OULU 604
A 604

ACTA

UNIVERSITATIS OULUENSIS

Pasi Kuvaja

SOFTWARE PROCESS CAPABILITY AND MATURITY DETERMINATION

BOOTSTRAP METHODOLOGY AND ITS EVOLUTION

UNIVERSITY OF OULU GRADUATE SCHOOL;
UNIVERSITY OF OULU,
FACULTY OF SCIENCE,
DEPARTMENT OF INFORMATION PROCESSING SCIENCE

ACTA UNIVERSITATIS OULUENSIS
A Scientiae Rerum Naturalium 604

PASI KUVAJA

SOFTWARE PROCESS CAPABILITY AND MATURITY DETERMINATION

BOOTSTRAP methodology and its evolution

Academic dissertation to be presented with the assent of the Doctoral Training Committee of Technology and Natural Sciences of the University of Oulu for public defence in Auditorium IT116, Linnanmaa, on 4 December 2012, at 12 noon

UNIVERSITY OF OULU, OULU 604

Acta Univ. Oul. A 604, 604

Supervised by
Professor Markku Oivo
Professor Jouni Similä

Reviewed by
Professor Mark C. Paulk
Professor Jarmo Ahonen

ISBN 978-952-62-0029-3 (Paperback)
ISBN 978-952-62-0030-9 (PDF)

ISSN 0355-3191 (Printed)
ISSN 1796-220X (Online)

Cover Design
Raimo Ahonen

JUVENES PRINT
TAMPERE 604

Kuvaja, Pasi, Software process capability and maturity determination. BOOTSTRAP methodology and its evolution

University of Oulu Graduate School; University of Oulu, Faculty of Science, Department of Information Processing Science, P.O. Box 3000, FI-90014 University of Oulu, Finland

Acta Univ. Oul. A 604, 604
Oulu, Finland

Abstract

Software process assessment and improvement came under the spotlight in the discussion of software engineering when the Software Engineering Institute published the maturity model for software process capability determination in 1987. Since then, several new approaches and standards have been developed. This thesis introduces a European software process assessment and improvement methodology called BOOTSTRAP, which was initially developed in an ESPRIT project starting from lean and kaizen philosophy. The focus is on the evolution of methodology and how it was developed, using an experimental research approach. The work covers also enhancements to the methodology investigated in the SPICE, PROFES and TAPISTRY projects. The enhancements expand the original methodology into new specific application areas, keep it compliant with new quality standards and certification, improve the efficiency of the assessment method, enhance the focus from process to product and strengthen improvement monitoring and support. To address these areas, the new BOOTSTRAP methodology releases offer tailored and enhanced assessment reference models and enhanced assessment and improvement methods. The new features also facilitate more frequent and even continuous assessments with software measurement-based indicators.

The thesis explains the origin and features of BOOTSTRAP software process assessment and improvement methodology and how it was developed for professional use. The discussion starts with the evolution of the methodology. Then the new trends and demands are introduced and new features of the BOOTSTRAP methodology described. The conclusion discusses how the methodology developed to be able successfully to support professional software process assessment, to align it with the evolution of software engineering, to adopt the features and requirements of the underlying standards in order to conform to the requirements set by ISO 15504 standard and to become validated in practice.

Keywords: assessment, assessment methodology, assessment model, BOOTSTRAP, capability, capability level, CMM, CMMI, Drive-SPI, ESPRIT, ESSI, EUREKA, improvement, ISO 15504, ISO 9001, kaizen, lean, MONICA, MOOSE, process model, PROFES, software process, SPAM, SPICE, SPIRE, TAPISTRY, technology support

Kuvaja, Pasi, Ohjelmistoprosessin kyvykkyyden ja kypsyyden arviointi. BOOTSTRAP menetelmä ja sen kehittyminen
Oulun yliopiston tutkijakoulu; Oulun yliopisto, Luonnontieteellinen tiedekunta, Tietojenkäsittelytieteiden laitos, PL 3000, 90014 Oulun yliopisto
Acta Univ. Oul. A 604, 604
Oulu

Tiivistelmä

Ohjelmistoprosessin arvioinnista ja parantamisesta tuli ohjelmistotekniikan keskeinen kiinnostuksen kohde kun Carnegie-Mellon yliopiston ohjelmistotekniikan instituutti SEI julkaisi kypsyysmallinsa ohjelmistoprosessin kyvykkyyden arviointiin vuonna 1987. Siitä lähtien maailmalla on syntynyt lukuisa määrä uusia malleja ja standardeja tälle alueelle. Tässä väitöskirjassa esitellään eurooppalainen ohjelmistoprosessin arviointi- ja parantamismenetelmä BOOTSTRAP, joka kehitettiin alun perin Euroopan unionin ESPRIT tutkimusohjelman rahoittamassa projektissa lähtien japanilaisesta ohut-ajattelusta (Lean) ja sen jatkuvan parantamisen periaatteesta (Kaizen). Esitys keskittyy menetelmän kehittymiseen ja siihen miten menetelmä käytännössä kehitettiin käyttäen kokeellista tutkimustapaa teollisessa ympäristössä. Työ kattaa myös alkuperäiseen menetelmään tehdyt laajennukset, jotka syntyivät yhteistyössä SPICE, PROFES ja TAPISTRY projekteissa tehdyn tutkimuksen tuloksena. Tehdyt laajennukset mahdollistavat menetelmän käytön uusilla sovellusalueilla, takaavat menetelmän yhteensopivuuden alan laatu- ja sertifiointistandardien kanssa, parantavat menetelmän tehokkuutta, laajentavat menetelmän käyttöaluetta prosessin arvioinnista sisältämään myös tuotteen kehittämisen arvioinnin ja vahvistavat parantamisen seurantaa ja tukemista. Toteuttaakseen näiden uusien ominaisuuksien vaatimukset uudet BOOTSTRAP menetelmän julkistukset tarjoavat räätälöityjä ja laajennettuja mallikuvauksia arviointien tekemiseksi sekä entistä täydellisempiä lähestymistapoja arviointien suorittamiselle ja parantamiselle. Menetelmän uudet ominaisuudet mahdollistavat myös usein toistuvien arviointien suorittamisen ja jopa jatkuvan arvioinnin ohjelmisto-mittauksia hyödyntäen.

