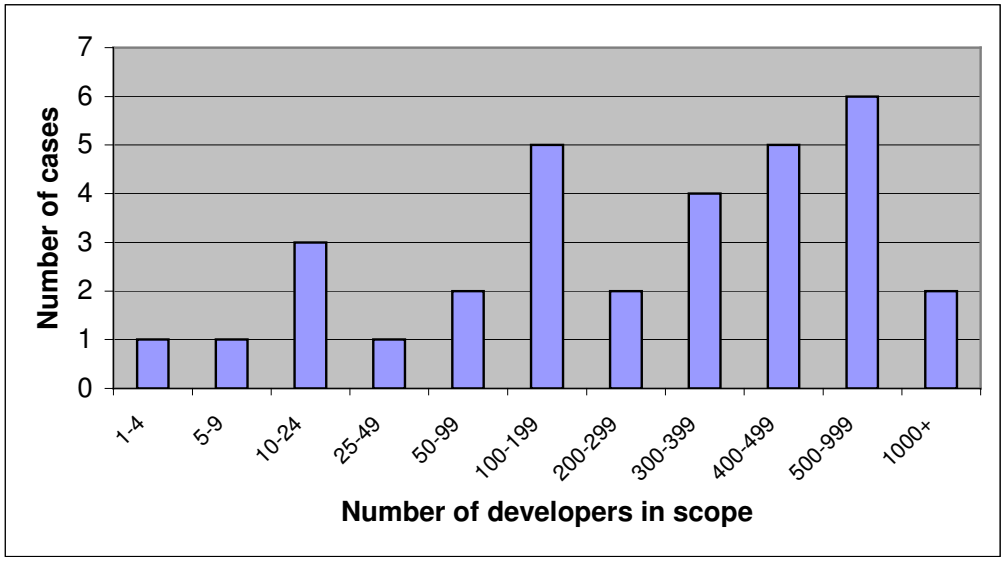


Figure 4.7: Number of developers in scope

4.2.9.2 Comment

The information presented is from a total of 32 cases. The remaining 39 cases are missing information about the number of developers in scope. We can see that the vast majority of cases (75%) where scope is reported are case stories with between 100 and 1000+ developers in scope. Only 16% of the cases explicitly describe scopes with less than 25 developers.

Tore Dybå indicated that the general approach towards SPI *differs* when comparing small and large software organizations [130]. It would then be reasonable to believe that small

organizations need case stories by small organizations, whilst large organizations need case-stories by large organizations. A count for 1995, cited in [131], indicates that approximately 49%[14] of all software developers working in the U.S., work for a company with less than 100 employees. This accounts for 97.2% of all U.S. software companies – actually only 2.8%[15] of the companies have more than 100 employees!

Assuming the situation worldwide is more or less as in the U.S., the conclusion from this is that only 25% of the case stories are aimed on 97.2% of the companies. This is a skewed distribution, by any standard. We need more case studies to alleviate this situation.

4.2.10 Self-proclaimed Return On Investment (ROI)

4.2.10.1 Presentation of data

ROI (x : 1)	*Freq.*	*Percent*	*Valid Percent*
2.5	1	1.4%	11.1%
6	1	1.4%	11.1%
6.35	1	1.4%	11.1%
6.77	1	1.4%	11.1%
7.5	1	1.4%	11.1%
7.7	1	1.4%	11.1%
7.75	1	1.4%	11.1%
9	1	1.4%	11.1%
19	1	1.4%	11.1%
Total	*9*	*12.7%*	*100%*
Missing	62	87.3%	
Grand Total	71	100%	

Table 4.7: Distribution of self-proclaimed ROI

[14] 49% is calculated from 549333 out of a total of 1117475 employees.
[15] 2.8% is calculated from 2089 out of a total of 75172 companies.

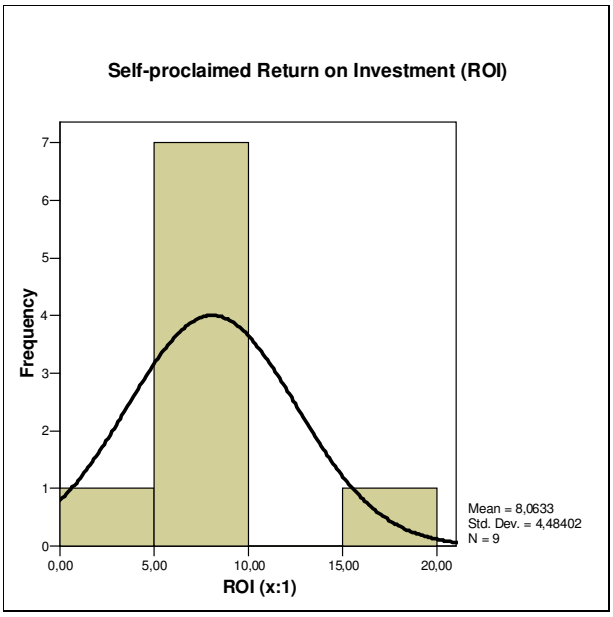


Figure 4.8: Distribution of self-proclaimed ROI

4.2.10.2 Comment

Only 9 case studies has reported a ROI (62 cases have no ROI calculated), whilst the distribution is wide; from 2.5 to 19. This is significant because of the low number of cases. Still, we see a rather peculiar, counting the low sample and high spread, clustering of values at around 7. The median of 7.5 thus gives a better view of the central clustering than the average of 8.1.

One thing is clear for sure: the numbers of calculated ROI indicate an extremely high economic payoff for the investment. Even the lowest number, an ROI of 2.5 to 1, sure beats putting money in the bank. To put it bluntly, if the number of an average ROI of 8.1:1 was correct and possible to calculate with (why this is not possible, though, see section 6.2), it would be economic lunacy *not* to start a SPI initiative.

If we're comparing the articles with calculated ROI to industry sector, we find that out of 9 (100%) articles, 7 (77.8%) articles come from defense and aerospace-related scopes of development (counting in-house U.S. Air Force software organizations), and 1 (11.1%) article comes from civilian industry scope (Hewlett Packard [116], to be precise). 1 (11.1%) article has an unknown scope. Civilian industry sector is thus again grossly underrepresented in reporting ROI.

4.2.11 Not saying it with ROI vs. not saying anything at all

4.2.11.1 Presentation of data

Number of cases		Info on investment		
		yes	*no*	Total
Info on effect	*yes*	**7**	18	25
	no	3	**34**	37
	Total	*10*	*52*	***62***

Table 4.8: Cross-tabulation of info on investment and effects

4.2.11.2 Comment

What about the case studies that haven't calculated ROI, what do those studies got to offer? We have coded every article for information on *investment* and *effect*, and the results are presented in table 4.8.

For the cases that haven't reported a ROI (n=62), more than half of them (34) does not care to report *any* information relating to either investment or effect of the software process improvement initiative. 3 of the cases have information on the investments committed, but no viable information on effects. Moving further up, 18 of the cases have no information on investment, *but* has given viable indication on the effects of the initiative. "Best of class" are 7 cases that has both information on investment and effect, but no calculated ROI.

It can be argued that giving information about the investment, without giving us information about the effect, is rather hopeless to learn from for a company that wants to calculate the benefits of software process improvement (only three of the cases were in this condition). On the contrary, giving us information about the effect without giving information about the investment might be of some better use – at least some indication of the benefits can be derived (18 of the cases).

Giving information *both* of investment and effects is definitively something that makes more sense than its alternatives. Sitting with information on both sides – why didn't these companies have a go at calculating ROI because of it's potential dazzling effects (if you get some of the scores as reported in section 4.2.10, of course) on management for justifying the investment made on improvement? Of course, there *might* be the answer that it is just laziness – management don't need any more justifications, and thoughts like "we don't need to brag to others about ROI in a case study" or whatnot. A more disturbing alternative is that the company tries to "mellow" the fact that the actual ROI they tried to calculate was very

low. Finally, an option is that the company has taken a responsible stance towards calculation of ROI: ROI is so hard to calculate for SPI, that they haven't ventured into it (see section 6.2.3 for why this might be a responsible stance).

4.2.12 Enablers and disablers

4.2.12.1 Presentation of data

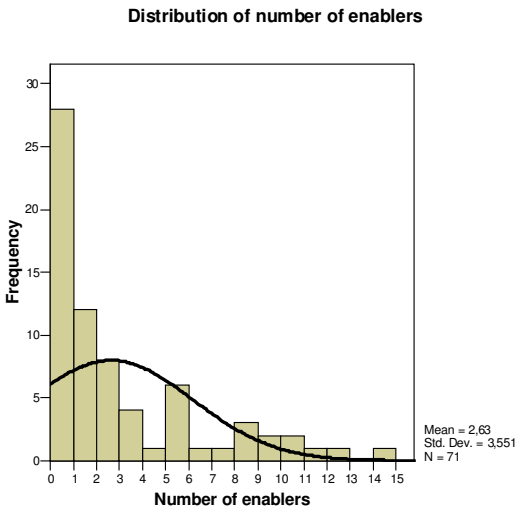


Figure 4.9: Distribution of number of enablers

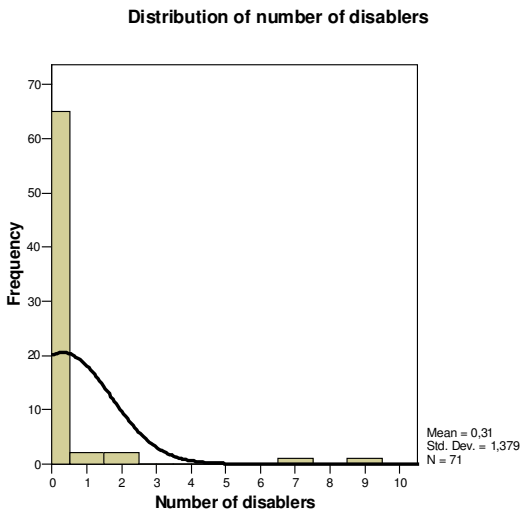


Figure 4.10: Distribution of number of diablers

4.2.12.2 Comment

The figures 4.9 and 4.10 are based on respectively 187 enablers and 22 disablers. As for the enablers, quite a few case studies have up to 10 explicit success factors mentioned. Only 28 articles have no success factors mentioned at all.

On the flipside, only 6 case studies mention factors that explicitly contribute to an initiative's demise. Of these 6 case studies, only 1 case study is a case study where the initiative was deemed as a "failure". We believe that this is no problem, and that with long-lasting processes like SPI, most successful initiatives are likely to experience both setbacks and successes.

4.2.13 Parallel standards and frameworks

4.2.13.1 Presentation of data

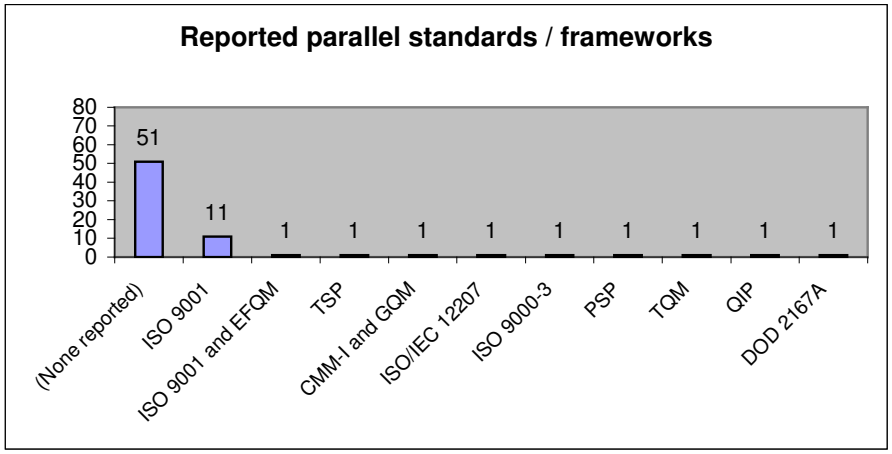


Figure 4.11: Parallel standards and frameworks

4.2.13.2 Comment

We wanted to have a look at co-existing quality frameworks and standards mentioned in the case studies. The recognized ISO 9001 is mentioned in 11 of the case stories, and is by far the largest "sidekick" for the CMM. In addition, the following standards and frameworks are mentioned, each within a single case study: Team Software Process (TSP; by SEI), CMM-I and Goal Question Metric, ISO/IEC 12207, ISO 9000-3, Personal Software Process (PSP; by SEI) and DOD 2167A. References to all of the standards are readily available in the *Frameworks Quagmire* [63]. Not in the quagmire is the general Deming-inspired *Plan-Do-Check-Act* (PDCA) cycle of Total Quality Management (TQM) [17], the EFQM Excellence model [132], and Basili's Quality Improvement Paradigm (QIP) [133].

4.2.14 Ending CMM level for case stories

4.2.14.1 Presentation of data

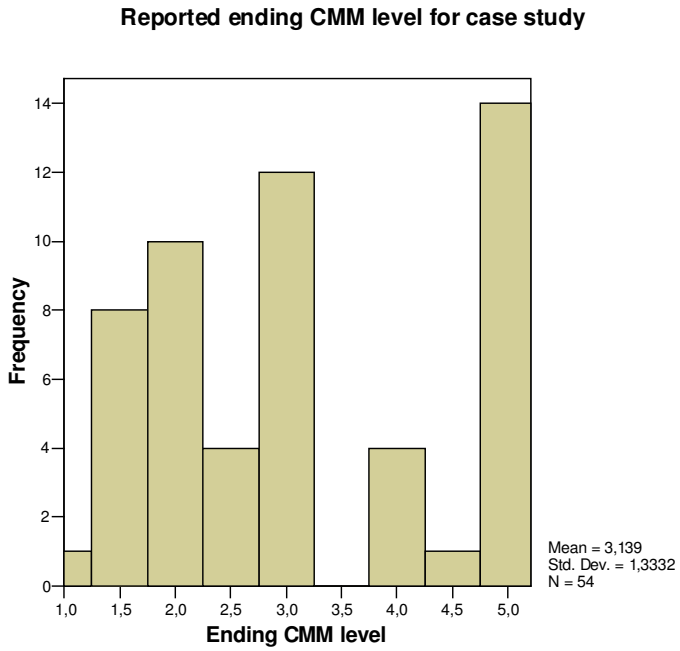


Figure 4.12: Reported ending CMM level for case study

4.2.14.2 Comment

As we see from figure 4.12, from those case studies that have reported an "ending level" (54 cases), level 5 is the most "reported" level (not including "half levels"[16]). Still, the majority of case studies are between level 1 and 3.5. This is healthy, because getting to know organizations that have climbed the first (few) level(s) of CMM is nice for gathering information about CMM in preparing embarking on CBS. Of course, level 5 case studies is overrepresented compared to the relative number of reported level 5 organizations in the world [28], but this slight skewing of distribution is not regarded as a problem, exactly because of the solid number of organizations giving case studies from the lower levels.

[16] As pointed out earlier, in section 3.5.2 (the coding scheme), half levels are used if the authors of the article emphasize that they are partially fulfilling the demands for a certain level. For instance, explicitly partial fulfilment of level 2 would be coded as level 1.5.