Väitöskirjassa kuvataan yksityiskohtaisesti BOOTSTRAP menetelmän lähtö-kohdat ja ominaisuudet ja se kuinka menetelmä onnistuttiin kehittämään ammattimaiseen ohjelmistoprosessin arviointiin ja parantamiseen sopivaksi. Ensin kuvataan menetelmän kehittyminen ja sitten edetään alan uusien kehitystrendien ja vaatimusten esittelyyn siihen kuinka BOOTSTRAP menetelmä uudet ominaisuudet vastaavat näihin vaatimuksiin. Yhteenvedossa osoitetaan kuinka kehittämisessä onnistuttiin saamaan aikaan uusi menetelmä, joka sopii ammattimaiseen ohjelmistoprosessin arviointiin, vastaa kaikilta osin alan kehittymisen vaatimuksia, sisältää alan standardien vaatimukset täyttävät käytännössä koestetut ominaisuudet, jotka takaavat menetelmän vastaavuuden ISO 15504 standardin vaatimuksiin.

Asiasanat: arviointimalli, arviointimenetelmä, BOOTSTRAP, CMM, CMMI, Drive-SPI, ESPRIT, ESSI, EUREKA, ISO 15504, ISO 9001, jatkuva parantaminen, kyvykkyys, kyvykkyystaso, MONICA, MOOSE, ohjelmistoprosessi, ohjelmistoprosessin arviointi, ohut-tuotanto, PROFES, prosessimalli, prosessin parantaminen, SPAM, SPICE, SPIRE, TAPISTRY, teknologiatuki

To my family

Acknowledgements

Writing this thesis took me seven years, distributed over thousands of random fifteen-minute writing periods, except for the last “scrum”, which filled my summer holidays in the cool, rainy summer of 2012. During that time I was able really to enjoy the privilege of focusing full-time on writing and thinking, without the stress of preparing, managing and reporting on research projects. For that privilege I have to thank my colleagues Dr. Kari Liukkunen and Dr. Jouni Markkula, who helped me clear my calendar for the writing. Additionally, my supervisors, Professor Markku Oivo and Professor Jouni Similä, injected me full of motivation to engage in this final summer of writing, and Professor Harri Haapasalo encouraged and supported me greatly during this time. I am deeply grateful to all of them for giving me the opportunity of my life to complete the thesis.

The research reported in this thesis was conducted in international research projects and institutes during the years 1990 to 2004. The research started in the BOOTSTRAP project, which was part of the Third Research Framework Programme funded by the European Commission and, for the Finnish participants, by the Finnish Funding Agency for Technology and Innovation (TEKES), as Finland was not at that time a member of the European Union. I would like to thank Professor Volkmar Haase and Dr. Richard Messnarz from the Technical University of Graz in Austria for inviting the Finnish consortium to participate in the project and Professor Günter Koch for leading the project successfully. Professor Jouni Similä and Professor Samuli Saukkonen supported the idea and made it possible for Finland to participate.

The research continued in the BOOTSTRAP Institute, which was established as a spin-off from the project. The Department of Information Processing Science joined as an academic member of the Institute. I am very grateful to the directors of the Institute, Professor Roberto Galimberti, Mr. Scott Hansen and Mr. Peter Bölter, who offered me an excellent opportunity to participate in the worldwide “bootstrapping research and development experience” at the Institute internally and in cooperation with the international SPICE project, which covered all continents. I am grateful too to the project manager, Mr. Alec Dorling, for allowing me to serve as a work package co-leader for the development of the process improvement guide.

The research continued within the Institute in a European research project called TAPISTRY, which was part of the ESPITI programme of the ESSI initiative, funded by the European Union Fourth Research Framework Programme. In parallel to TAPISTRY, the research was also conducted in the PROFES project, which was accepted as one of 23 candidates for the “the basket of process improvement” in the

last call of the Fourth Research Framework Programme. I would like to thank project manager, Mr. Frank van Latum, for his cooperation and for educating me about the interior workings of the European embedded systems industry.

In the course of this research, I have cooperated, discussed, argued and shared understanding with more than 200 members of the projects, all of whom were knowledgeable, leading edge experts and researchers. There being so many, it is not possible for me to name the individuals here but I am deeply thankful to all of them for giving me such a superb opportunity to learn about an international way of life. I thank also the European Union and TEKES, especially Mr. Matti Sihto, for organising funding for the research. Additionally, I would like to thank the Tauno Tönning Foundation for awarding me a grant to assist in finalising the thesis.

Serious research is never the work of just one person, although the results have too often been reported in one person's name. This thesis too is based on research results that were created by a great number of participants acting together. The results were reported in six publications by the key researchers and the author of this thesis. I thank all co-authors for their contributions.

I thank also all members of the Department of Information Science and especially Professor Pentti Kerola for offering me basic education on software engineering and an excellent environment in which to grow as an internationally recognised researcher. I am indebted to all staff of the department for their marvellous collaboration and colleagueship. During the work in the department, I had the opportunity to establish a dynamic research group called M-group with Professors Markku Oivo and Jouni Similä. M-group is international and deeply networked and cooperates with the industry in Europe and all over the world. I thank all M-group members for their efficient cooperation, support and inspiring team work.

I respectfully thank the pre-examiners of this thesis, Professor Mark C. Paulk from Carnegie-Mellon University and Professor Jarmo Ahonen from the University of Eastern Finland, for their thorough and encouraging reviews of this work. Finally, I thank my beloved wife, Kaisa, and daughter Paula, and my son in law Tuomas who have whole-heartedly supported me throughout my journey.

You are not important but life is!

List of abbreviations

CMM	Capability Maturity Model
CMMI	Capability Maturity Model Integrated
DoD	Department of Defence
Drive-SPI	Risk Driven Software Process Improvement
EER	Explorative empirical research
ESA	European Space Agency
ESPRIT	European Information Technologies Programme
IDEAL	Software improvement model by SEI
ISO	International Standardisation Organisation
ISO 15504	ISO standard number 15504
ISO 9000	ISO standard number 9000
ISO 9000-3	ISO standard number 9000 part 3
ISO 9001	ISO standard number 9001
PDCA	Plan–Do–Check–Act
PROFES	PROduct Focused software procESs improvement
SCAMPI	Software Capability Appraisal Method for Process Improvement
SEI	Software Engineering Institute
SEL	Software Engineering Laboratory
SPAM	Software Portability Assessment Method
SPI	Software process improvement
SPICE	Software Process Improvement and Capability dEtermination
SPIRE	Self-assessment based software process improvement
SPU	Software producing unit
TAPISTRY	A software process improvement approach tailored for small enterprises
TEKES	The Finnish Funding Agency for Technology and Innovation

List of original publications

I Kuvaja P & Bicego A (1994) BOOTSTRAP – A European assessment methodology. Software Quality Journal 3(3): 117–127.

II Similä J, Kuvaja P & Krzanik L (1995) Bootstrap: a software process assessment and improvement methodology. International Journal of Software and Knowledge Engineering 5(4): 559–584.

III Bicego A & Kuvaja P (1996) Software process maturity and certification. Journal of Systems Architecture 42(8): 611–620.

IV Kuvaja P (1999) BOOTSTRAP 3.0 – A SPICE conformant software process assessment methodology. Software Quality Journal 8(1): 7–19.

V Kuvaja P, Palo J & Bicego A (1999) TAPISTRY – A Software process improvement approach tailored for small enterprises. Software Quality Journal 8(2): 149–156.

VI Kuvaja P, Maansaari J, Seppänen V & Taramaa J (1999) Specific requirements for assessing embedded product development. In: Oivo M & Kuvaja P (eds) Proceedings of International Conference on Product Focused Software Process Improvement – Profes '99. VTT Symposium Series 195: 68–85.

Table of contents

1 Introduction

This thesis arose out of work carried out originally in the BOOTSTRAP project[1] (Kuvaja & Bicego 1993; Koch 1993) and continued in the SPICE[2] (Dorling 1993) and TAPISTRY[3] (Kuvaja *et al.* 1999b) projects on the behalf of the BOOTSTRAP Institute (Kuvaja *et al.* 1994) and PROFES[4] (Oivo *et al.* 1999; Bicego *et al.* 1997) project, along with the wider European industry. The aim of the projects was to develop an assessment methodology for professional use, specifically in the European industry. The thesis will elaborate the development results, starting from the methodology prototype, proceeding to the professional product and ending with product differentiation for special purposes, and show their interchange with and impacts on the evolution of the external knowledge base and use environment. The research methodology applied was constructive (Järvinen 1999) and empirical (Wohlin *et al.* 2003) in nature and applied ideas of design science (March & Smith 1995), Deming's (Deming 1986) Plan–Do–Check–Act (PDCA) cycle and Basili's (Basili 1992) Goal–Question–Metrics (GQM) paradigm. The thesis comprises the published contributions of the author in the above-mentioned projects.

1.1 Background and overview

The software process assessment and improvement movement in software engineering started in the USA in the 1980s, when the Department of Defence (DoD) became concerned about software problems within their military systems. In order to improve the situation, the DoD launched a bid to American research institutes to improve their contractors' software quality. Two main approaches that were recognised were Vic Basili's measurement-based software process improvement (Basili & Weiss 1984) and Humphrey´s approach for software

[1] An ESPRIT Project No. 5441, BOOTSTRAP, funded by the European Commission during 1990–1992.

[2] Software Process Improvement and Capability dEtermination – SPICE, an international project set up by ISO JTC1, Technical Committee 7 (Software engineering) Working Group 10 (Software process assessment) to develop initial working draft material for the forthcoming standard ISO 15504 "Information technology – Software process assessment".

[3] ESSI Esprit Project (No. 24238), called "TAPISTRY", Tailored Application of Software Process Improvement Techniques for Small Enterprises, funded by the European Commission during 1996–1997.

[4] ESPRIT Project No. 23239, PROFES (PROduct Focused improvement of Embedded Software processes), funded by the European Commission during 1997–1999.

(development) process capability evaluation (Humphrey 1989). Humphrey's group at the Software Engineering Institute (SEI) at Carnegie-Mellon University in Pittsburgh published "A Method for Assessing the Software Engineering Capability of Contractors" in 1987 (Humphrey & Sweet 1987). This was the starting point for a worldwide phenomenon called "software process assessment".

In Europe, the American experiences inspired an Esprit project called BOOTSTRAP (from 1989 to 1992) to develop a European process assessment methodology based on Humphrey's model and the ISO 9000 standards, which were becoming the main references also for the software industry. Further, the software lifecycle standard developed by the European Space Agency (ESA) (ESA 1991 February 1991) was used as one of the initial references. The ISO 9000 series formed a family of standards (ISO 9000:1987 1987) and guidelines that specify the minimal requirements for a quality system to underpin the relationship between the purchaser and the supplier within a contractual agreement. Within these norms, ISO 9001 (ISO 9001:1987 1987) specifies the requirements for the quality system of an organisation, covering the entire product lifecycle, including design, development, production, installation and servicing. The standard applies to a large group of products, including among others hardware and software. Additionally, a guide dedicated to software (named ISO 9000-3) was provided to help in applying the ISO 9001 requirements to a software organisation (ISO 9000-3:1991 1991).

Development in software measurement also continued, perhaps the best-known effort being that of Basili's group, which formed the Software Engineering Laboratory (SEL) together with the National Aeronautics and Space Administration's Goddard Space Flight Center (Basili & Green 1994). Similarly, where software development was seen as processes, software process modelling became one possible method of improving the processes (Kellner 1989; Basili 1992; Rombach & Verlage 1993).

The three main types of approach that constitute the main avenues of a phenomenon called software process improvement (SPI), which has its origins in total quality management (TMQ) (Juran 1986a; Juran 1986b; Deming 1981), are CMM-based, standard-based, and measurement-based. The CMM-based approaches apply software process assessment for diagnosing the needs for improvement. The standard-based approaches focus to certify the conformity towards the requirements of the applied standard and indicate the topics for improvement based on the found deficiencies. The measurement-based approaches derive from measurement theory (Fenton 1991: 17–19), and use a set

of metrics for monitoring process performance. No matter which of these three improvement approaches is applied, SPI can be seen in general as the discipline of characterising, defining, measuring and improving software management and engineering processes, leading simultaneously to successful software engineering management, higher product quality, greater product innovation, faster cycle times and/or lower development costs. The motivation behind the SPI comes from the needs and business goals of an organisation, which are often centred on achieving enhanced customer satisfaction and greater competitiveness in cases where the software offer is greater than the market demand[5]. For software producing organisations (SPU), the key management concerns become drivers that initiate software process improvement throughout the organisation, with goals of higher software quality, lower development and maintenance costs, shorter time to market or increased predictability and controllability of software work products and processes.