4.2.15 Number of years per CMM level

4.2.15.1 Presentation of data

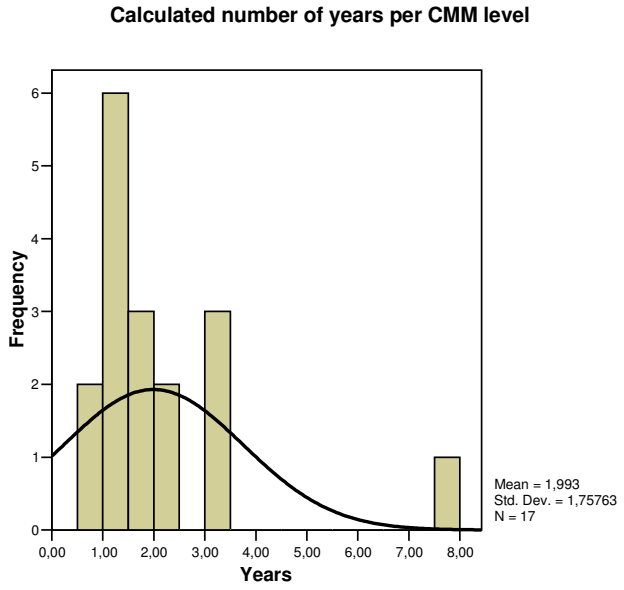


Figure 4.13: Calculated number of years per CMM level

4.2.15.2 Comment

The variable *yearsRatio* was created by performing the following calculation on the data:

$$yearsRatio = \frac{(endYear - startYear)}{(endLevel - startLevel)}$$

This was only possible for 17 cases, due to missing data for the rest. There are several other problems with this calculation. For one thing, it doesn't take into account that an initiative could have been started late one year, and completed early another year (the accounts of when the initiatives started/stopped within a year were on the whole weak for all the case stories). But it is reasonable to believe (by general probability) that this is more or less evenly distributed between the months of a year.

Anyway, the average time to "move up" a level is, as figure 4.13 shows, 2 years (1.993, to be precise). Even with the low number of cases as the basis for calculation, this number is surprisingly close to the calculations in the CMM Maturity Profile [28], where the average

number of years to move up a level, based on all organizations that has reported assessment results from 1987 to present, is 1.7 years (1.708, to be precise[17]).

4.3 Case study quality tool

As mentioned is section 2.3.3, appropriate amounts of contextual information is thought to be a prerequisite for facilitating learning from case studies. To measure the potential for learning from a given CMM case study in this review, all other factors being equal, we have developed a tool to do this.

The tool is a set of conditions, with an associated score. If the case study scores on a condition, it is awarded the associated score. The score can be negative, to indicate a particularly compromising condition. The scores are finally added together to create a "quality index" for that particular case study.

Condition code	*Description*	*Score*	*Rationale, special comment*
SECTOR	Is the industry sector of the company given?	+1	-
SCOPE	Is the number of developers in the scope given?	+1	-
INVEST	Are any numbers given for the investment effort put into the initiative?	+1	-
EFFECT	Is any information regarding effects of the initiative given?	+1	-
ROI	Is an explicitly calculated ROI present?	+2	ROI is a very concrete figure, and no matter how it is calculated, it is very easy to relate to for the reader.
ENABLER	Are any enablers given?	+2	Enablers are of value for knowing how to commit your improvement initiative.
ENABLER5	Are more than five enablers given?	+1	
DISABLER	Are any disablers given?	+2	Disablers are of value for knowing how to not commit your improvement initiative.
DISABLER5	Are more than five disablers given?	+1	
LEVELS	Are both start and ending level given?	+2	We regard proper information of what CMM levels are in question as contextual information of high value.

[17] The average of 24, 20, 25 and 13 months (the numbers given in the referenced maturity profile) is 20.5 months. This equals to 1.708 years.

NOLEVELS	Are no levels given?	-2	If no indication whatsoever is given regarding the levels, it is very hard to relate to the initiative described in more than in a general, uncertain way. This is penalized.
DURATION	Is start and end year, or number of months given?	+3	Any indication of how long the initiative lasted will also prove as invaluable for the readers' interpretation of the case study.

Table 4.9: Case study quality tool

The score can be –2 at its lowest, and 17 at its highest. Because the quality tool is an abstraction of all the best qualities from all the case stories, it is unrealistic to hope that any case story scores "top" of the 71 reviewed.

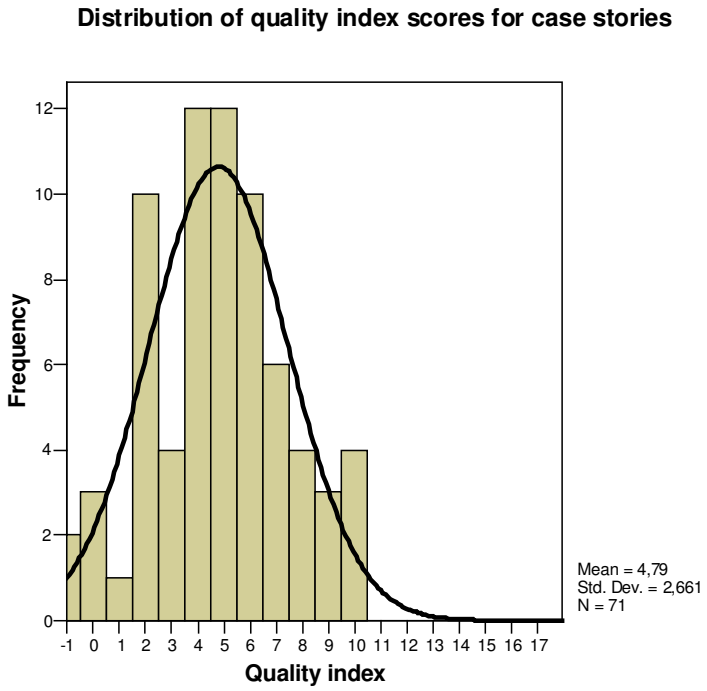


Figure 4.14: Distribution of quality index scores for case stories

Nevertheless, we see from figure 4.14 that the distribution for case story quality is peaking relatively low on the scale, at 4.79. Even if we have constructed a "hypothetical measure" of how the "ideal" CMM case study could have been, things look sobering: we easily see that more than half of the case stories have score of 4 or less, which is *less* than 1/3 of the highest

possible score (17). We can for sure draw one conclusion from this: practitioners would like the case stories to be of better quality to be able to use them more efficiently.

An interesting question is: are there any particular attributes about the "best" and the "worst" case stories with regards to quality? Yes, we actually found an apparent relationship between reported *industry sector* and case story *quality*. We grouped all companies related to aerospace and defense, including U.S. Air Force in-house organizations, in one category: "D&A". Table 4.10 illustrates the result.

		Type of industry		
	Case stories total	D&A (see above)	Unreported industry	Other industry
Worst quality (Index -1 to 3)	20 (100%)	6 (30%)	4 (20%)	10 (50%)
Best quality (Index 6 to 10)	27 (100%)	16 (59%)	2 (7%)	9 (33%)

Table 4.10: Worst and best quality case studies, per industry

Amongst the "worst" case stories (the ones with a quality index less than 3) there are 20 case stories, which of 6 come from defense and aerospace related companies and U.S. Air Force in-house production. In other words, of the worst case stories, 50% come from "civilian sector". Of the "best" case stories (the ones with a quality index equal to, and greater than 6) there are 27 case stories, there are 16 articles D&A, whilst only 9 that are surely from other industries.

Because of the low sample size, these numbers are susceptible to influence from the "unreported" variable, but in general the tendency is nevertheless that amongst the higher quality case studies, aerospace and defense-type software organizations are overrepresented. Combining these findings with the fact that "civilian" CMM case studies are underrepresented to begin with compared to the industry sector of assessed companies with a given CMM level, things look bleak for the civilian sector. They're not only underrepresented, as pointed out earlier in this chapter, they have the worst quality case studies as well.

5 Qualitative analysis

5.1 A review of "success factors" for CBS

It is known that organizations embarking in SPI initiatives need and seek knowledge on how to implement SPI in the best manner (a SEI-report by Goldenson and Herbsleb states that 67% want "guidance" in software process improvement activities [134]). Unveiling the success factors, the "enablers", of software process improvement, is a recurring theme throughout all of SPI literature. Because the literature review we do in this thesis is (to our knowledge) the most extensive ever done in the field of CMM case stories, we focused specifically on "reported enablers" of CMM in the stories, in the pursuit of building "the ultimate list of CMM enablers".

We argue that there are basically two methods of deriving success factors in a quantitative way: committing surveys with "human" respondents, and doing quantitative analyses of literature (literature reviews, like this one).

Examples of unique surveys are a SEI study [134], a Norwegian study [14], and an UK study [135]. By doing a literature search at i.e. INSPEC [41] there *seems* to be more studies, because the mentioned "core" studies (at least the SEI and UK study) are referenced in several other articles, for instance [136] and [137].

5.1.1 Critique of an existing quantitative study

An example of a quantitative literature analysis study is *Success Factors of Organizational Change in Software Process Improvement* [50]. They dish up a list of 10 success factors, the first few ranked from the top are "management commitment and support", "managing the improvement project", "tailoring improvement activities", "change agents and opinion leaders", "stabilizing changed processes" etc. But what was Stelzer and Mellis' methodology? Well, they *pre-defined* 10 success factors based on a qualitative first-round of explorative research. Then they examined a whole lot of CMM case study articles, and made a coding scheme *based on how the 10 success factors fit* with each and every CMM case study they reviewed (a total of 31 cases). This can be compared to a rather brutal "top-down"-approach in extracting the validity of case stories.

But it doesn't stop there. One of the 10 success factors postulated in the article with an indisputable fabricated feel to it, is the term "unfreezing the organization". This is of course because it's derived directly from the terms ("unfreezing – change – refreezing") which the classic organizational theorist Lewin used in describing organizational change [138]. Stelzer and Mellis' operationalizes "unfreezing the organization" as: "To overcome this resistance [to change] an additional force is required, a force sufficient to break the habit and to unfreeze the custom." (p. 240), alternatively: "Degree to which inner resistance of an organizational system to change is overcome" (p. 232).

We wanted to take this definition to an "acid test". Fortunately, the coding sheet was included as an appendix to the article, so we could inquire into how "loose" they would allow an article to fit with a success factor. Very loose, indeed, that was, we discovered. Examining the *Corning Inc.* [103], *Hewlett-Packard* [116] and *Schlumberger* [124] articles, which all were coded by Stelzer and Mellis for "unfreezing the organization", we found *no explicit reference whatsoever* to anything we could within a reasonable "slack" of interpretation connect to the two definitions (and *of course* no reference to "unfreezing the organization" as a literal term).

If this wasn't enough, many of the case stories that are used in the review, are of very dubious quality. Some of them consisted entirely of *PowerPoint* slides (an example is [139]), and for reasons elaborated on in section 3.2, we don't see such case studies fit as to make objects of scientific study in this context.

Stelzer and Mellis thus make the mistake of using *survey* methodology on *qualitative* text, they have a hammer, and look for anything that vaguely resembles a nail. Of course, in articles were a success factor like "management commitment" is mentioned explicitly (for instance [68]), this methodology can have it's strengths. But, as demonstrated above, with vague concepts like "unfreezing the organization", things are likely to quickly lose intersubjectivity, and scientific validity accordingly. In this review, we aim and hope to do quite the contrary of Stelzer and Mellis, as we start with no pre-defined set of success factors.

5.1.2 The protocol used in this review

The protocol for gathering success factors is the following:

1. We define "success factors" (as synonymous with "enablers" and) as a subset of "lessons learned". That means that a "lesson learnt", when an article emphasizes this, doesn't have to be a "success factor", but that a "success factor" necessarily is a lesson learnt (this excludes "success factors" of articles like [108], were the authors obviously haven't walked the walk, but *are* talking the talk). *An example:* Under the "lessons learnt" heading in the Hughes Aircraft-article [68], the phrase "pride is the most important result" is reported as a "lesson", but obviously this does not make any natural match as a "success factor".

2. To count one "success factor" as being *equal* across several articles, we demand a *literal match*. Thus, "management commitment" is *not* the same as "management sponsorship". In fact, in the *Corning Inc.* case study [103], *both* "management commitment" *and* "management sponsorship" are specifically mentioned as *separate* success factors, thus giving a strong argument for *discrimination based on literalness*. One important finding in this study is that a reference to *management commitment* is only found in as few as *seven* of the 71 case studies! (Though success factors *relating to* management commitment is found in 24 case studies. But it has to be noted: not more than 24!)

3. There is one exception to the protocol. In the articles [62] and [94], the authors list almost an identical set of enablers in the two articles, even if they are talking about different scopes. To avoid that equality of authors result in a heightened frequency of some reported success factors, we have excluded duplicates in this single case.

There is a problem with this protocol. It's really dependent on the qualitative "horizon of understanding" (see section 3.4 above) of the reader. Because, what is, in case of doubt, to be counted as a success factor or not? There is a slight weakness with this, as it would be in any review that has to resort to qualitative interpretation. But we feel that we, in general, have "our methodological feet dry" to a much greater degree than Stelzer and Mellis.

Because, all in all, we have provided a list of success factors where "interpretational subtleties" to a much lesser degree is "cut away". From this list software engineering researchers and practitioners alike can read the enablers "uncut", and when they are particularly interested in a given enabler, or are uncertain about anything, they can go to the literature and review the case story where that explicit enabler is present.

5.1.3 Groupings of success factors

The success factors (from here referred to as "enablers") were for *convenience only* grouped in to 26 different categories. The rationale for grouping was to present the enablers in a more intelligible way than possible with just listing all the 187 different enablers in one "flat" list. It has to be *strongly emphasized* that the grouping categories were *arbitrarily* made! It follows that many of the success factors *could make a fit in several categories*, though I have made no effort to do this, because such a grouping would probably always remain disputable.

One example of the problem of making success factors fit in categories, is the success factor "the SEPG is chartered for long-term process improvement [79]", which I have placed in the category named "Overall strategy and focus", and (by first looks, maybe surprisingly) *not* in "The existence of the SEPG" category. I did this placement of this particular success factor because my interpretation of the success factor (in its original context) is that it wants to say something about the necessity of the *prolonged* existence of the SEPG, which I believe is a matter of "Overall strategy and focus" for the organization. Most of the other success factors in the "The existence of the SEPG" category mostly relate to the importance of there *being* a SEPG to coordinate CBS.

It is thus up to the reader to agree, or disagree, to the correctness of placing a certain enabler in a certain category. There *is* a certain possibility for the reader to somehow reproduce the same groupings as I have done. Because of this I find it relevant to show a statistic of how many enablers that scored on the relative categories. Though, the information in table 5.1 can *only* be used on the basis on the previously mentioned disclaimers.

Category of success factor	*Frequency*	*Percent*
"All That Management Commitment"	24	13%
"Massaging the organization"	20	11%
Planning implementation	14	7%
Performing implementation	11	6%
Communication and organizational learning	10	5%

Overall strategy and focus	9	5%
The existence of the SEPG	9	5%
Ownership and dedication	8	4%
Commitment in general	8	4%
Climbing the CMM levels	6	3%
Interpreting the CMM your own way	6	3%
The “human” (socio-technical) dimension	6	3%
Learn from others	5	3%
Training your organization	5	3%
Process adaptation	4	2%
Business orientation towards quality improvement	4	2%
Avoiding process for its own sake	4	2%
Do It Your Own Way	3	2%
Knowing your organization	3	2%
Coordination with the rest of the organization	3	2%
Combining with other initiatives	3	2%
Measurement	2	1%
Competence	2	1%
The customer	2	1%
Verifying implementation	2	1%
Sum categorized enablers	173	93%
Enablers that didn’t make an obvious fit in other categories	14	7%
Total	187	100%

Table 5.1: Categories of success factors

5.1.4 What’s really in a success factor?

As reported below, we see that success factors on average are widely different from case story to case story. There are two possible and very different explanations for this – the first one which relates to a possible low validity of success factors due to the limited cognitive abilities of the person(s) authoring the case story, and the last one which relates to the fact that success factors for SPI in general may vary:

1. The human mechanisms for reporting success factors are subjective, elusive and highly individual (almost random).

2. The actual success factors for SPI vary greatly from organization to organization.

To start with the first point, there is not a written “receipt” with instruction on how to write a CMM case study. If such a receipt stated explicitly: report *all* the success factors necessary for your SPI initiative, maybe the picture would have looked differently – and that for instance “management commitment” would have been more prevalent amongst the success

factors in "most" of the studies. A parallel effect is that any individual is likely to have a personal (hermeneutic – see section 3.4 above) horizon of understanding of any given (organizational) situation. In an organizational context, this idea can also be fruitfully explained with Herbert Simon's term "bounded rationality" [140] – any individual in an organization can only relate to his or hers "part" of the whole organizational picture, and will thus report accordingly. Following Simon's theory, *any* CMM case study will be sub-optimal and more or less flawed. As Magne Jørgensen emphasizes [30], we simply cannot *expect* from an actor in an organization that he or she has the necessary oversight to provide correct success factors. It is hard enough, or maybe impossible, for an external observer to pinpoint what "enabled" an improvement initiative. For an *internal* observer, by definition not even having access to an "objective" view of the organization, the challenge is even greater. This is a rather pessimistic view, but nonetheless important – because it potentially undermines much of the rationale for using "success factors" as guiding advice in CBS.