Nearly all assessment-based improvement approaches include or are based on Watts Humphrey's six-step improvement cycle (Humphrey 1989). Of these, the best-known is the IDEAL model (McFeeley 1996), which describes a series of steps for making improvements to software processes in the CMM assessment framework. Together with CMM, it has become the de facto standard model in CMM-based software process improvement. The original model was developed by SEI (Humphrey 1993) and successfully applied to a number of practical cases in industry (Humphrey *et al.* 1991). The latest development of these models has resulted in the Capability Maturity Model Integration (CMMI) and the Software Capability Appraisal Method for Process Improvement (SCAMPI). Additionally, so-called "SPICE" assessment approaches have been developed both internally by individual companies and by consortia such as Auto-Spice. They meet the requirements of the ISO 15504 suite of standards and guidelines. BOOTSTRAP methodology forms one step in this development path and builds on the initiatives and traditions developed earlier (Koch 1993) and was fed directly into the development of ISO 15504 standard as one of the background models (Paulk & Konrad 1994b), as described above.

[5] The other two cases are the situation where the offer is equal to market demand and where the offer is less than market demand. In the former case, the quality attribute is called "fitness for use" and in the latter "conformance to specifications or standards" (Juran 1992: 9–11).

1.2 Scope of the research

The aim of this thesis is to illuminate the issues related to defining, developing, maintaining and using software process assessment methodology in professional software process improvement, in the light of the evolution of the field. The research reported here has been a part of that evolution and the result is one of the recognised relevant methodologies, called BOOTSTRAP.

Software process assessment has its origins in the total quality management (TQM) movement, and derives from the basic assumption that the quality of manufactured products is largely determined by the quality of the processes that produce them (Deming 1982). Put at its simplest, "assessment is seen just as a determination of how various parts of a software project (such as people, tasks, tools, (internal) standards, and resources) interact to produce software" (Bollinger & McGowan 1991: 25). More broadly: "the objectives of software process assessment are to understand and improve how an organisation uses its resources to build high quality software" (Bollinger & McGowan 1991: 25). According to Humphrey (Humphrey 1989: 14), the first step in software process improvement is to understand the current status of the process[6]. The objectives of the assessment are to learn how the organisation works, to identify its major problems and to enrol its opinion leaders in the change process (Humphrey 1989). All these elements are quite important during the first stage of improvement in order to make it continuous and involve senior management leadership and support for the change[7].

In general, software process assessment may be used for two different purposes, namely software development *capability determination* and software *process improvement*, as shown in Figure 1. Within the context of process *capability determination*, software process assessment is concerned with analysing the capability achievements of selected processes against a target process capability profile, in order to identify the risks involved in undertaking a project that uses the selected processes or in establishing customer relationships with the organisation performing the processes. The proposed capability may be based on the results of previous process assessments in the target organisation or on the assessment carried out for the purpose of establishing the proposed

[6] The subsequent steps are: develop a vision of the desired process, establish a list of required process improvement actions in order of priority, produce a plan to accomplish the required actions, commit the resources to execute the plan and start over at step 1 (Humphrey 1989: 14).
[7] See the principles of improvement in Humphrey's work (Humphrey 1989: 19).

capability in the potential client organisation (Kuvaja *et al.* 1995a). Capability determination is used mainly in second-party assessment context, but even in this context the target organisation should take the results as indication for internal process improvement. If that will not happen it might lead to dysfunctional behaviour as Austin (Austin 1996) has pointed to happen to an organisation in measurement context.

Within the *process improvement* software process, assessment provides the means of characterising the current practice in an organisational unit[8] in terms of capability. The results of the evaluation are analysed in the light of the organisation's goals and business needs in order to understand the strengths, weaknesses and risks inherent in the processes. This, in turn, leads to the ability to determine whether the processes are effective in achieving their goals and to identify significant causes of poor quality or overruns in time or costs. These provide the drivers for prioritising improvements for the processes (Kuvaja *et al.* 1995a).

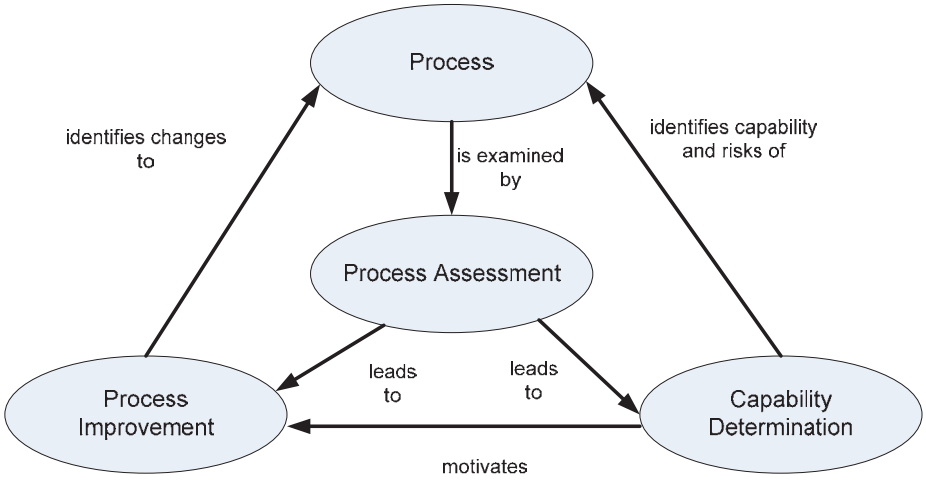

Fig. 1. Software process assessment context (Kuvaja *et al.* 1995a:.3)**.**

Methodologically, software process assessment is understood as *"disciplined examination of the processes used by an organisation against a set of criteria to*

[8] The part of the organisation that is the subject of an assessment. An organisational unit is typically part of a larger organisation, for example a specific project or set of related projects or a part of an organisation responsible for a particular product or product set.

determine the capability of those processes to perform within quality, cost and schedule goals. The aim is to characterise current practice, identifying strengths and weaknesses and the ability of the process to control or avoid significant causes of poor quality, cost and schedule performance" (ISO/IEC-JTC1/SC7/WG7/SG1 1992). As already mentioned, process assessment is performed either during a *process improvement* initiative or as a part of a *capability determination* exercise. As such, it is invoked by and returns results to either the improvement or the capability determination (Dorling 1993). In either case, there should be an input to the process assessment, defining the purpose (why the assessment is being carried out), scope (what processes should be assessed) and what constraints, if any, will apply in the assessment. The assessment input also defines the responsibility for carrying out the assessment and gives definitions for any processes within the scope of the assessment that are variants of the standard reference processes of the approach applied. Figure 2 illustrates the entire context of a software process assessment as defined initially in the SPICE project (Dorling 1993). It shows the assessment input and output, as well as other relevant components and their relationships in a professional assessment.

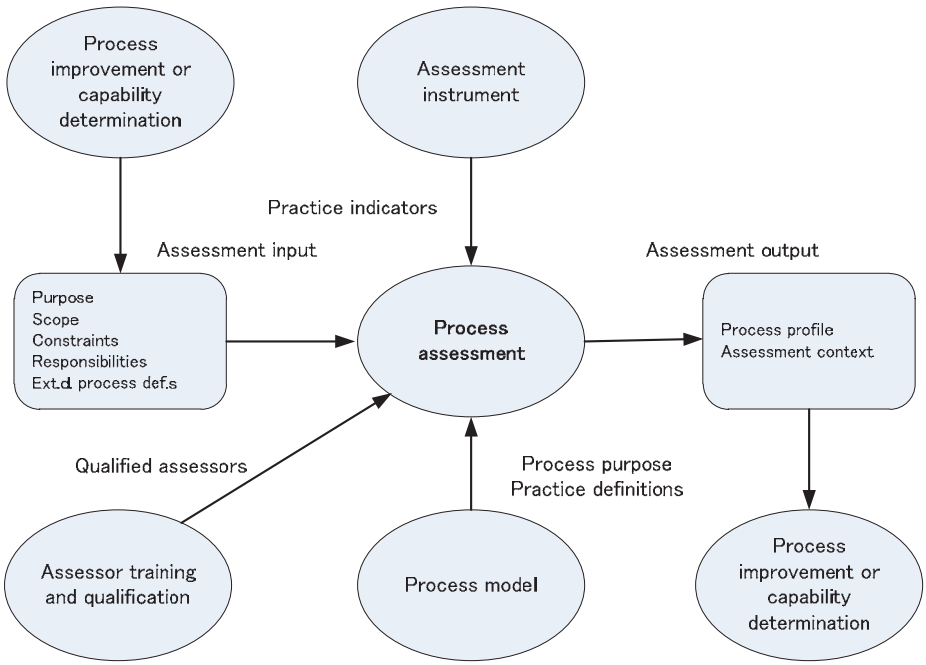

Fig. 2. Software process assessment (Dorling 1993)**.**

In this thesis *the focus is on assessment-based software process improvement,* which is intended to become continuous in the target organisation and whose goals are aligned with the company's own goals (Kuvaja & Bicego 1993). This assumes that the improvement will be based on the results of software process assessment. The idea conforms to Humphrey's (Humphrey 1989) cycle of six steps, as set out in Table 1. The cycle starts with process assessment (to understand the current status), proceeds by a series of improvement activities and ends by beginning a new improvement cycle. In the BOOTSTRAP methodology, identification of the organisation's needs and business goals was added as a preliminary step before the assessment, in order to guide the assessment and improvement planning (Bicego & Kuvaja 1996). This idea was further conveyed to the SPICE project[9], as was also the idea of considering software process improvement as a continuous process.

[9] As well as to ISO/IEC standard 15504 (ISO/IEC 15504-4:2004 2004: 14).

Table 1. Improvement of software development organisation (Humphrey 1989: 14).

Step	Contents
1	Understand the current status of the development process
2	Develop a vision of the desired process
3	Establish a list of required process improvement actions in order of priority
4	Produce a plan to accomplish the required actions
5	Commit resources to execute the plan
6	Start over at step 1

In continuous improvement, an organisation is supposed to move continuously around an improvement cycle, where the improvement is accomplished in a series of steps or specific actions such as introducing new or changed practices into software processes or removing old ones. An important step in the improvement cycle (Kuvaja *et al.* 1995b) is the execution of a software process assessment to

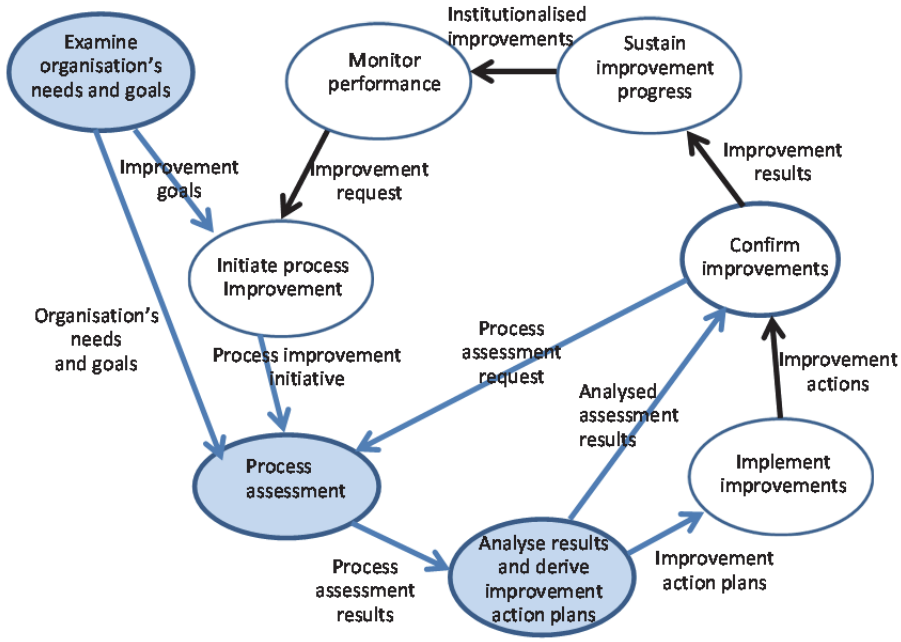

Fig. 3. Scope of research in the continuous software process improvement cycle.

understand the current (initial) state of an organisation's software processes, to use the results of the assessment to formulate and prioritise improvement plans and subsequently to confirm the improvements resulting from the actions performed (see Figure 3).