The second point is rather interesting, because it accounts for that the findings of this article review actually can shed some more, and improved, light on what success factors are necessary for software process improvement. Keep in mind that most results that emphasize the classical "management commitment" and "business orientation" success factors come from the *structured* surveys mentioned in section 5.1, above. The strength of the qualitative review aspects of this article review is that is does not try to "force-fit" a certain experience on any given article, but simply reports (to its best effort) what the article actually reports. In this perspective, when we're taking the report seriously, it is rather peculiar that success factors allegedly related to "massaging the organization" clocks in on *second* place of the kind of success factors that are most likely to be present in a CMM case study. This is in very serious contrast to the Stelzer and Mellis-study [50], where the "unfreezing the organization"-success factor (se section 5.1.1, above) is rated as the *least* (of ten) important success factor for CBS.

To sum it up: when asking "what's really in a success factor", we are rather unsure of what to answer. Taking the above points into account, the whole concept of using "success factors" as guidance for CBS can easily be thrown suspicion on. If not the "bounded rationality" or "personal horizon of understanding" of the author diminishes validity of success factors enough, the fact that different organizations report so different success factors can possibly send the whole notion of the usefulness of "success factors" into exile.

Because most other research into SPI (including us in the list in 5.1.5, below) is vehemently trying to chisel and preserve any success factor they can get hold of, I will not linger on these critical remarks. That said, because of the here-mentioned problems of validity, an advice to practitioners and researchers is that a thorough skeptical stance should be taken to any "success factor" reported in any literature (including this).

5.1.5 Complete listing of success factors

Below follows the complete listing of success factors for CMM improvement. A total of 187 explicit success factors were found. The enablers were found in 43 of the case studies.

"All That Management Commitment"	*n=24*
Enabler and references	*Freq.*
management commitment [68] [38] [85] [88] [100] [31] [103]	7
management sponsorship [76] [103]	2
top management commitment, sponsorship and resource investment [61]	1
management commitment needed at *all levels* [62]	1
secure management support [119]	1
top management commitment [56]	1
leadership [83]	1
sponsorship from all levels of management [79]	1
management commitment and support [98]	1
senior management sponsorship [36]	1
support by management [99]	1
sponsorship for SPI by senior management [57]	1
strong sponsorship by chief executive officer [124]	1
top management support [34]	1
senior management support [112]	1
senior management leadership and support [60]	1
unrestricted senior management commitment [115]	1

Massaging the organization	*n=20*
Enabler and references	*Freq.*
change of infrastructure [88]	1
selling at individual level [88]	1
managers must be convinced of SPI value [62]	1
overcoming resistance to change (is the hardest part) [62]	1
set realistic expectations for senior management [119]	1

Address resistance and culture [92]	1
keep management involved [93]	1
have a complete and representative cross-section of the company participate in the process [100]	1
cooperate with developers [31]	1
assessments useful for motivating SPI [124]	1
assessments must be followed by active efforts to encourage formation of software improvement activities [124]	1
in-house groups should be trained together to effect a cultural change [124]	1
staff buy-in and support [103]	1
spend time ensuring everyone knows what is wanted to get out of the assessment [127]	1
Cultural change [34]	1
visionary person [34]	1
cheerleader person [34]	1
change agents [34]	1
individual buy-in [34]	1
senior management buy-in [120]	1

Planning implementation	*n=14*
Enabler and references	*Freq.*
make baselines before you begin [84]	1
set checkpoints and milestones [84]	1
make three documents early: Quality Management System manual, process improvement plan and profile (based on gap analysis) [84]	1
carefully select pilot projects [119]	1
start a process initiative from the top level process [119]	1
treat each SPI initiative as a project [36]	1
have a schedule of a year or less [36]	1
a baseline of its current software process and performance [57]	1
Creation of a multi-year process improvement plan [107]	1
don't bite off more than you can chew [129]	1
recognize that improvement needs will change [60]	1
schedule your improvements [94]	1
plan for organizational process focus [94]	1
focus efforts on new projects – it's hard to change projects at low maturity once they have started [94]	1

Performing implementation	n=11
Enabler and references	*Freq.*
start a measurement program immediately [84]	1
start improvement activities soon after an assessment [119]	1
improvement has to be institutionalized [83]	1
enforcement is good for implementation [75]	1
a delivery system to deploy the processes [36]	1
don't argue with a framework after it's accepted [36]	1
keep processes simple [36]	1
Manage the risks to the SPI project [36]	1
the "big bang" approach to deployment can work [36]	1
keep the pressure [124]	1
Starting to define processes is itself the major improvement activity in Level 1 organizations [104]	1

Communication and organizational learning	n=10
Enabler and references	*Freq.*
adopt a common vocabulary [119]	1
mechanism to diffuse decisions and lessons learned [90]	1
communication [83]	1
give high priority to communication and feedback [92]	1
keep everybody in the company well informed about intentions, plans and benefits (use key communicators) [100]	1
communicate constantly [124]	1
shared language [34]	1
train and communicate, effectively and often [129]	1
inform everybody that SPI is "real work" [60]	1
communication and education [60]	1

Overall strategy and focus	n=9
Enabler and references	*Freq.*
"improvement management" as overall concept [38]	1
focus on improvement in general, not on CMM level [88]	1
time [83]	1
Address improvement as a project [92]	1
align with strategic plan [92]	1
the SEPG is chartered for long-term process improvement [79]	1
a quality improvement goal and business incentive for achieving it [57]	1

tackle process improvement with right mindset (comment som rar greie!) [129]	1
Address all areas of improvement [60]	1

The existence of the SEPG	*n=9*
Enabler and references	*Freq.*
a focal point for the SPI effort, the SEPG, is necessary [68]	1
establish a software process engineering group [119]	1
the SEPG understands its changing role as an organization matures [79]	1
software engineering process group (SEPG) composition [96]	1
a dedicated group for performing process management activities [36]	1
one or more focused working groups, staffed and with charters [57]	1
dedicated process engineer(s) [31]	1
a central, experienced team should participate in software improvement activities [124]	1
SEPG leader [60]	1

Ownership and dedication	*n=8*
Enabler and references	*Freq.*
project manager dedicated to SPI [88]	1
SEPG members are process owners[18] [79]	1
have executors of the process define the process [36]	1
processes should be developed by those who use them [99]	1
management steering group that owns responsibility for improvements [57]	1
groups must choose their own method of improvement so that "ownership" occurs [124]	1
employee participation [104]	1
involving people in the change process [126]	1

Commitment in general	*n=8*
Enabler and references	*Freq.*
commitment [83]	1
continually pursue SPI at every level of the organization [83]	1
consistent, stable, senior management [83]	1
a percentage of time for software engineers and managers to work on SPI [57]	1
"top-down" and "bottom-up" support [129]	1
high staff morale [112]	1
there are no silver bullets – invest time, talent and commitment [94]	1

[18] This particular enabler could also probably been in the SEPG-category. For this, refer to the discussion in 5.1.3, above.

Support from across the organization to the SEPG [120]	1

Climbing the CMM levels	*n=6*
Enabler and references	*Freq.*
understanding the "critical jumps" from level 1 to 2 and level 3 to 4 [61]	1
avoid thinking you'll stay if reach a given level [61]	1
continuous instead of staged progression [84]	1
view an assessment as a vehicle for further process assessment [79]	1
understand the practices one level above your current level [75]	1
some level 4 and 5 KPAs can be implemented together [75]	1

Interpreting the CMM your own way	*n=6*
Enabler and references	*Freq.*
avoid using CMM as a "checklist" [61]	1
do not use CMM as a checklist [83]	1
treat the CMM as a guide [76]	1
tailor the process [31]	1
don't assume you can adopt someone else's processes [129]	1
start with assessing the intent of each KPA so that you can determine how it fits into your environment (a top-down-approach) [62]	1

The "human" (socio-technical) dimension	*n=6*
Enabler and references	*Freq.*
Manage the human dimension of the process improvement effort [119]	1
Process improvement requires additional people skills [119]	1
Address all "socio-technical aspects": tasks, structure, technology, people, culture [92]	1
Manage culture [93]	1
management of the human dimension of the process improvement effort [98]	1
change management practices (account for that you're dealing with humans) [31]	1

Learn from others	*n=5*
Enabler and references	*Freq.*
benchmarking other, similar organizations' CMM practices, and learning from them [118]	1
get support from organizational change experts [119]	1
use lessons from other organizations [92]	1
apply industry best practices whenever relevant [92]	1

get outside help [129]	1

Training your organization	*n=5*
Enabler and references	*Freq.*
train all users of the processes methods and tools [119]	1
Training program [91]	1
education [83]	1
a budget for staff skill growth in software engineering and management [57]	1
Training in statistical process control [94]	1

Process adaptation	*n=4*
Enabler and references	*Freq.*
adapt processes directly to the project's needs [84]	1
adapt the CMM to the environment of the team [35]	1
take into account project and developer needs [99]	1
utilize checklists [129]	1

Business orientation towards quality improvement	*n=4*
Enabler and references	*Freq.*
business orientation [85]	1
tie process improvement activities to business objectives [119]	1
a strong tie to the business case [96]	1
tie the improvements in our product process to the business goals of the organization [126]	1

Avoiding process for its own sake	*n=4*
Enabler and references	*Freq.*
emphasize productivity, quality and cycle time (avoid process for its own sake) [62]	1
embrace process improvement for reasons other than to merely achieve a maturity level [83]	1
emphasize performance, not documents that gather dust [75]	1
don't work for the framework, let it work for you [36]	1

Do It Your Own Way	*n=3*
Enabler and references	*Freq.*
do not copy process documents from other organizations [62]	1
don't be afraid to fail [60]	1

tailor your improvements to your needs [60]	1

Knowing your organization	*n=3*
Enabler and references	*Freq.*
Identify management needs, expectations and understanding of the problem [119]	1
figure out how you really work before you write processes [129]	1
document what you do, not what you think you should be doing [129]	1

Coordination with the rest of the organization	*n=3*
Enabler and references	*Freq.*
coordinate project processes with organizational processes from the start [75]	1
strong coordination across the organization [34]	1
Involve everyone [60]	1

Combining with other initiatives	*n=3*
Enabler and references	*Freq.*
Balanced Score Card as basis concept for the management of the company [38]	1
team software process [95]	1
combine SPI with TQM programme [105]	1

Measurement	*n=2*
Enabler and references	*Freq.*
measure improvements on the product, not the process [84]	1
keep metrics simple and with value for project management [36]	1

Competence	*n=2*
Enabler and references	*Freq.*
software process expertise (in the organization) [68]	1
professional software engineers [38]	1

The customer	*n=2*
Enabler and references	*Freq.*
client orientation from the start [38]	1
inform customer about process changes [62]	1

Verifying implementation	*n=2*
Enabler and references	*Freq.*
conduct process audits [119]	1
conduct team effectiveness surveys [119]	1

Enablers that didn't make an obvious fit in other categories	*n=14*
Enabler and references	*Freq.*
define roles [92]	1
use both industry and organization expertise to obtain targets [93]	1
tested assessment process [103]	1
use of their own, custom-made assessment tool [87]	1
focus improving both product and process [84]	1
both management and technical activities necessary [84]	1
a dedicated organization for software process improvement (within the organization) [84]	1
Allocate adequate resources [84]	1
use of automated tools [38]	1
focus on improving new projects early [62]	1
use in-house people to define processes, not outside process experts [62]	1
continuity [60]	1
small organizations (150-) must minimize the limitations of its smaller size and maximize the benefits inherent in its culture [117]	1
competitive pressure to improve quality [103]	1

Table 5.2: Complete list of enablers for CBS

5.1.6 The flipside of success factors: CMM disablers

A total of 25 disablers were found, distributed on 6 of the case stories. The protocol for extracting CMM disablers is analogous to that of success factors (see section 5.1.2, above), but here we are searching for factors that are explicitly mentioned as being contra-productive for improvement.

Only one categorization is made, solely because of the amount of disablers present in the very special article by Florence [122] concerning an undisclosed organization's failed attempt to reach CMM level 4.

The problematic shift from level 3 to level 4	n=7
Disabler and references	*Freq.*
level 4 is a paradigm shift from level 3 [122]	1
standards on contract provided for processes and artifacts at level 3, but not level 4 [122]	1
too few people involved with SPI at level 4 [122]	1
level 3 based on business goals, level 4 done for process sake [122]	1
many published examples for level 3, but few for level 4 [122]	1
many experienced assessors for level 3, but not for level 4 [122]	1
commitment, funding and cooperation existed at level 3, but were not adequate for level 4 [122]	1

Uncategorized	n=15
Disabler and references	*Freq.*
not involving project members in various decisions at company level [90]	1
using the CMM as checklist [83]	1
turf politics [57]	1
lack of definitive software quality goals [57]	1
staff shortages of SE, management and SPI [57]	1
loss of key technical skills due to employee turnover [57]	1
lack of commitment of continued leadership for software improvement [57]	1
Making SPI sufficiently high priority [57]	1
invisible, wavering or sporadic senior management sponsorship [57]	1
lack of focus on change as a part of the engineering job [57]	1
lack of willingness and discipline to change [57]	1
well-coached groups at assessments [124]	1
it is extremely difficult to discover the true picture of the organization's practices [124]	1
risky to introduce [126]	1
difficult to keep on track because factors that interact with the organization is changing [126]	1

Table 5.3: Complete list of disablers for CBS

5.2 A review of the rationale for calculation of ROI

To better get a grip at the rationale for calculating ROI in case studies of CBS, we have taken a qualitative look at the 9 case studies giving a ROI. In table 5.4, we present the company and scope of the case study, the proclaimed ROI, an examination into how the ROI was

calculated, relevant comments, and finally an account if the potential of Hawthorne-like effects were mentioned.

Company, scope and reference	*ROI*	*Way of calculating*	*Comment*	*Hawthorne-like effects mentioned?*
Oklahoma City ALC (Aircraft Software Division) [60]	6.35:1	None given.	-	-
Motorola (unknown scope) [61]	2.5:1	None given.	-	-
Raytheon (Software systems laboratory) [91]	7.7:1	"Philips Crosby Associates approach" to help quantify the benefit of improvements made to ongoing projects. Measures "cost of conformance" vs. "cost of non-conformance."	Very thorough calculation over four pages with lots of data available in article. Totally exceptional compared to other articles. "Best of class."	Yes. Dion explicitly state that "benefits [...] derived from improved competitive position, higher morale and the lower absenteeism and attrition rates" (p. 34) were not subtracted from the calculation.
Oklahoma City ALC (SED) [82]	7.5:1	A productivity increase of 10X was calculated over 8 years. Savings (the basis for ROI) was calculated based on the cost had there not been a productivity increase.	A very simplified calculation compared with [91].	-
Ogden ALC (SED) [77]	19:1	Calculation can be viewed simplified at best, obfuscated at worst, and states that it takes into account that savings from SPI can be taken in as far as five years into the future [!].	Substantial amounts of missing data. No way to verify calculation whatsoever.	-
Raytheon (unknown scope) [120]	6:1	A productivity increase of 144%[19] over a period of 3 years whilst spending 6% of the annual budget on process improvement. Because this article does not have scope information, it's not known if it's the same scope (both are Raytheon) as presented in [91]. But it can be noted that the ROI has "decreased" from 7.7:1 in 1993 to 6:1 in 2001.	Apparently some missing data. No way to verify calculation.	-
Hewlett-Packard ("One HP division") [116]	9:1	Some effects are given (reduced cycle time etc.), but no clear relation from given data to calculated ROI.	Calculation is obviously simplified in its printed form.	-

[19] Clarification: The authors are wise enough to specify this number as an increase from a relative level of 1 to a relative level of 2.44. (An increase of 144% could also be interpreted as an increase from a relative level of 1 to 1.44. This totally pre-school statistical fallacy is unfortunately common in software engineering research, and also clings to certain numbers in the Standish Group reports. For more on that problem, see [6].

Motorola (Government Electronics Division) [62]	6.77:1	Calculation is based on a hypothetical project.	Rather thorough calculation, but numbers are a bit "from the blue".	-
Boeing (Space Transportation Systems) [78]	7.75:1	Comparing amount of added effort in "design" with reduction in rework "design", "code and unit test" and "integration and system test".	Calculation is simple but convincing.	-

Table 5.4: Matrix of rationale for calculating ROI for nine case studies

No calculation is equal. All calculations are based on different methods of calculating and different numbers.

It's usually the same few companies that calculate ROI. Even if we have not enough information to conclude that the articles write about the same scopes (sometimes the scopes are unknown), this presents an unnerving trend: that extremely few software organizations have provided the numbers that we "take for granted" regarding ROI for CBS.

That said, the Raytheon article of 1993 [91] is *very* believable in all of its calculations. Unless Raytheon deliberately *forged* their findings, this calculation of ROI can be referred to as the *state of the art* of ROI calculations. For more on the implications on this, see section 6.2.3.2.

6 Discussion and implications

6.1 Learning from CMM case studies

6.1.1 Learning based on context and quality index

As pointed out in section 2.3.3, a certain degree of similarity of *context* between the case study and that of the organization embarking on CBS seems to be essential to facilitate learning from CMM experience reports.

How representative in context are the universe of CMM experience reports relative to the universe of software organizations? In the quantitative analysis (chapter 4), we found a lot of skewed results. In section 4.2.4 we pointed out that "Military/Federal" and "DoD/Fed Contractor" sectors were grossly overrepresented in the case-stories compared to the "Commercial/In-house"-sector when using the (non-scientific) CMM maturity profile published by SEI.

If nationality, and implicitly, national culture is a contextual factor to be reckoned with (which some musings [141] indicate, but because of the limited research on this, we won't go into detail on that here), it is noted in section 4.2.7 that at least 58% of the case studies are from U.S. companies. Even if we were unable to obtain hard numbers for the global software industry, we find it unlikely that as much as 58% of the world's software-producing companies originate in the U.S..

When it comes to the size of companies that are reporting, as pointed out in section 4.2.9, the results are even more skewed. We found that 97.2% of all U.S. software companies have less than 100 employees. Yet, no less than 75% of the case stories document *improvement scopes* (which, to add assault to injury, is just a subset of total employees of a given company) with *more* than 100 employees!

Clearly, the conclusion to these findings is: the distribution of companies that has committed case studies is in general *not* representative for the typical distribution of companies thought to have commenced in (or are potential candidates to commence in) CBS. In other words; the average company wanting to embark on CBS is a lot more likely to find most of the case

stories as not relevant to their particular context. We suggest in the next section a typology to deal with this.

6.1.2 A typology of CMM case studies

Based on matching number of contextual factors, it can be measured to which degree a given case-story is sensible to learn from for a given organization.

This can be expressed by roughly categorizing case studies in three different categories; *enabling*, *inspiring* and *entertaining*.

Category of case study for reader	*Factors needed to reproduce*	*(Practically) possible to reproduce?*	*Description*
enabling (very high match in context)	adequate funding and dedication	yes, highly	"Enabling" case studies are enabling proof of that "something can be done" (if the case study isn't a fraud, of course). This kind of case story can be said to be common in classical engineering disciplines and medicine. The blueprints and technology exists, it's only a matter of funding.
inspiring (medium match in context)	as above, + one or more unknown factors	yes, somewhat	"Inspiring" case studies are case studies that can be used as guideline for implementing whatever the case study describes. But, it is somewhat apparent that the reader's circumstances are different from that of the author, and that the reader has to do his or hers own critical research to be able to reach it's goals.
entertaining (low match in context)	as above, + unattainable structural factors	no	"Entertaining" case studies are those where obvious structural factors differ so strongly from the author's situation to that of the reader, that the reader easily identifies them and sees the author's depicted situation as unrealistic.

Table 6.1: Three types of CMM case studies

The suggested power of this attempted typology is that it crucially points out that different case stories should have different effects on different kinds of readers. The allegedly high failure rate of CBS initiatives (see section 2.2.4) is obviously caused by that a high proportion of CMM case studies for its readers are either in the *inspiring*, or *entertaining* category. But *not* enabling.

This can be confusing for the reader, when most of the case studies presented in traditional engineering, for example, *are* of an enabling nature. They follow the "normal" logic: "if it can be done here, it can be done everywhere". It is naturally the responsibility of the *reader* to evaluate what kind of category the case study falls into for the reader's situation.

6.1.2.1 A case against learning

The *classical* case study of the CMM is that of the U.S. Space Shuttle Onboard Software project [59], from which a condensed version was published in the official, printed book version of the SW-CMM v.1.1, released in 1994 [22]. Most software engineering people that has read this account of the CMM level 5 organization will probably attest to that it is a mesmerizing account of an *ideal state* of software engineering, with (literally) zero-defect software and an almost unbelievable software project *nirvana* where "they have only missed 1 deadline in 15 years" (p. 119).

The Space Shuttle story is obviously sadly of *entertaining* value, only. *Everybody* else than NASA (as far as I know) in the world knows that they're not building space shuttle-software. And in addition, most other organizations don't exist in a controlled and not competition-prone environment, with a massive and relatively stable funding from the most resourceful government in the world. From a crude perspective of case story *transferability*, it is, in all its glory, probably the most useless CMM case story in the world!

The Space Shuttle story indicates a general problem of CMM case studies: they are seldom, if ever, of an *enabling* nature.

6.1.3 Why software process improvement is special

But having a set of context variables in place isn't necessarily enough for successful applicability of the experiences of a given CMM case story. Building on established theories of organizational learning, T. Dybå [14] has pointed out that SPI is a *socially constructed*

learning process (p. 257). Socially constructed, because the *learning* takes place in the *social interplay* between the members of the organization. Specifically, successful learning is "a continuous and simultaneous dialectic interplay between the knowledge that the organization has established over the time, and the knowing of the organization's members in their respective contexts" (p. 258). Because the learning process is based on the experiences of the organizations as whole, and the cognitive abilities of their members, no less than *every* SPI initiative is in principle and practice of a *unique* and *un-reproducible* nature.

This leads to the conclusion that, from a theoretical point of view, there can be *no* CMM case-studies of an enabling effect comparable to a typical case-studies of classical engineering and medicine. The fact that there are no studies of "enabling" nature, should be strongly communicated to the software engineering community, because the consequences are rather serious. The only counter-measure to the uncertainty "one or more unknown factors" introduces for an organization, is in fact meticulous research into what the organization feels are "special" for themselves. Only by devoting time getting a certain grasp on its own peculiarities, the organization can perform successful CBS. Again, this is theory. It has to be tested by empirical research.

6.2 The question of failure rate and ROI

6.2.1 Introduction

In this section we will discuss the potential for operating with valid impressions of the failure rate of CMM-based improvement, and it's potential Return On Investment (ROI).

6.2.2 The problem of the failure rate

We have rather clearly illustrated in section 2.2.4, that there are no good empirical data for the failure rate of SPI. This brings us into an interesting question of philosophy of science: on what assumptions can we say that there is a failure rate for SPI? I'll give a few examples:

1. There is literally a myriad of articles published on the subject of CMM. As mentioned in section 3.2.2, a search on INSPEC [41] returned 439 hits on the keywords "capability maturity model". It is a proposition that all these articles wouldn't be published on conferences and in journals if there wasn't a certain sense and a

generalized need in the community for "knowing how to do it right". This need could of course *theoretically* be spawned alone by a general fear of failing in a gargantuan task like committing CBS, but I find it most likely the impetus is also driven by a general knowledge of a few or more organizations that really *have* failed.

2. The sheer size of the task can make organizations likely to fail, and those organizations that have succeeded frequently emphasize that the effort was "no walk in the park". Even in organizations like the resourceful U.S. Air Force, where it had been dictated that all in-house software development organizations should be on level 3 by 1998 [60], challenges were obviously abundant in SPI activities. Melodramatic quotes like the following two tell their own stories: "Finally, it should be understood that what we went through is extremely stressful" [56] and "The race to improve software processes in the Air Force is fast, furious, and formidable; many racers have found it nearly fatal. Often, it's necessary to rethink how the race is run, to pause, and to catch your breath before resuming the uphill battle" [74]. When such exclamations come from one of the most resourceful industry sectors that commit SPI, it adds hail to the probability of that failing SPI is a very real risk.

3. Then we have the effect of what is popularly described by the saying "only the winners write history". For our own account in this review, the almost total absence of case studies of organizations that have *failed* their CMM initiative (pointed to in section 4.2.6), is by all probability no *representative* account of the situation at all. Because: where is the economic incentive for a company to write a decent case study about their own failure? From the record in the review we conducted, it's nothing at all. The two articles, [122] and [125], describing failed improvement initiatives are both written by external authors, and the companies studied are anonymous. As we point to in section 4.2.3, at least 70% (and possibly up to 75%) of authors that has been involved in case study writing in our review, *are* either employed with the company, or *have been* employed with the company. Authors obviously aren't particularly eager to "blow the whistle" on a failed initiative, when the resulting reactions from the company are known to be severe [142].

Of course, we can't *quantify* this failure rate (that is: we know, from this review, of two explicit accounts of SPI failures). Let's then call it a *qualitative* failure rate. This is not

saying that there will *never be* a decent, quantifiable failure rate to navigate by. For instance, a well-designed, large-scale survey on software companies would probably yield interesting and statistically significant indications on a failure rate. (But no one has yet cared to do this.) Such a number would be very interesting, because, amongst other things, a *true* representation of Return On Investment for SPI would only be possible to calculate with failure rate put into the equation.

6.2.3 The question of ROI

Return On Investment, ROI, the *sine qua non* for embarking on CBS. If it doesn't pay off to invest in SPI, who would invest in it? Undisputable scientific evidence of ROI for SPI is very hard to come by, as we will show in the following sections. Still, the need for results of ROI is clear and present. If science can be compared to a boat continually being rebuilt out in the middle of the sea [143], software engineering research related to the ROI of CBS is a castaway in the water grabbing whatever wreckage it can get hold of (case stories). We'll see how buoyant the wreckage is.

6.2.3.1 Previous quantitative research

There aren't many articles that really has taken the ROI from a quantitative perspective before. The two classics are a SEI-report, *Benefits of CMM-Based Software Process Improvement: Initial Results* [144] and an article from LOGOS International; *Return on Investment from Software Process Improvement as Measured by U.S. Industry* [145].

Broadman and Johnson present (from in general anonymous contributors, but by the look of things, several are probably from the ones mentioned in section 5.2 above) the numbers 1.5:1, 2:1, 4:1, 6:1, 7.7:1, 10:1, 1.26:1 and 5:1, for an average ROI of 4.68:1. Hersleb *et al.* got the numbers 3.5:1, 6:1, 6:1, 5:1 and 3.5:1, for an average ROI of 4.8:1.

A very recent article in *IEEE Software* by Solingen [146] does a dodgy review of published ROI in the literature, including the two above mentioned articles. By dodgy I mean that numbers from anonymous cases are included, without any possibility for checking if a ROI is

counted twice – once as an anonymous entry, and once as an official[20]. This put into the equation, Solingen ends up with an average of 7 (with a median of 6.6)[21].

So here we end up with four number: 4.68, 4.8, 7 and 8.1 (the latter number, 8.1, is from this review – see section 4.2.10). But what on earth does the ROI numbers really mean?

6.2.3.2 The problem of calculating ROI

As pointed out in section 5.2, the way for calculating ROI in the case stories differs widely for the few cases that actually has specified how they calculated ROI. This situation is coherent with that Broadman and Johnson points out that they experienced that different interpretations of ROI are used in classical textbooks, in [U.S.] Government and U.S. Industry [145].

Calculating the cost of SPI is typically done by dividing the measured benefits (in currency) by the measured costs (in currency). Calculating costs is usually seen as the easy part [146], as it can be derived from the number of man-hours put into software process improvement (which usually is based on some more or less fixed hourly cost the company operates with).

Calculating the *benefits* of SPI is another matter. A typical measure of benefit can be "reduced error density in delivered KLOC". It is a common perception (though not scientifically proven [30]) that catching errors early in the production process typically saves a software organization of large amounts of money, because fixing a post-delivery defect typically costs an organization a lot more than fixing the defect in the early cycles of development. This is problematic to quantify precisely, because many different departments are involved in resolving a post-delivery defect: customer support, design, development and testing, to name the obvious ones.

Another typical measure of benefit is production "effort per delivered KLOC". This is used in the Raytheon article [91] under the guise of "equivalent delivered source instructions per man-month." (p. 34). Let's recapitulate what we quote Kitchenham on in section 2.3.3: "experimental context [is] extremely important for software engineering research" [39]. "Effort per delivered KLOC" is thus also a problematic measure, because it's largely context

[20] Even though the possibility for this is low. Because the low sample of published ROI for CBS, the ROI reported is ironically more of a unique *serial number identifying a certain company*, than anything else...
[21] The Solingen article has a few articles that is not covered within this review. This is because the articles didn't fit the protocol used in this review (for more on this, see section 3.2).

dependent. Variations among projects within the organization, and the potential of influence from external factors, will surely influence this measure.

Software projects often fail catastrophically (see the introduction of this thesis). Another measure that could positively influence the probable benefits of a SPI initiative, is for instance the improved risk management processes prescribed by CMM. How would you quantify a probable reduced risk of the whole project going astray? If the success of a project was critical to a whole company's very existence – how would you value, in ROI terms, the effect of potentially reducing a disastrous risk from (the numbers from thin air) 5% to 1%? This scenario would probably take the whiff out even the most imaginative and self-confident of estimators.

In addition to these measures, there are sources of "noise" that influence the value of measuring SPI-related benefits. There are many examples of this. Few companies perform process improvement in a vacuum – most try multiple ways of remaining competitive [147]. How do you know if it's not the effects of other improvement initiatives that you are measuring? As Sheard puts it: "Suppose your time-to-market metric decreases. Is this because you implemented improvements leading to Level 3 in the SW-CMM? [...] Or maybe you are now making products that is much closer in content to other products and thus new versions can come out faster?" Another "spurious" effect is pointed to by Dion: maybe it's the effort to provide engineers with new development tools that is paying off? [91]

And then we haven't even talked about the so-called *Hawthorne-effect* [148] as a source for noise. Elton Mayo apparently discovered in studying the Hawthorne Plant of the Western Electric Company in Illinois that apparently the mere effect of observing and attending to the subjects of a study (making them feel valuable), increased production by those affected by the study. It has to be mentioned that how the results from Western Electric Company should be interpreted is controversial. Closer readings of Mayo's studies even suggest that the very existence of the effect is suspect (for more on this, see Berkeley Rice's highly readable and humorously named "The Hawthorne defect" [149]). Still, we think it is likely and fruitful to suppose that a "Hawthorne-like" effect is likely to put it's obfuscation on things in the context of SPI. Our argument is that being subject to SPI is a rather large change of work practices, and often emphasized as a *change of culture* [150], and such culture changes are likely to affect productivity in some positive (or negative – see below) way. Dion also mentions the

Hawthorne effect explicitly in the Raytheon case study [91] as a potential source for distortion of his calculated ROI.

A given "Hawthorne" effect could also be thought to "backfire" in some cases, making some kind of *negative* "Hawthorne" effect manifest. The case study by Sampson [114] indicates a severe incident of backfire (people actually leaving the company because of the SPI initiative) in the following, slightly laconic passage, indicating a dramatic reality: "The loss of non-adopters (people leaving because they were unwilling to use the software quality system) was limited to less than ten percent of the organization." Such a "backfiring" Hawthorne-effect would in any case have tremendous impact on the calculated benefits for the improvement initiative.