This work covers the following activities and their inputs and results, as highlighted in blue in Figure 3:

- examine organisation's needs and goals
- perform process assessment, and
- analyse results and derive improvement action plans.

The other activities described in Figure 3 are considered to belong to the organisation's own management, as they require decision making, organisational changes, resource allocations and investments.

1.3 Research objectives and questions

This research grew out of the software problems called the "software crisis" (Brooks Jr 1987), which stimulated several initiatives to tackle the problem, one of which is software process assessment-driven improvement, as described in section 1.2. The best-known initiative was established by the Software Engineering Institute at Carnegie-Mellon University, resulting in an approach called Capability Maturity Model (CMM) (Paulk *et al.* 1991; Paulk *et al.* 1995). The success of CMM initiated a new phenomenon called "software process assessment and improvement" and many follow-up initiatives were launched worldwide, among them a European research and development (R&D) project named BOOTSTRAP[10] (Kuvaja & Koch 1992; Koch 1993).

The initial idea of the BOOTSTRAP project was encouraged by the initial SEI Model (Humphrey & Sweet 1987) at the beginning of BOOTSTRAP project and then the Capability Maturity Model (CMM) (Weber *et al.* 1991; Paulk *et al.* 1991), which was considered to have good potential for improving the competitiveness of the European software and software-intensive industry, despite not fitting perfectly into the context owing to its background (see Tully *et al.*

[10] The BOOTSTRAP project was defined and accepted as one of the pathfinders for the European System & Software Initiative (ESSI) established and funded by the European Commission. ESSI was a software technology transfer programme aimed at catalysing market pull and motivating software producers and users to introduce new methods and tools for software development and maintenance (Koch 1993: 387).

1999: 64). In the European software and software-intensive industry, small and medium-sized companies made up more than 80% of the total at that time, whereas CMM had originally been developed for large software contracting companies of the US Department of Defense. A further European need was to have an assessment methodology that would help software companies to comply with ISO 9000 requirements and sustain them in continuous improvement, rather than just perform software development capability evaluation (SDCE) to an American military standard (DoD-STD-2167A), which was the background reference model of CMM. During the BOOTSTRAP project, criticism was levelled at CMM by Bollinger and McCowan (Bollinger & McGowan 1991), specifically its scoring and rating, evaluation of technology usage and method for dealing with process risks. Commenting on the criticism, Humphrey and Curtis (Humphrey & Curtis 1991) granted, for example, the problems of scoring and agreed on the need to improve the rating, although in their view the structure of the stages was firm and based on statistical analysis and a long history of development in total quality management (Humphrey & Curtis 1991: 48). That discussion gave support to the original motivation for the BOOTSTRAP project.

The mission of the BOOTSTRAP project was to lay the groundwork for technology-transfer programmes in European industry to help software producers and users to improve their software quality through process improvement and introduce new methods and tools for software development and maintenance. The aim was to develop means to analyse the current state of affairs, identify potential for change and motivate and support the changes in software engineering instituted by real companies (Koch 1993: 387). In practice, this required the development of a software process assessment and improvement methodology for the European context. The methodology was meant to be used by all European software industry and software-intensive product development companies including non-defence sectors such as banking, insurance and administration, as well as software product development, embedded systems development and telecoms.

From the European industry context, this meant that the assessment and improvement approach should:

1. fit the assessment-driven software process improvement initiatives implemented also in SMEs
2. support application of the requirements of international quality standards recognised in European industry

3. provide reliable results for improvement initiatives
4. keep the contents comparable with other worldwide accepted software process assessment and improvement approaches and standards (for benchmarking certification purposes, for example), and
5. avoid the criticism directed at existing assessment and improvement approaches.

At the beginning of the BOOTSTRAP project, the research group became familiar with "lean thinking" (Womack *et al.* 1991; Dertouzos *et al.* 1989), which was projected to take over all areas of industrial activity including software production (Koch 1993, based on Cusumano 1991). The BOOTSTRAP hypothesis, developed in accordance with this thinking, was that before any investments are made in technology (T) upgrade (i.e. tools and infrastructure), it is critical to invest in methodology and methods (M) (i.e. how to build solutions) and before that in the working organisation (O) (i.e. how to organise software development and maintenance) (Koch 1993). This basic hypothesis was considered a means of helping software development out of the crisis on a company level and it established the foundation for all research and development efforts of the BOOTSTRAP methodology[11].

More specifically, the BOOTSTRAP methodology development goals derived from the concept of kaizen in lean thinking, namely:

- to recognise existing problems of the organisation
- to support gradual change (improvement) in the organisation
- to involve all personnel in the change
- to get management support and commitment for the change
- to maintain and apply a set of relevant standards and at the same time to seek new ways of upgrading the standards, and
- to provide a method and instruments for problem identification, i.e. for precisely defining where the organisation stands and what gradual changes are to be recommended in the next steps (Koch 1993).

According to Koch (Koch 1993: 388), kaizen brings the following aspects to software process improvement in an organisation:

- recursiveness and regularity

[11] It has also been applied in the subsequent research projects (SPICE, TAPISTRY, Drive-SPI (Maupetit *et al.* 1995) and PROFES).

- feedback looping (retrospectives)
- self-organisation, and
- self-development (autopoiesis) or "bootstrapping".

At the completion of the BOOTSTRAP project, an international initiative called Software Process Assessment and Capability dEteremination (SPICE) (Dorling 1993) was established for the development of an international standard for approaches. All recognised process assessment and improvement methodology developers participated in the project, including BOOTSTRAP (Kuvaja & Bicego 1993; Kuvaja *et al.* 1994), CMM (Humphrey & Sweet 1987; Paulk *et al.* 1991), Healthcheck (Norris *et al.* 1994), Trillium (Coallier *et al.* 1993; Coallier *et al.* 1994), Quantum (Barford *et al.* 1992), STD (Craigmyle & Fletcher 1993), Software Quality Improvement Method (Thomson & Mayhew 1994a; Thomson & Mayhew 1994b) and TickIT (TickIT 1992; Ould 1992). The general tendency was to develop better approaches for software process assessment and improvement, and in SPICE specifically to contribute to the contents of the forthcoming standard and learn from other approaches.

This thesis cannot cover all the ideas, initiatives, needs, requirements, problems or targets mentioned above. The research problems of the thesis are defined by focusing only on those specific research problems that are covered in the original publications of the thesis. Starting from that idea and based on the introduction above, the research problems to be investigated in this thesis can be described as follows.