I have until now tried to make the foundations for a philosophical point, and that is: deriving the benefits of software process improvement in the effort to produce a empirically sane number for a ROI is probably impossible by *definition* because of all the sliding, and unknown, factors involved, the Hawthorne effect included. I hope to have illustrated this problem above.

We still want to use the concept ROI, though. But before we can use it, we have to emphasize the following points:

1. ROI is by nature *not calculable*, as we have tried to show above. Only approximate values for ROI can *ever be hoped to be approached*, no matter how meticulous we perform the calculation. There will always be "one more" effect that hasn't been accounted for, or could be recalculated in a more precise matter.

2. Just because only approximate calculations can be performed (based on individual assumptions of what numbers to count or not), ROI is tremendously prone to *bias*. This bias can be both positive and negative. The most obvious is *positive* bias, where the persons calculating the ROI pick numbers and effects that make the final number "look good". *Negative* bias can also be a problem. For an company-independent researcher calculating a ROI, his unconscious aim is maybe to make a "low" number (as to make a flare and contrast the "high" numbers typically reported by commercial authors). He thus becomes extremely afraid to over-emphasize anything, and might

disregard certain numbers and effects. This may result in lowering the result in the final calculation.

From the faltering argumentation for the foundation for calculation of ROI, demonstrated in this chapter (and the review of the mainly teetering calculations in section 5.2), the whole concept of ROI for CBS seems to be in dire straits. But, from a pragmatist approach, I still find it reasonable to salvage the wreckage and talk about ROI for CBS.

I argue this because of two points:

1. There is one case story with a *very* convincing calculation of ROI published (the Raytheon case story [91] of 1993), as pointed out in section 5.2. It is, put simply, *the exception* to the rule I have stated above. I strongly recommend that article as an introduction to *how* ROI-calculations could be done. Of course, there *is* a serious problem with only having *one* good ROI calculation, and having half a dozen calculations that basically don't come close to the level of this article, but the very existence of this article *proves the point* that very convincing ROI calculations for CBS *can* be performed, as far as I am concerned.

2. There is a horizon of difference between having no *valid* calculations of ROI (but having a dozen or so of approximations), and having *no* calculations of ROI whatsoever. The argument is, from the above, that ROI *can only be approximated* (and sometimes to a very high degree, like the mentioned [91]), never calculated precisely. And we should have no problem allowing the articles mentioned in section 5.2 to be exactly that, *approximations*.

Now for the pragmatist approach, which makes us able to keep the notion of ROI as a valuable tool. In a bold article in *IEEE Software*, Solingen [146] presents a pragmatic view on ROI. He argues that (my italics) "detailed ROI calculations *aren't necessary*. [Because] [i]t's usually sufficient to know the ROI's relative value: is [the ROI] positive, break-even, or negative?"

This leads us to the inevitable conclusion that the ROI on CBS seems to be persistently *high* amongst our given cases. An average ROI of about 8:1 (see section 4.2.10) definitively beats

putting money in the bank. Even a ROI of 2:1 does that, and that is a lower number than the most conservative calculation in this review (2.5:1).

Well, what to do? Let's at first try to use a polemical way of arguing - what is the simplest explanation of the following:

1. All companies that have reported an approximate ROI in case studies have to such a degree biased and/or incomplete calculations of ROI that the *actual* ROI for organizations initiating successful CBS is *lower* than, let's say 2:1.

2. The companies that have reported an approximate ROI may be positively (and/or negatively[22] – we must not forget that!) off-the-mark in their calculations, but it is by practical probability unlikely that the "actual" average ROI for these companies that succeeded in SPI is *lower* than 2:1, because of the average of 8:1, and the dispersion of values from 2.5:1 to 19:1.

It is the author's view that the second explanation is the simplest. This is because there *are* examples in the literature of thorough and almost non-disputable (at least for the educated observer – check for yourself) calculations of ROI. Those two articles presents two (although very different) calculations of ROI that are thorough and believable, and they are the Raytheon article of 1993 by Dion [91] (as mentioned) and the very recently published Solingen article[23] [146] of 2004. It has to be mentioned that both these articles end with peculiarly high ROI's, respectively 7.7:1 and 13:1. This strengthens my point.

This, I reason, indicates strongly that the probability for that the companies have an *actual* ROI (even if this, as we have pointed out, is just an asymptotic, theoretical measure – but as mentioned, this doesn't mean we cannot come *close* to it) that is greater than 2:1, is *very high*. Conversely, that the companies have an *actual* ROI on their CBS that is *lower* than 2:1, is *very low*.

[22] By "negatively" in this context we mean that they made a (far) too conservative calculation, comparing with the potential "real" ROI.
[23] Which is *not* a part of this review, because it is excluded from the selection criteria, see section 3.2.

The discussion has to be left at this stage. Only more empirical evidence, and new, sincere calculations of ROI in upcoming CMM case stories can further justify the point I have tried to put forward.

6.2.3.3 "Potential ROI" versus "successful ROI"

There has been surprisingly little focus (actually: nothing as we know of) in the literature emphasizing the difference between "successful ROI" and "real ROI". The only ROI we have been talking about until now is "successful ROI". Definition:

successfulROI = ROI expected when a CMM SPI initiative is regarded as successful

But, as pointed out in section 6.2.2, there *is* a failure rate for CMM. The *potential* ROI for a company that has not yet embarked on SPI is largely influenced by this failure rate.

potentialROI = successfulROI adjusted for failure rate

Mathematically, this can be expressed as the following:

$$potentialROI = successfulROI \times \frac{(100 - failureRate)}{100}$$

A critical assumption is made for the sake of simplicity at this point: we assume the *average* ROI of a *failed* CBS-initiative, is 0:1. This assumption is made on the grounds that we believe it is likely that in some cases a failed CBS-initiative still has some positive effects for the company (maybe, for instance, some improved project management processes were institutionalized, even if they weren't quite up the KPAs of CMM level 2 [ROI > 0:1]), in other cases effects might be negative (negative staff reactions, lower productivity, even sabotage of processes [ROI < 0:1]), yet in other (hypothetical) cases, there might be no effect at all (ROI = 0:1).

The other great challenge with "potential ROI" is of course that we have two unknown variables in the equation – successful ROI and failure rate. Still, if we assume that successful ROI from CMM-based SPI in no higher than 10, we can generate the matrix illustrated in table 6.3 with different values of *potential ROI* for different values of *failure rate* and *successful ROI*.

Calculated potential ROI

Failure rate in percent	Successful ROI 2	3	4	5	6	7	8	9	10
10	1.8	2.7	3.6	4.5	5.4	6.3	7.2	8.1	9.0
20	1.6	2.4	3.2	4.0	4.8	5.6	6.4	7.2	8.0
30	1.4	2.1	2.8	3.5	4.2	4.9	5.6	6.3	7.0
40	1.2	1.8	2.4	3.0	3.6	4.2	4.8	5.4	6.0
50	1.0	1.5	2.0	2.5	3.0	3.5	4.0	4.5	5.0
60	0.8	1.2	1.6	2.0	2.4	2.8	3.2	3.6	4.0
70	0.6	0.9	1.2	1.5	1.8	2.1	2.4	2.7	3.0
80	0.4	0.6	0.8	1.0	1.2	1.4	1.6	1.8	2.0
90	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
100	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

Table 6.3: Matrix of calculated potential ROI

The matrix can be more easily be comprehended with a surface chart viewed from above, as shown in figure 6.1.

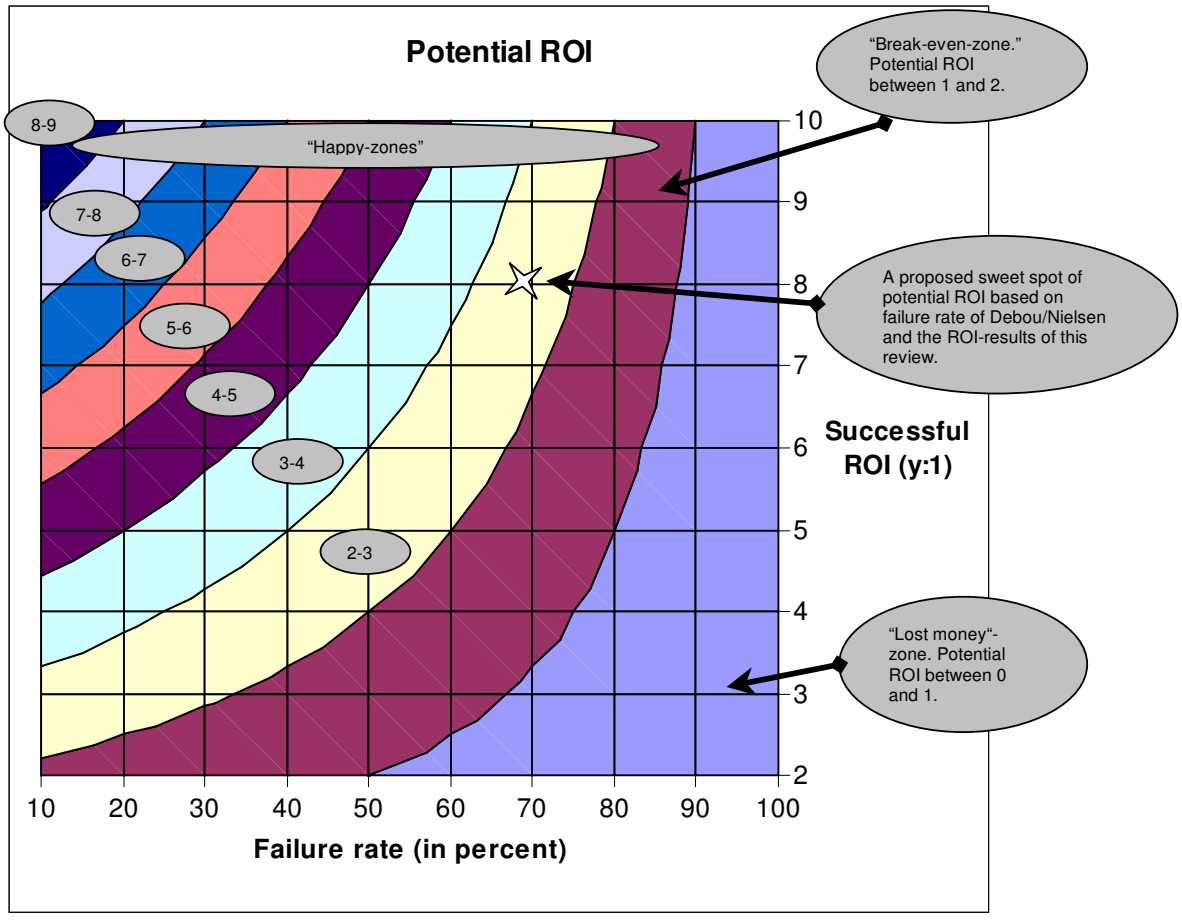


Figure 6.1: Surface chart viewed from above of calculated potential ROI

It has to be noted that *everything* in this model is rather uncertain because of the problematic nature of calculating ROI. This is why we consider also a potential ROI of up to 2:1 to be within a "break-even"-range.

If we *assume* that failure rate is somewhere close to Nielsen and Debou's assumptions (see section 2.2.4), let's say 70%, *and* the average successful ROI, at least for aerospace and defense-organizations, is somewhere close to 8:1, we end up with a potential ROI of 2.4:1. The positive thing with this is that with our current knowledge of successful ROI and failure rate, end of with a rather forgiving number for ROI. On the downside, we see that this sweet spot is very sensitive to fluctuations of both variables. If a more correct average ROI for CBS is down at 5:1 or 6:1, we're suddenly in the "break-even" or "lost-money"-zones, both obviously deterrent to investing in CBS. If the actual failure rate actually is significantly higher than 80%, things look particularly bleak for CBS, because then you need a very high average ROI to make things pay off. If, on the other hand, failure rate is lower, like between 40 and 60 percent, CBS seems like a very sensible venture, with a potential ROI between 3:1 and 4.5:1.

The *potential ROI* is the important number for politicians and policy-makers wanting to know whether to recommend CMM-based SPI as a "general" measure for a given country's software development industry. As of now, the potential ROI for CBS – from what we can know – is about 2.4:1. We see that despite a high potential failure rate of about 70%, these numbers indicate that CBS as a whole *still* is a profitable venture of a nation's software industry. But further research in failure rate, successful ROI and the ROI of failed CBS initiatives is desperately needed to able to make conclusive statements about potential ROI for CBS.

7 Conclusions

We have set out to answer three questions in this thesis:

Q1: To what extent is it possible to learn from case stories of CBS?

In judging the potential for learning from the 71 case studies, we evaluated both the *quality* of the case studies, and how *representative* they were compared to the CMM Maturity Profile released by SEI [28]. In *quality* we measure the *amount of contextual information* provided in a particular case study.

Proximity in context is important for two reasons. First, it is important for learning, because the reader needs to be familiar with context of the case study to simply *understand* what a case study is about. Secondly, it is assumed in software engineering that only in a similar context it will be possible to reproduce the results of a given context.

With regards to *quality*, we found that the amount of contextual information in the articles on average is *low*. This influences *negatively* the reader's ability to make rational choices to whether or not an article is appropriate to learn from. With regards to *representativeness*, we found that the distribution of companies that has committed case studies in general is *not* representative for the typical distribution of companies thought to have commenced in CMM-based improvement as described by the CMM Maturity Profile.

The in general low *quality* of case studies, combined with their low *representativeness*, make them a *dubious tool* for the software community. The studies can at best be rated as *inspiring*, at worst only *entertaining*. Very few, *if any*, of the CMM case studies in the literature can be said to be of *enabling* nature, as to *ensuring* a successful process improvement initiative by following the example sketched in the CMM case study.

It is thus our conclusion to this question that *inspirational learning can be done* from CMM case studies, but it is in the reader's best interest to not to fall in the "trap" of doing CMM-based improvement as "prescribed" in a case study. Because software process improvement is a *socially constructed learning process*, every improvement effort is *unique* to that particular organization – any amount of contextual similarity between organizations can only take you "so far". That said, we regard information about what an organization thinks to be

the most important enablers of a successful improvement effort as useful input in an organization's improvement initiative. These "success factors" vary greatly from case study to case study, therefore we have included a comprehensive list of all enablers we have found reading the articles. Taking into account the many weaknesses of reported success factors, we still believe this list can be inspirational for a SEPG in checking that they are somewhat aligned with what others believe was essential to the success of their improvement initiative.

Q2: What is the rationale for calculating ROI in case stories of CBS?

We found that the rationale for calculating ROI in the case stories is based on widely differing assumptions on how to perform the calculation. On average, the very few case studies that had a calculated ROI did not account for *Hawthorne*-like effects in their calculation. Also, the calculated ROI's come from a very small sample of companies, almost exclusively from the aerospace and defense industry sector. We opt for being *pragmatic* in calculation of ROI as ROI never will be anything else than an "approximate" number anyway. Therefore, average values are of value. Though, it is important to know whether numbers like ROI's of 8.1:1 (as we found) are representative of the industry as a *whole*, or if this is only a feature of the aerospace and defense industry. As a notion for further research, we urge small- to medium-sized companies in civilian sector that have embarked, or will embark, on CBS, to gather relevant data and do their very best to make calculations of ROI, and eventually publish these calculations to the literature.

Q3: Are the costs for CBS higher than the benefits for the industry as a whole?

If the failure rate of 70% in CBS is more or less accurate, and the average ROI of 8.1:1 found in this review is more or less representative, CBS still is more or less profitable for the IT industry as a whole. Even if it does not make sense in the individual sense, the *average* company embarking on CBS can on *average* expect a ROI of about 2:1, accounting for the failure rate. This number is of course not interesting to the individual organization embarking on CBS, but is interesting for policy-makers wanting to increase a given country's IT profitability i.e. by more active approaches towards CMM. It has to be said that this number is extremely sensitive to variations in ROI and failure rate, so if the actual failure rate is higher than 70% (or the actual ROI is lower), the costs for the industry as a whole tend to equal out, or get higher than the benefits. More research in the failure rate of CMM is

desperately needed to justify a further investment in CBS, both on a company scale, and on a industry (country) scale.