> *Research question 1 (Q1): What should a software process assessment methodology include, such that it* ***supports professional software process assessment****?*

To answer this question needs clarification of what a professional software process assessment means in this context. Software process assessment is based on the essential assumption that assessment consists of evaluating the capability of an organisation and its processes by comparing performance against some form of structured process model that serves as a yardstick, allowing the development of a ''rating'' of maturity or capability (Rout *et al.* 2007). To do this in a professional way requires that the assessment have a stable form and contents that can be communicated to all stakeholders. The formality of the assessment presupposes that the approach is *disciplined and coherent*; provides *objective, reliable and repeatable evaluations* and fits *different application and business*

areas. For the assessment to be disciplined and coherent, it must be a well-defined process that includes objects, actors and operations. The objects include the process model that is used as a reference in evaluation, objective evidence that is collected from the entities that are the subject of assessment and process attributes that are the objects of evaluation[12]. The process assessment model contains process and capability dimensions. The actors are assessors, sponsors and other stakeholders involved in the assessment or having an interest in its results. The operations include assessment planning, data collection and validation, scoring and rating the process attributes and reporting the assessment. To provide objective, reliable and repeatable evaluations requires clear and transparent scoring scale and rating principles, trained and experienced assessors, supportive performance and capability indicators and documentation of the assessment context. To be suitable for different application and business areas, the process reference model used must include processes that fit the assessment target (Rout *et al.* 2007).

> *Research question 2 (Q2): What should a software process assessment methodology include, such that it* ***supports continuous software process improvement****?*

Continuous software process improvement is one of the cornerstones of professional software process assessment and improvement methodology for use in the European software-intensive industry. Here the concept of continuous improvement comes from the kaizen principle, recognised as one of the starting points of the BOOTSTRAP project. Kaizen is a Japanese philosophy for process improvement; the name comes from the Japanese words "kai" and "zen", which translate roughly as "to break apart and investigate" and "to improve upon the existing situation"[13]. The idea of continuous improvement necessarily implies that it is constant and never-ending (Womack *et al.* 1991) and proceeds iteratively by small steps. Applying kaizen in an organisation is an acknowledgement at the outset that every organisation has problems, which provide opportunities for improvement. Improvements through kaizen have a process focus. Therefore, the companies that apply Kaizen philosophy place emphasis on understanding why a process works, whether it can be modified or replicated somewhere else in the company and how it could be improved. Kaizen means continuous improvement,

[12] Instead of "evaluation", the term "measurement" might also be used, but in the assessment context, "evaluation" fits better, as the measurement is done using an ordinal scale with subjective scoring.
[13] See http://www.michailolidis.gr/pdf/KAIZEN08.pdf, which refers to the Kaizen Institute.

involving everyone in the organisation from top management to workers, all of whom contribute improvements by understanding and communicating how their own work activities fit into the process and may influence change. In essence, support for continuous improvement means that the improvement is guided by the *organisation's own* improvement *goals*, based on its own problems, is carried out through *small steps iteratively* and aims to *involve all personnel*.

> *Research question 3 (Q3): What should a software process assessment methodology include, such that it* ***supports the requirements of relevant standards and their evolution****?*

Standards may be seen as documented agreements containing technical guidelines to ensure that materials, products, processes, representations and services are fit for purpose (Allen & Sriram 2000). One type of standards is process-oriented or prescriptive, where descriptions of the activities and processes are standardised and provide a methodology to perform processes in a consistent and repeatable way (Allen & Sriram 2000). This is the role of the standards considered here, reflected in the BOOTSTRAP project's original statements that it should consider international quality initiatives that specifically apply in Europe. Accordingly, the relevant standards should involve *international standards* (ISO, IEEE), *European standards* (ESA) and *de facto standards* (CMM, CMMI).

> *Research question 4 (Q4): What should a software process assessment methodology include, such that* ***it supports different assessment settings****?*

Software process assessments can differ in many ways while aiming for the same overall target, applying the same principles and producing comparable results. What makes assessment types different from one another is their settings, which include the *purpose, mode, scope and context of the assessment*. The assessment purpose is the statement provided as part of the assessment input[14], defining the reasons for performing the assessment (ISO/IEC 15504-1:2004 2004). In principle it is either process improvement or capability determination. Additionally it may be benchmarking, for example against ISO 9000 requirements. The assessment mode can be self-assessment, second-party assessment or third-party assessment. The assessment scope is a definition of the boundaries of the assessment, encompassing the organisational limits of the

[14] "Information required before a process assessment can commence" (ISO/IEC 15504-1:2004 2004: 2).

assessment, the processes to be included and the context within which the processes operate (for example software development or product development) (ISO/IEC 15504-1:2004 2004). The assessment context is defined by the process context, which is a set of factors that influence the judgement, comprehension and comparability of the assessment results (ISO/IEC 15504-1:2004 2004).

1.4 Research strategy

This study aims to contribute to understanding a phenomenon called software process assessment and improvement, as explained in the previous sections (1.2, 1.3). More specifically, the objective of the research is to contribute to the development of assessment and improvement methodology. The research approach used in the thesis belongs to the category of constructive research, occasioned by research questions (Järvinen 1999) stated in the previous section (1.3). According to Järvinen, "…constructive research is applied research that builds a new artefact and this process is based on existing knowledge and/or new technical, organisational etc. advancements. Then the utility of the new artefact is sooner or later evaluated" (Järvinen 1999: 59). The artefact here is the assessment and improvement methodology called BOOTSTRAP. The methodology development will happen in collaboration between practitioners from industry and researchers in European and international R&D projects and evaluations will be done in numerous practical experiments.

The conduct of the research is to be iterative and happen in experimental cycles. The scientific basis and structure of the activities of the research derive from Deming's cycle (Deming 1986), which proposes one full experimental cycle to include Plan–Do–Check–Act (PDCA) steps in sequence. The PDCA cycle is further supported by the ideas of design science defined by March and Smith (March & Smith 1995) as including build, evaluate, theorise and justify activities, and concepts of artefact and performance as the topic and result of the activities. Additionally, design science defines the concepts of external knowledge base and (use) environment.