We have in this thesis given advice to practitioners (on the rationale for learning from case studies, and a list of reported "success factors" – even if we have emphasized that these should be used with caution), company management (on the rationale for calculation of ROI for CMM) and policy-making authorities (on the profitability of CBS to the industry as a whole). We emphasize that more research into ROI (in both failed and successful CBS initiatives) and failure rate of CBS must be done to confirm or reject the validity of the answers to our two last research questions.

8 References

[1] E. Schaanning, *Kommunikative maktstrategier*. Oslo: Spartacus, 1993.

[2] B. Gabrielsen, *Lutefisk på prærien*. Oslo: Gyldendal, 2001.

[3] Plato, "Theaetetus," in *Plato - Complete Works*, J. M. Cooper, Ed. Indianapolis, Cambridge: Hackett Publishing Company, 1997, pp. 157-234.

[4] Standish Group, "CHAOS report," Standish Group 1995.

[5] Standish Group, "CHAOS report," Standish Group 2003.

[6] M. Jørgensen and K. J. Moløkken-Østvold, "How Large Are Software Cost Overruns? Critical Comments on the Standish Group's CHAOS Reports," *(To be published)*, 2004.

[7] F. P. Brooks, "No Silver Bullet - Essence and Accidents of Software Engineering," *Computer*, vol. 20, pp. 10-19, 1987.

[8] I. Sommerville, *Software Engineering*. Harlow: Addison-Wesley, 2001.

[9] K. Beck, *Extreme Programming Explained*. Boston: Addison-Wesley, 2000.

[10] B. W. Boehm, "A spiral model of software development and enhancement," *IEEE Computer*, vol. 21, pp. 61-72, 1988.

[11] S. P. Keider, "Why systems development projects fail," *Journal of Information Systems Management*, vol. 1, pp. 33-38, 1984.

[12] J. Herbsleb, D. Zubrow, D. Goldenson, W. Hayes, and M. Paulk, "Software quality and the Capability Maturity Model," *Communications of the Acm*, vol. 40, pp. 30-40, 1997.

[13] O. Ngwenyama and P. A. Nielsen, "Competing values in software process improvement: An assumption analysis of CMM from an organizational culture perspective," *IEEE Transactions on Engineering Management*, vol. 50, pp. 100-112, 2003.

[14] T. Dyba, "Enabling Software Process Improvement: An Investigation of the Importance of Organizational Issues. (Doctoral dissertation)." Trondheim: University of Trondheim, 2001.

[15] J. Bach, "Enough About Process - What We Need Are Heroes," *IEEE Software*, vol. 12, pp. 96-98, 1995.

[16] F. W. Taylor, *Principles of Scientific Management*. New York: Norton Library, 1967.

[17] W. E. Deming, *Out of the crisis*. Cambridge: Cambridge University Press, 1986.

[18] R. Aguayo, *Dr. Deming - The American Who Taught the Japanese About Quality*. New York: Carol Publishing Group, 1990.

[19] P. B. Crosby, *Quality is free : The Art of Making Quality Certain*. New York: McGraw-Hill Book Company, 1979.

[20] L. Johnson, "A view from the 1960s: how the software industry began," *Annals of the History of Computing, IEEE*, vol. 20, pp. 36-42, 1998.

[21] G. R. Trimble, Jr., "A brief history of computing. Memoirs of living on the edge," *Annals of the History of Computing, IEEE*, vol. 23, pp. 44-59, 2001.

[22] M. C. Paulk, C. V. Weber, B. Curtis, and M. B. Chrissis, *The Capability Maturity Model : Guidelines for Improving the Software Process*. Indianapolis: Addison-Wesley, 1994.

[23] M. Paulk, B. Curtis, M. Chrissis, and C. Weber, "Capability Maturity Model for Software (Version 1.1) (http://www.sei.cmu.edu/pub/documents/93.reports/pdf/tr24.93.pdf)," SEI Technical Report, Pittsburgh, PA 1993.

[24] SEI, "Compiled list of Organizations who have Publicly Announced their Maturity Levels after having an Appraisal Performed (http://seir.sei.cmu.edu/pml/index.asp)," SEI Report, Pittsburgh, PA 2003.
[25] H. Saiedian and R. Kuzara, "SEI capability maturity model's impact on contractors," *IEEE Computer*, vol. 28, pp. 16-26, 1995.
[26] P. Abrahamsson, "The Role of Commitment in Software Process Improvement. (Doctoral dissertation)." Oulu: University of Oulu, 2002.
[27] C. Debou, "Goal-Based Software Process Improvement Planning," in *Better Software Practice for Business Benefit : Principles and Experiences*, R. Messnarz and C. Tully, Eds. Los Alamitos, California: IEEE Computer Society, 1999.
[28] SEI, "Process Maturity Profile SW-CMM 2003 Year End Update," Carnegie Mellon University 2004.
[29] P. A. Nielsen, "E-mail posting elaborating on calculation done to derive failure rate for CMM in article "Competing Values in Software Process Improvement: An Assumption Analysis of CMM From an Organizational Culture Perspective" (Ngwenyama, O. and Nielsen, P. A., in: IEEE Transactions on Engineering Management 2003). Received by," H. U. Steen, Ed. Oslo, 2003.
[30] M. Jørgensen, "Master thesis supervision with Professor Magne Jørgensen, Simula Research Laboratory," H. U. Steen, Ed. Oslo, 2003.
[31] J. Payne and S. Griffith, "Test Software at Texas Instruments: What SEI Level is Appropriate?," presented at AUTOTESTCON '95. 'Systems Readiness: Test Technology for the 21st Century', 1995.
[32] A. Rainer and T. Hall, "An Analysis of Some 'Core Studies' of Software Process Improvement," *Software Process Improvement and Practice*, vol. 4, pp. 169-187, 2001.
[33] B. Kitchenham and L. Pickard, "Case Studies for Method and Tool Evaluation," *IEEE Software*, vol. 12, 1994.
[34] K. M. Nelson, M. Buche, and M. Ghods, "The journey to IS organizational maturity: a case study of a CMM level 3 IS organization," presented at Challenges of Information Technology Management in the 21st Century. 2000 Information Resources Management Association International Conference, 2000.
[35] J. Batista and A. D. de Figueiredo, "CMM in a Micro Team: A Case Study," presented at EuroSPI, 1998.
[36] P. Jalote, "Lessons Learned in Framework-Based Software Process Improvement," presented at Proceedings of the Ninth Asia-Pacific Software Engineering Conference, 2002.
[37] D. E. Harter, M. S. Krishnan, and S. A. Slaughter, "Effects of process maturity on quality, cycle time, and effort in software product development," *Management Science*, vol. 46, pp. 451-466, 2000.
[38] B. van der Wal, "Life on Level 5," presented at EuroSPI, 2001.
[39] B. A. Kitchenham, S. L. Pfleeger, L. M. Pickard, P. W. Jones, D. C. Hoaglin, K. El Emam, and J. Rosenberg, "Preliminary guidelines for empirical research in software engineering," *IEEE Transactions on Software Engineering*, vol. 28, pp. 721-734, 2002.
[40] P. M. Wortman, "Judging research quality," in *The Handbook of Research Synthesis*, H. Cooper and L. V. Hedges, Eds. New York: Russell Sage Foundation, 1994.
[41] IEE, "INSPEC database (http://www.iee.org/publish/inspec/)," 2004.
[42] STSC, "Crosstalk - The Journal of Defense Software Engineering," U.S. Air Force Software Technology Support Center, 2004.

[43] D. M. Raffo, J. V. Vandeville, and R. H. Martin, "Software process simulation to achieve higher CMM levels," *Journal of Systems and Software*, vol. 46, pp. 163-172, 1999.
[44] R. Oshana and F. P. Coyle, "Implementing cleanroom software engineering into a mature CMM-based software organization," presented at Proceedings of the 1997 International Conference on Software Engineering, ICSE-97, 1997.
[45] Crosstalk, "NSWC PHD Dam Neck Receives CMM Level 3 Rating," *Crosstalk*, 1998.
[46] R. Dion, "Elements of a Process-Improvement Program," *IEEE Software*, vol. 9, pp. 83-85, 1992.
[47] T. Mehner, T. Messer, P. Paul, F. Paulisch, P. Schless, and A. Volker, "Siemens process assessment and improvement approaches: experiences and benefits," presented at Proceedings of The Twenty Second Annual International Computer Software and Applications Conference, Compsac '98, 1998.
[48] C. D. Buchman, "Software process improvement at AlliedSignal Aerospace," presented at Proceedings of the Twenty Ninth Hawaii International Conference on System Sciences, 1996.
[49] D. L. Johnson and J. G. Brodman, "Applying CMM project planning practices to diverse environments," *IEEE Software*, vol. 17, pp. 40-7, 2000.
[50] D. Stelzer and W. Mellis, "Success factors of organizational change in software process improvement," *Software Process: Improvement and Practice*, vol. 4, 1998.
[51] M. L. Wald and J. Schwartz, "Shuttle Inquiry Uncovers Flaws in Communication," in *The New York Times*, 2003.
[52] H.-G. Gadamer, *Truth and Method*. London: Sheed & Ward, 1975.
[53] W. A. Stock, "Systematic coding for research synthesis," in *The Handbook of Research Synthesis*, H. Cooper and L. V. Hedges, Eds. New York: Russell Sage Foundation, 1994.
[54] G. Amsjø, "Kongsberg Defence & Aerospace - The only Norwegian company at CMM level 3? (Norwegian text) (http://www.abelia-innovasjon.no/page/main.php?content=643&instance=1686)," in *Abelia Innovasjon News*, 2004.
[55] F. McGarry and B. Decker, "What Is a Level 5?," presented at Proceedings of the 26th Annual NASA Goddard Software Engineering Workshop, 2002.
[56] T. Westaway, "How We Achieved Level 3," *Crosstalk*, 1995.
[57] H. Krasner and G. Scott, "Lessons Learned from an Initiative for Improving Software Process, Quality and Reliability in a Semiconductor Equipment Company," presented at Proceedings of the Twenty Ninth Hawaii International Conference on System Sciences, 1996.
[58] G. Kenni, "The evolution of quality processes at Tata Consultancy Services," *IEEE Software*, vol. 17, pp. 79-88, 2000.
[59] C. Billings, J. Clifton, B. Kolkhorst, E. Lee, and W. B. Wingert, "Journey to a mature software process," *IBM Systems Journal*, vol. 33, pp. 46-61, 1994.
[60] W. H. Lipke and K. L. Butler, "Software Process Improvement: A success story," *Crosstalk*, 1992.
[61] N. Eickelmann, "An insider's view of CMM level 5," *IEEE Software*, vol. 20, pp. 79-81, 2003.
[62] M. Diaz and J. Sligo, "How software process improvement helped Motorola," *IEEE Software*, vol. 14, pp. 75-81, 1997.
[63] Software Productivity Consortium, "Frameworks Quagmire (http://www.software.org/quagmire/)," Software Productivity Consortium, 2004.
[64] IEEE, "IEEE Xplore - Document Archive (http://ieeexplore.ieee.org/)," 2004.

[65] ACM, "ACM Digital Library (http://www.acm.org/dl)," 2004.
[66] F. Guerrero and Y. Eterovic, "Adopting the SW-CMM in a Small IT Organization," *IEEE Software*, vol. 21, pp. 29-35, 2004.
[67] Thomson ISI, "ISI Web of Knowledge (http://www.isiknowledge.com)," 2004.
[68] W. S. Humphrey, T. R. Snyder, and R. R. Willis, "Software Process Improvement at Hughes-Aircraft," *IEEE Software*, vol. 8, pp. 11-23, 1991.
[69] SEI, "CMMI Web Site (http://www.sei.cmu.edu/cmmi/cmmi.html)," Carnegie Mellon University, 2004.
[70] SEI, "How Will Sunsetting of the Software CMM® Be Conducted (http://www.sei.cmu.edu/cmmi/adoption/sunset.html)," 2001.
[71] SEI, "Process Maturity Profile CMMI v1.1 2003 Year End Update," Carnegie Mellon University. 2004.
[72] K. Schwaber and M. Beedle, *Agile Software Development with SCRUM*: Prentice Hall, 2001.
[73] Wiley, "Software Process: Improvement and Practice (journal) (http://www3.interscience.wiley.com/cgi-bin/jhome/15482)," 2002.
[74] P. C. Oestreich and D. R. Webb, "The Race to Level 3," *Crosstalk*, 1995.
[75] P. W. Cosgriff, "The Journey to CMM Level 5: A Time Line," *Crosstalk*, 1999.
[76] P. W. Cosgriff, "The Right Things for the Right Reasons - Lessons Learned Achieving CMM Level 5," *Crosstalk*, 1999.
[77] L. G. Oldham, D. B. Putman, M. Peterson, B. Rudd, and K. Tjoland, "Benefits Realized from Climbing the CMM Ladder," *Crosstalk*, 1999.
[78] G. Yamamura and G. B. Wigle, "SEI CMM Level 5: For the Right Reasons," *Crosstalk*, vol. August, 1997.
[79] G. B. Wigle and G. Yamamura, "Practices of an SEI CMM Level 5 SEPG," *Crosstalk*, 1997.
[80] M. Kimsey and J. Fowler, "SEI CMM Level 5: A practitioner's Perspective," *Crosstalk*, 1997.
[81] G. P. Fulton, "SEI CMM Level 5: Lightning Strikes Twice ("Boeing Space Transportation Systems-case")," *Crosstalk*, 2002.
[82] K. A. Butler, "The Economic Benefits of Software Process Improvement," *Crosstalk*, vol. July, 1995.
[83] K. Butler, "Process Lessons Learned While Reaching Level 4," *Crosstalk*, 1997.
[84] F. McGarry and B. Decker, "Attaining level 5 in CMM process maturity," *IEEE Software*, vol. 19, pp. 87-+, 2002.
[85] F. Hara, "Irish Experiences with Software Process Improvement ("Motorola-case")," presented at EuroSPI, 2001.
[86] B. Fitzgerald and T. O'Kane, "A longitudinal study of software process improvement," *IEEE Software*, vol. 16, pp. 37-+, 1999.
[87] M. K. Daskalantonakis, "Achieving higher SEI Levels," *IEEE Software*, vol. 11, pp. 17-24, 1994.
[88] F. Hara, "Irish Experiences with Software Process Improvement ("S3-case")," presented at EuroSPI, 2001.
[89] T. J. Haley, "Software process improvement at Raytheon," *IEEE Software*, vol. 13, pp. 33-41, 1996.
[90] A. Billi, "An Experience of SEPG Organization," presented at EuroSPI, 1998.
[91] R. Dion, "Process Improvement and the Corporate Balance-Sheet," *IEEE Software*, vol. 10, pp. 28-35, 1993.
[92] M. Wakulczyk, "Success Is Not Accidental: CMM Level 2 in 2.5 Years," *Crosstalk*, 1997.

[93] A. T. Steadman, "USDA's National Finance Center: A 10-Month Journey to Level 2," *Crosstalk*, 1999.
[94] M. Diaz and J. Sligo, "How CMM Impacts Quality, Productivity, Rework and the Bottom Line," *Crosstalk*, 2002.
[95] B. Hefley, J. Schwalb, and L. Pracchia, "AV-8B's Experience Using the TSP to Accelerate SW-CMM Adoption," *Crosstalk*, 2002.
[96] G. P. Fulton, "SEI CMM Level 5: Lightning Strikes Twice ("Boeing Military Aircraft and Missiles Seattle Site-case")," *Crosstalk*, 2002.
[97] G. Natwick, "Integrating Metrics for CMMI and SW-CMM," *Crosstalk*, 2003.
[98] D. Damian, D. Zowghi, L. Vaidyanathasamy, and Y. Pal, "An industrial experience in process improvement: an early assessment at the Australian Center for Unisys Software," presented at Proceedings of the 2002 International Symposium on Empirical Software Engineering, 2002.
[99] S. K. Schaffner and K. S. White, "Software engineering practices for control system reliability," presented at Proceedings of the 1999 Particle Accelerator Conference, New York, 1999.
[100] T. Cromer and J. Horch, "From the many to the one - one company's path to standardization," presented at ISESS '99. Proceedings of the 4th IEEE International Software Engineering Standards Symposium and Forum., 1999.
[101] C. F. Machado, L. C. De Oliveira, and R. A. Fernandes, "Experience Report - Restructure of Processes based on ISO/IEX 12207 and SW-CMM in CELEPAR," presented at ISESS '99. Proceedings of the 4th IEEE International Software Engineering Standards Symposium and Forum., 1999.
[102] J. Rooijmans, H. Aerts, and M. vanGenuchten, "Software quality in consumer electronics products," *IEEE Software*, vol. 13, pp. 55-&, 1996.
[103] A. Johnson, "Software Process Improvement Experience in the DP/MIS Function," presented at ICSE 16. 16th International Conference on Software Engineering, 1994.
[104] G. Seshagiri, "Continuous process improvement - why wait till Level 5?," presented at Proceedings of the Twenty Ninth Hawaii International Conference on System Sciences, 1996.
[105] C. Hollenbach and R. Young, "Combining quality and software improvement," *Communications of the ACM*, vol. 40, pp. 41-45, 1997.
[106] R. H. Lordahl, "Starting a large scale software development project using the SEI CMM as the guiding vision," presented at Canadian Conference on Electrical and Computer Engineering, 1994.
[107] C. Y. Laporte and N. R. Papiccio, "Software and systems engineering process development and integration at Oerlikon Aerospace," *Software Process Newsletter*, pp. 10-17, 1998.
[108] R. Zahniser and D. Rizzo, "Software process reengineering. Getting to `Level 3' with teamwork," *American Programmer*, vol. 8, pp. 36-43, 1995.
[109] S. Zahran and K. Sanders, "A software process improvement framework: the theory and practice at Bull Information Systems," presented at SPI 95. The European Conference on Software Process Improvement. The European Experience in a World Context, 1995.
[110] S. Benno and D. Frailey, "Software process improvement in DSEG 1989-1995," *Texas Instruments Technical Journal*, vol. 12, pp. 20-28, 1995.
[111] M. Ishio, "Example of field quality improvement by raising the level of CMM (Capability Maturity Model)," presented at Sixteenth Annual Pacific Northwest Software Quality Conference Joint ASQ Software Division's Eighth International Conference on Software Quality, 1998.

[112] B. Dunseath, "Software process improvement for knowledge engineering," presented at Quality Challenge. Fifth International Conference on Software Management, 1997.
[113] D. Flam and V. Radotovic, "TQM in telecommunication software design," presented at ConTEL 97. 4th International Conference on Telecommunications, 1997.
[114] A. Sampson, "Utilizing the capability maturity model for software to achieve and retain ISO 9001 certification," presented at Fourteenth Annual Pacific Northwest Software Quality Conference, 1996.
[115] C. Knüvener, "A CMM Project Takes Off - A Project Milestone Report," presented at EuroSPI, 2003.
[116] D. E. Lowe and G. M. Cox, "Implementing the Capability Maturity Model for Software Development," *Hewlett Packard Journal*, vol. 47, pp. 6-14, 1996.
[117] D. P. Kelly and B. Culleton, "Process improvement for small organizations," *Computer*, vol. 32, pp. 41-+, 1999.
[118] G. C. Thomas and H. R. Smith, "Using structured benchmarking to fast-track CMM process improvement," *IEEE Software*, vol. 18, pp. 48-+, 2001.
[119] C. Y. Laporte and S. Trudel, "Addressing the People Issues of Process Improvement Activities at Oerlikon Aerospace," *Software Process Improvement and Practice*, vol. 4, pp. 187-198, 1998.
[120] P. Bowers, "Raytheon Stands Firm on Benefits of Process Improvement," *Crosstalk*, 2001.
[121] S. Otoya and C. Narciso, "An Experience: A Small Software Company Attempting to Improve its Process," presented at STEP '99. Proceedings of the Ninth International Workshop of Software Technology and Engineering Practice, 1999.
[122] A. Florence, "Lessons Learned in Attempting to Achieve Software CMM Level 4," *Crosstalk*, 2001.
[123] D. W. Drew, "Tailoring the Software Engineering Institute's (SEI) Capability Maturity Model (CMM) to a Software Sustaining Engineering Organization," presented at Conference on Software Maintenance, 1992.
[124] H. Wohlwend and S. Rosenbaum, "Schlumberger's Software Improvement Program," *IEEE Transactions on Software Engineering*, vol. 20, 1994.
[125] F. Cattaneo, A. Fuggetta, and L. Lavazza, "An experience in process assessment," presented at 17th International Conference on Software Engineering, 1995.
[126] N. L. M. Davies and M. M. Dumont, "SEI-based process improvement efforts at Digital," *Digital Technical Journal*, vol. 5, pp. 59-68, 1993.
[127] R. Francis, "Quality and process management: a view from the UK computing services industry," *Software Quality Journal*, vol. 2, pp. 225-38, 1993.
[128] J. Batista and A. D. de Figueiredo, "SPI in a very small team: a case with CMM," *Software Process Improvement and Practice*, vol. 4, pp. 243-50, 2000.
[129] S. E. Miller, "Process synergy: using ISO 9000 and the CMM in a small development environment," presented at Sixteenth Annual Pacific Northwest Software Quality Conference Joint ASQ Software Division's Eighth International Conference on Software Quality, 1998.
[130] T. Dyba, "Factors of software process improvement success in small and large organizations: an empirical study in the scandinavian context," *ACM SIGSOFT Software Engineering Notes*, vol. 28, 2003.
[131] M. E. Fayad, M. Laitinen, and R. P. Ward, "Thinking objectively: software engineering in the small," *Communications of the ACM*, vol. 43, pp. 115-118, 2000.
[132] EFQM, "EFQM Excellence Model (http://www.efqm.org/)," EFQM, 2004.

[133] V. Basili, "The experience factory and its relationship to other improvement paradigms," presented at 4th European Software Engineering Conference (ESEC), LCNS 717, 1993.
[134] D. Goldenson and J. D. Herbsleb, "After the appraisal: a systematic survey of process improvement, its benefits, and factors that influence success," SEI, Carnegie Mellon University, Pittsburgh, Pennsylvania 1995.
[135] T. Hall, A. Rainer, and N. Baddoo, "Implementing software process improvement: an empirical study," *Software Process Improvement and Practice*, vol. 7, pp. 3-15, 2002.
[136] K. E. Emam and D. R. Goldenson, "An Empirical Review of Software Process Assessments," *Advances in Computers*, vol. 53, pp. 319-423, 2000.
[137] A. Rainer and T. Hall, "Key success factors for implementing software process improvement: a maturity-based analysis," *Journal of Systems and Software*, vol. 62, pp. 71-84, 2002.
[138] K. Lewin, *Field theory in social science : selected theoretical papers*. New York: Harper & Brothers, 1951.
[139] W. Menezes and B. Eschermann, "Setting up SPI in a multi-cultural and de-centralised engineering company," presented at Second Annual European Software Engineering Process Group Conference, Amsterdam, 1997.
[140] H. Simon, *Models of man : social and rational : mathematical essays on rational human behavior in a social setting*. New York: Wiley, 1957.
[141] C. Dekkers, "Cultural Obstacles to Process Maturity (copy freely available online at the author's employer's web-site: http://www.qualityplustech.com/cutter_it_Apr2001.html)," in *The Cutter IT Journal E-Mail Advisor*, 2001.
[142] J. P. Near and M. P. Miceli, "Whistle-blowing: Myth and reality," *Journal of Management*, vol. 22, pp. 507-526, 1996.
[143] O. Neurath, "Protokollsätze," *Erkenntnis zugleich Annalen der Philosophie*, vol. 3, pp. 204-214, 1932.
[144] J. Herbsleb, A. Carleton, J. Rozum, J. Siegel, and D. Zubrow, "Benefits of CMM-Based Software Process Improvement: Executive Summary of Initial Results," SEI, Carnegie Mellon University, Pittsburgh, Pennsylvania 1994.
[145] J. G. Broadman and D. L. Johnson, "Return on Investment from Software Process Improvement as Measured by U.S. Industry," *Crosstalk*, 1996.
[146] R. Solingen, "Measuring the ROI of Software Process Improvement," *IEEE Software*, vol. 21, pp. 32-38, 2004.
[147] S. A. Sheard, "The Shangri-La of ROI," *Software Productivity Consortium NFP*, 2000.
[148] E. Mayo, *The human problems of an industrial civilization*. New York: McMillian, 1933.
[149] B. Rice, "The Hawthorne defect: Persistence of a flawed theory," *Psychology Today*, vol. 16, pp. 70-74, 1982.
[150] F. Cattaneo, A. Fuggetta, and D. Sciuto, "Pursuing Coherence in Software Process Assessment and Improvement," *Software Process Improvement and Practice*, vol. 6, pp. 3-22, 2001.

9 Appendix A – Common abbreviations

Acronym	*Explanation*
CBS	CMM-based SPI
CMM	Capability Maturity Model - when used alone, it refers to the SW-CMM
CMM-I	Capability Maturity Model Integration – the effort from SEI that took the place of SW-CMM v2.0, and is a combination of several other maturity models in addition to SW-CMM, see [69]
DOD	U.S. Department of Defense (also DoD)
KLOC	Kilo Lines Of Code = 1000 lines of source code - standard measure in software engineering.
KPA	Key Process Area
ROI	Return On Investment
SED	Software Engineering Division (commonly used in company scopes – see section 4.2.8)
SEI	Software Engineering Institute (at Carnegie Mellon University) – the origin of the CMM
SEPG	Software Engineering Process Group - group of people responsible for coordinating the SPI effort in an organization
SPI	Software Process Improvement
SW-CMM	Software CMM – refers to the first published maturity model from SEI, see [22]

10 Appendix B – Data

10.1 Introduction

For purposes of reduction of space conservation and printability, only the 22 categories necessary to reproduce the analyses in the thesis has been included (as pointed out in section 3.5.2, 29 different categories were initially coded.) In 10.2 a legend to the coding categories is given, and in 10.3 a description of the variable values for those variables that have "non-intuitive" values. The coded data is presented in section 10.4.

10.2 Legend to the coding categories

The following characters can be shown in the "special" field, with the appropriate explanation:

M = *variable is multi-response*
V = *see below for specification of variable contents*

Label	*Explanation*	*Special*
#AU1	Author affiliation of article	M V
#AU2	Author affiliation of article	M V
#JO	Place of publish	V
#YE	Year of publish	
#SU	Reported success?	V
#CO	Country of scope	V
#IN1	Industry sector	M V
#IN2	Industry sector	M V
#SC	Company and scope	V
#DS	Number of developers in scope	V
#ROI	Reported ROI [x:1]	
#INV	Information on investment reported in article	V
#OE	Information on other effects reported in article	V
#E	Number of enablers reported in article	
#D	Number of disablers reported in article	
#PA	Parallel standards mentioned in article	V
#ST	Starting CMM level	
#EN	Ending CMM level	
#STY	Starting year	
#ENY	Ending year	
#SEI	SEI involvement	V
#Q	Quality index score for article	(See section 4.3)
#CYL	Calculated years per level	(See section 4.2.15)

10.3 Explanation of non-intuitive variable values

#AU1 and #AU2:
- 1 = 'From same company'
- 2 = 'University or other external research facility employee(s)'
- 3 = 'Journal/magazine editorial staff'
- 4 = 'External consultant(s)'
- 5 = 'Employees of SEI or Carnegie Mellon University'
- 6 = 'External to company (unspecified)'
- 7 = '(unknown)'
- 8 = 'External - formerly employed in company'

#JO:
- 1 = 'IEEE Software'
- 2 = 'Software Process Improvement and Practice'
- 3 = 'IEEE Transactions on Software Engineering'
- 4 = 'IEEE Computer'
- 5 = 'Crosstalk'
- 6 = 'Other conference proceedings'
- 7 = 'Communications of the ACM'
- 8 = 'Company-linked technical journal'
- 9 = 'Software Quality Journal'
- 10 = 'Management Science'
- 11 = 'Software Process Newsletter'
- 12 = 'American Programmer'

#SU:
- 0 = 'no'
- 1 = 'yes'

#CO:
- 1 = 'USA'
- 2 = 'India'
- 3 = 'Ireland'
- 4 = 'Canada'
- 5 = 'Italy'
- 6 = 'Australia'
- 7 = 'Brazil'
- 8 = 'USA and Israel'
- 9 = 'The Netherlands'
- 10 = 'UK'
- 11 = 'Japan'
- 12 = 'USA and India'
- 13 = 'Germany'
- 14 = 'Croatia'

#IN1 and #IN2:
- 1 = 'Telecommunications'
- 2 = 'Aerospace'
- 3 = 'Missing'
- 4 = 'Defense'
- 5 = 'ICT Consulting'
- 6 = 'Generic software and hardware'
- 7 = 'US Government subcontractor'
- 8 = 'US Air Force in-house'
- 9 = 'US Government in-house'
- 10 = 'Semi-conductor related'
- 11 = 'Oil-related'
- 12 = 'Investment management'
- 13 = 'Automotive'

#SC:
1 = 'Motorola Cellular Department'
2 = 'SEAS / Computer Sciences Corporation'
3 = 'Hughes Aircraft SED'
4 = 'Atos Origin India'
5 = 'Motorola Cork'
6 = 'S3 Silicon and Software Systems'
7 = 'Motorola Government Electronics Division'
8 = 'Raytheon Electronic Systems'
9 = 'Sodalia developer department'
10 = 'Raytheon Software systems laboratory'
11 = 'Sacramento Air Logistics Center SED'
12 = 'Ogden Air Logistics Center SED'
13 = 'Oklahoma City Air Logistics Center SED'
14 = 'Boeing Defense and Space Group - Space Transportation System'
15 = 'North American Aerospace Defense - System Support Facility'
16 = 'US Department of Agriculture - National Finance Center'
17 = 'General Dynamixs Decision System SED'
18 = 'Naval Air Systems Command - AV-8B Joint System Support Activ'
19 = 'Boeing Military Aircraft and Missiles - Seattle Site'
20 = 'Harris Corporation - Government Communications Systems Div'
21 = 'Unisys - Australian Center for Unisys Software (ACUS)'
22 = 'Thomas Jefferson National Accelerator - Controls Group'
23 = 'COLSA Corporation - Software development departments'
24 = 'CELEPAR - All software employees'
25 = 'Texas Instruments Test Engineering Department ...'
26 = 'Philips Electronics - Philips TV division'
27 = 'Corning Incorporated - Information Services Division'
28 = 'Advanced Information Services - (organization-wide)'
29 = 'Litton PRC - Systems Integration Unit (and more)'
30 = 'Unisys GDG - New Shipborn Aircraft Program'
31 = 'NASA - Space Shuttle Onboard Software Project'
32 = 'Oerlikon Aerospace - Laser-guided missile-air defence system'
33 = 'AIM Management Group Inc - Telecomm. and Systems dep.'
34 = 'Bull Information Systems - (organization-wide)'
35 = 'Texas Instruments - Defense Systems and Electronics Group'
36 = 'NEC - Communication Systems'
37 = 'Bombardier Aerospace Group - Short Brothers ICAD project'
38 = 'Ericsson Nikola Tesla - Software Design Centre'
39 = 'Tektronix, Inc. - Product instrumentation group ...'
40 = 'Method Park Software AG - Software development department'
41 = 'Hewlett Packard - "One HP division"'
42 = 'Silicon and Software Systems - Software department'

#DS:
1 = '1-4'
2 = '5-9'
3 = '10-24'
4 = '25-49'
5 = '50-99'
6 = '100-199'
7 = '200-299'
8 = '300-399'
9 = '400-499'
10 = '500-999'
11 = '1000+'

#INV and #OE:
0 = 'no'

1 = 'yes'
2 = 'missing'

#PA:
1 = 'ISO 9001'
2 = 'TSP'
3 = 'CMM-I and GQM'
4 = 'ISO/IEC 12207'
5 = 'ISO 9000-3'
6 = 'PSP'
7 = 'TQM'
8 = 'QIP'
9 = 'ISO 9001 and EFQM'
10 = 'DOD 2167A'

#SEI:
0 = 'no indication of SEI involvement whatsoever'
1 = 'suspected SEI involvement - but only non-conclusive evidence'
2 = 'SEI involvement by a high degree of likelihood'
3 = 'SEI involvement - explicit mention in article'

10.4 The coded data

Title	Author / Year	#AU1:	#AU2:	#JO:	#YEA:	#SU:	#CO:	#IN1:	#IN2:	#SC:	#DS:	#ROI:	#INV:	#OE:	#E:	#D:	#PA:	#ST:	#EN:	#STY:	#ENY:	#SEI:	#Q:	#CYL:
Achieving higher SEI levels	Daskalantonakis1994	1	0	1	1994	1	0	1	1	1	11	0.00	0	0	1	0	0	1.0	2.0	1992	1993	0	7	1.00
An insider's view of CMM level 5	Eickelman2003	3	0	1	2003	1	2	3	3	0	0	2.50	2	2	4	0	0	0.0	5.0			0	4	
Attaining Level 5 in CMM process maturity	McGarry2002	1	0	1	2002	1	0	2	2	2	10	0.00	1	1	11	0	1	0.0	5.0	1994	1998	0	10	
Software process improvement at Hughes Aircraft	Humphrey1991	5	7	1	1991	1	1	4	2	3	10	0.00	1	1	3	0	0	2.0	3.0	1987	1990	3	9	3.00
Life on Level 5	vanderWal2001	1	0	6	2001	1	2	5	5	4	9	0.00	1	1	6	0	1	1.0	5.0	1996	1999	0	10	0.75
Irish Experiences with Software Process Improvement	Hara2001	4	0	6	2001	1	3	1	1	5	9	0.00	0	0	2	0	0	0.0	4.0		1997	0	4	
Irish Experiences with Software Process Improvement	Hara2001b	4	0	6	2001	1	3	6	6	6	6	0.00	0	0	5	0	1	0.0	2.0		1999	0	4	
How software process improvement helped Motorola	Diaz1997	1	0	1	1997	1	0	7	4	7	8	6.77	2	2	9	0	0	2.0	5.0	1989		1	8	
Using structured benchmarking to fast-track CMM process improvement	Thomas2001	1	0	1	2001	1	1	2	4	0	0	0.00	0	0	1	0	0	0.0	3.0	1998		0	6	
The evolution of quality processes at Tata Consultancy Services	Kenni2000	1	0	1	2000	1	12	5	5	0	10	0.00	0	1	0	0	1	0.0	5.0	1996	1999	3	5	
Software process improvement at Raytheon	Haley1996	1	0	1	1996	1	1	2	4	8	11	0.00	1	1	0	0	0	0.0	3.0		1991	0	4	
Addressing the People Issues of Process Improvement Activities at Oerlikon Aerospace	Laporte1998	1	4	2	1998	1	4	2	2	0	0	0.00	0	0	15	0	0	1.0	2.0		1997	0	4	
A longitudinal study of software process improvement	Fitzgerald1999	1	2	1	1999	1	3	1	1	5	8	0.00	0	0	0	0	0	1.0	4.0	1993	1997	0	5	1.33
CMM in a Micro Team: A Case Study	Batista1998	2	0	6	1998	1	0	3	3	0	3	0.00	0	1	1	0	0	1.0	1.5			0	6	
An Experience of SEPG Organization	Billi1998	1	0	6	1998	1	5	3	3	9	7	0.00	0	0	1	1	1	2.0	3.0	1995	1997	0	8	2.00
Process Improvement and the Corporate Balance Sheet	Dion1993	1	0	1	1993	1	1	2	4	10	9	7.70	2	2	1	0	0	1.0	3.0	1988	1991	3	10	1.50
How We Achieved Level 3	Westaway1995	1	0	5	1995	1	1	8	8	11	0	0.00	0	0	1	0	0	1.0	2.5	1991		0	4	
The Race to Level 3	Oestreich1995	1	0	5	1995	1	1	8	8	12	9	0.00	0	0	0	0	0	1.5	3.0	1992	1995	0	5	2.00
The Economic Benefits of Software Process Improvement	Butler1995	1	0	5	1995	1	1	8	8	13	9	7.50	2	2	0	0	0	1.0	2.0	1990	1993	3	8	3.00
Process Lessons Learned While Reaching Level 4	Butler1997	1	0	5	1997	1	1	8	8	13	8	0.00	0	0	10	1	0	1.0	4.5	1990	1996	0	10	1.71
SEI CMM Level 5: For the Right Reasons	Yamamura1997	1	0	5	1997	1	1	2	2	14	0	7.75	2	2	0	0	0	1.0	5.0		1996	3	4	

Title	Author / Year	#AU1:	#AU2:	#JO:	#YEA:	#SU:	#CO:	#IN1:	#IN2:	#SC:	#DS:	#ROI:	#INV:	#OE:	#E:	#D:	#PA:	#ST:	#EN:	#STY:	#ENY:	#SEI:	#Q:	#CYL:
SEI CMM Level 5: A practitioner's Perspective	Kimsey1997	1	0	5	1997	1	1	2	2	14	0	0.00	0	0	0	0	0	0.0	5.0		1996	1	2	
Success Is Not Accidental: CMM Level 2 in 2.5 Years	Wakulczyk1997	1	0	5	1997	1	1	8	8	15	6	0.00	1	0	8	0	0	1.0	2.0		1996	0	9	
Practices of an SEI CMM Level 5 SEPG	Wigle1997	1	0	5	1997	1	1	2	2	14	0	0.00	0	0	5	0	0	0.0	5.0		1996	0	4	
The Journey to CMM Level 5: A Time Line	Cosgriff1999	1	0	5	1999	1	1	8	8	12	0	0.00	0	0	5	0	0	1.0	5.0	1991	1998	1	7	1.75
Benefits Realized from Climbing the CMM Ladder	Oldham1999	1	0	5	1999	1	1	8	8	12	0	19.00	2	2	0	0	0	0.0	0.0			0	2	
The Right Things for the Right Reasons - Lessons Learned Achieving CMM Level 5	Cosgriff1999b	1	0	5	1999	1	1	8	8	12	0	0.00	0	0	2	0	0	0.0	0.0			0	2	
USDA's National Finance Center: A 10-Month Journey to Level 2	Steadman1999	1	0	5	1999	1	1	9	9	16	0	0.00	0	0	3	0	0	1.0	2.0	1997	1998	0	7	1.00
Raytheon Stands Firm on Benefits of Process Improvement	Bowers2001	4	0	5	2001	1	1	2	4	0	0	6.00	2	2	2	0	0	0.0	4.0		1998	1	5	
Lessons Learned in Attempting to Achieve Software CMM Level 4	Florence2001	4	0	5	2001	0	0	3	3	0	0	0.00	0	0	0	7	0	3.0	3.0			0	3	
How CMM Impacts Quality, Productivity, Rework and the Bottom Line	Diaz2002	1	0	5	2002	1	1	2	4	17	8	0.00	1	1	5	0	0	1.0	5.0		2001	0	6	
AV-8B's Experience Using the TSP to Accelerate SW-CMM Adoption	Hefley2002	1	5	5	2002	1	1	2	4	18	2	0.00	0	0	1	0	2	1.0	2.0		2001	3	7	
SEI CMM Level 5: Lightning Strikes Twice	Fulton2002	1	0	5	2002	1	1	2	4	19	0	0.00	0	0	2	0	0	3.0	5.0	2000	2001	0	7	0.50
SEI CMM Level 5: Lightning Strikes Twice	Fulton2002b	1	0	5	2002	1	1	2	4	14	0	0.00	0	0	0	0	0	3.0	5.0		1996	0	5	
Integrating Metrics for CMMI and SW-CMM	Natwick2003	1	0	5	2003	1	1	7	7	20	0	0.00	0	0	0	0	3	0.0	4.0		2002	0	2	
Process Improvement for Small Organizations	Kelly1999	1	8	4	1999	1	3	10	10	42	6	0.00	1	0	1	0	1	1.5	1.5	1997		0	5	
An industrial experience in process improvement: an early assessment at the Australian Center for Unisys Software	Damian2002	1	2	6	2002	1	6	5	5	21	0	0.00	0	1	2	0	1	1.0	2.0	2001	2002	0	8	1.00
Lessons Learned in Framework-Based Software Process Improvement	Jalote2002	2	0	6	2002	1	2	3	3	0	0	0.00	1	0	12	0	0	0.0	0.0			0	2	
What Is a Level 5?	McGarry2002b	1	0	6	2002	1	1	2	2	2	10	0.00	0	1	0	0	0	1.0	5.0	1994	1998	0	6	1.00
An Experience: A Small Software Company Attempting to Improve its Process	Otoya1999	1	2	6	1999	1	6	5	5	0	0	0.00	0	1	0	0	0	1.0	1.5	1996		0	2	
Software engineering practices for control system reliability	Schaffner1999	1	0	6	1999	1	1	9	9	22	0	0.00	0	1	3	0	0	1.0	0.0			0	5	
From the many to the one - one company's path to standardization	Cromer1999	1	0	6	1999	1	1	6	6	23	5	0.00	0	0	3	0	1	1.0	3.0			0	4	

Title	Author / Year	#AU1:	#AU2:	#JO:	#YEA:	#SU:	#CO:	#IN1:	#IN2:	#SC:	#DS:	#ROI:	#INV:	#OE:	#E:	#D:	#PA:	#ST:	#EN:	#STY:	#ENY:	#SEI:	#Q:	#CYL:
Experience Report - Restructure of Processes based on ISO/IEX 12207 and SW-CMM in CELEPAR	Machado1999	1	0	6	1999	1	7	5	5	24	6	0.00	0	0	0	0	4	0.0	0.0			0	0	
Lessons Learned from an Initiative for Improving Software Process, Quality and Reliability in a Semiconductor Equipment Company	Krasner1996	1	4	6	1996	1	8	10	10	0	0	0.00	0	0	7	9	0	0.0	0.0			0	5	
Tailoring the Software Engineering Institute's (SEI) Capability Maturity Model (CMM) to a Software Sustaining Engineering Organization	Drew1992	1	0	6	1992	1	1	2	2	0	0	0.00	0	0	0	0	0	0.0	0.0			0	-1	
Test Software at Texas Instruments: What SEI Level is Appropriate?	Payne1995	1	0	6	1995	1	1	2	4	25	0	0.00	0	1	5	0	0	1.0	2.0	1989		0	5	
Software Quality in Consumer Electronics Products	Rooijams1996	1	8	1	1996	1	9	6	6	26	0	0.00	0	1	0	0	0	0.0	1.5	1988	1993	0	6	
Schlumberger's Software Improvement Program	Wohlwend1994	1	8	3	1994	1	1	11	11	0	0	0.00	0	1	8	2	5	0.0	0.0	1990		0	5	
Software Process Improvement Experience in the DP/MIS Function	Johnson1994	1	0	6	1994	1	1	10	10	27	0	0.00	0	1	5	0	0	0.0	0.0	1991		3	3	
Continuous process improvement-why wait till Level 5?	Seshagiri1996	1	0	6	1996	1	1	5	5	28	0	0.00	1	1	2	0	6	1.0	1.5	1992		3	6	
Combining quality and software improvement	Hollenbach1997	1	0	7	1997	1	1	2	4	29	0	0.00	1	1	1	0	7	1.0	3.0		1996	0	9	
Starting a large scale software development project using the SEI CMM as the guiding vision	Lordahl1994	1	0	6	1994	1	4	4	4	30	6	0.00	0	0	0	0	0	0.0	0.0	1989		0	0	
An experience in process assessment	Cattaneo1995	2	0	6	1995	0	5	5	5	0	0	0.00	0	0	0	0	8	0.0	0.0			0	-1	
Journey to a mature software process	Billings1994	8	6	8	1994	1	1	2	2	31	0	0.00	0	1	0	0	0	3.5	5.0	1984	1989	3	6	3.33
SEI-based process improvement efforts at Digital	Davies1993	1	0	8	1993	1	1	5	6	0	0	0.00	0	0	2	2	0	0.0	0.0			1	3	
Quality and process management: a view from the UK computing services industry	Francis1993	4	0	9	1993	1	0	3	3	0	10	0.00	0	0	1	0	0	1.0	0.0			0	2	
Effects of process maturity on quality, cycle time, and effort in software product development	Harter2000	2	5	10	2000	1	0	3	3	0	0	0.00	0	1	0	0	0	1.0	3.0			2	1	
Software and systems engineering process development and integration at Oerlikon Aerospace	Laporte1998b	1	0	11	1998	1	1	2	4	32	5	0.00	0	0	1	0	1	0.0	0.0	1992		0	2	
Software process reengineering. Getting to `Level 3' with teamwork	Zahniser1995	1	6	12	1995	1	1	12	12	33	0	0.00	0	0	0	0	0	1.0	1.0			0	2	

Title	Author / Year	#AU1:	#AU2:	#JO:	#YEA:	#SU:	#CO:	#IN1:	#IN2:	#SC:	#DS:	#ROI:	#INV:	#OE:	#E:	#D:	#PA:	#ST:	#EN:	#STY:	#ENY:	#SEI:	#Q:	#CYL:
A software process improvement framework: the theory and practice at Bull Information Systems	Zahran1995	1	0	6	1995	1	10	5	6	34	0	0.00	0	0	0	0	9	0.0	0.0	1991		0	0	
Software process improvement in DSEG 1989-1995	Benno1995	1	0	8	1995	1	1	6	6	35	0	0.00	0	1	0	0	10	0.0	3.0	1989	1992	3	6	
SPI in a very small team: a case with CMM	Batista2000b	7	2	2	2000	1	5	6	6	0	3	0.00	0	1	0	0	0	1.0	1.5			0	5	
The journey to IS organizational maturity: a case study of a CMM level 3 IS organization	Nelson2000b	7	2	6	2000	1	1	3	3	0	10	0.00	0	1	8	0	0	1.0	3.0		1997	0	4	
Process synergy: using ISO 9000 and the CMM in a small development environment	Miller1998	6	0	6	1998	1	1	6	6	0	3	0.00	0	0	9	0	1	1.0	1.5			0	4	
Example of field quality improvement by raising the level of CMM (Capability Maturity Model)	Ishio1998	1	0	6	1998	1	11	6	6	36	0	0.00	0	1	0	0	0	2.0	2.5	1993	1997	0	6	8.00
Software process improvement for knowledge engineering	Dunseath1997	1	0	6	1997	1	10	2	2	37	1	0.00	0	1	2	0	0	1.0	2.5	1996		0	5	
TQM in telecommunication software design	Flam1997	1	0	6	1997	1	14	1	1	38	0	0.00	0	0	0	0	0	0.0	1.5	1996		0	2	
Utilizing the capability maturity model for software to achieve and retain ISO 9001 certification	Sampson1996	1	0	6	1996	1	1	6	6	39	0	0.00	0	1	0	0	1	0.0	2.5		1994	0	3	
Software Process Improvement: A success story	Lipke1992	1	0	5	1992	1	1	4	4	13	7	6.35	2	2	10	0	0	0.0	0.0	1989		2	6	
A CMM Project Takes Off - A Project Milestone Report	Knüvener2003	1	0	6	2003	1	13	13	13	40	4	0.00	0	0	1	0	0	1.0	0.0	2001		0	4	
Implementing the Capability Maturity Model for Software Development	Lowe1996	1	0	8	1996	1	1	6	6	41	0	9.00	2	2	0	0	0	1.0	2.0	1994	1995	3	7	1.00