Although the Goal–Question–Metrics (GQM) approach (Basili 1992) was originally developed for measurement purposes, its principles can also be applied to this research approach by seeing measurement from a wider perspective and understanding it as evaluation. On that assumption, GQM is considered here to support the cyclic structure when applied to the entire research process. This allows the concepts of Goal–Question–Metrics to be related to the PDCA cycle,

as illustrated in Figure 4. The concepts may then be applied to an entire research project or one research cycle. In the former, the goal is derived from motivations and research problems of the project and in the latter from the goals of the research cycle, which become more precise during each cycle.

A modified conceptual structure based on the assumptions characterised above forms an overall framework for the research strategy of this thesis. The framework is outlined in Figure 4; the elements (which are given in italics at their first appearance) will be described in more detail in the subsequent paragraphs.

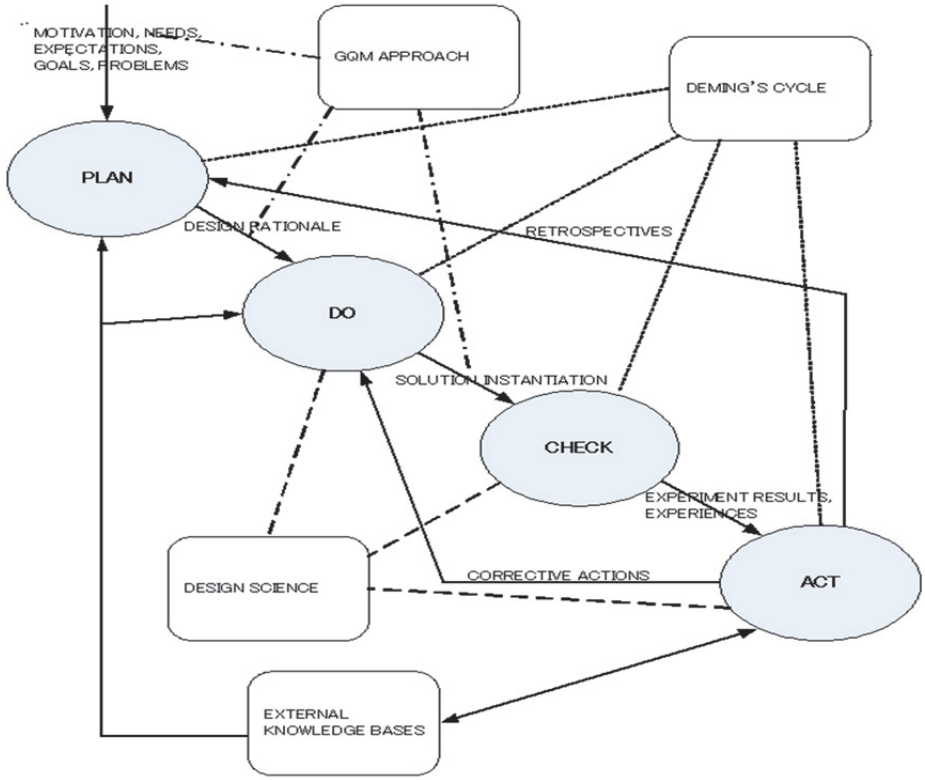

Fig. 4. Research approach.

Plan, the first step in the PDCA cycle, begins with discovering and analysing the motivations, goals, targets, expectations, needs or problems that initiate the cycle.

The analysis is done with the state of the art, state of the practice and state of the experience captured from external sources. This is followed by choosing and deriving a specification[15] for the solution design or concept, which, together with the initiating reasons, forms the *design rationale*[16]. The design rationale defines what to construct or build and why, and is, therefore, in line with the concept of the *question* part of the GQM approach.

In the *Do* step, the requirements set out in the design rationale are used as guidelines to build or construct the *artefact*. Applicable knowledge from external sources is also used in the building process. The artefact can be a construct, model, method, solution, instance, etc. In this research the artefact is an instance of BOOTSTRAP software process assessment and improvement methodology (Kuvaja et al 1994). The applicable knowledge can be descriptions and/or experiences of related approaches or background standards available from external sources or knowledge bases.

The purpose of the *Check* step is to experiment with the solution instance in practice and collect results and feedback from the behaviour of the solution instance and experiences of its use. Collecting the results can be considered as measurement (in a wider perspective), which is in line with the concept of the *measure* stage of the GQM approach. The results constitute lessons learned that will form the input for the next step.

In the *Act* step, the lessons learned are analysed and evaluated, *corrective actions* defined for an immediate construct, the *solution instantiation, evaluation results and experiences* packaged and communicated for the next experimental cycle, and *contributions* prepared to be communicated to academia and external industry, which form the external knowledge base (Hevner & Chatterjee 2010; March & Smith 1995: 259).

The entire PDCA cycle may include both internal and external iterations. Internal iterations insert corrective actions into the experimental cycle and external cycles are initiated with new understanding of the artefact and its context (March & Smith 1995: 259).

During the experimental research cycles, literature analysis, interviews with experts, experts' external reviews, brainstorming, interactive workshops,

[15] An assessment method is in itself a product designed from specifications (April & Coallier 1995b: 175).

[16] Design rationale is the notion that design goes beyond merely accurate descriptions of artefacts, such as specifications, and articulates and represents the reasons and the reasoning processes behind the design and specification of artefacts (Moran & Carrol 1996: 2).

experiments, pilot testing through assessments and evaluations, experience analysis and wider retrospectives are used as actual research methods. The approach is quite similar to that used for example in CMM development (compare (Humphrey & Curtis 1991: 45).

The research strategy described in this section has been applied in the activities that produce the contribution reported in this thesis. The application is described in chapter 3.

1.5 Outline of the thesis

This thesis consists of six sections, outlined in Figure 5, which shows the main contents of and logical interrelationships between the sections. In the first section, the research area, problem and research strategy are defined. The research area comprehends both the background and the scope of the thesis. The research problem is further elaborated into specific research questions, according to subtopics of interest. The research strategy is aligned with the research questions defined and describes the dynamics of the activities to be followed in conducting the research.

In the second section, related research efforts are outlined in order to set the contents of the thesis in their scientific context and to demonstrate the new knowledge the thesis is bringing to light. The section is focused on the approaches and their evolution, which have a direct link to BOOTSTRAP methodology development and its evolution. It gives an account of CMM evolution from its beginnings, other assessment approaches that have been recognised in the area and BOOTSTRAP methodology development. There follows an overview of the international reference standards.

In section three, the developments of BOOTSTRAP methodology are presented in three phases, aligned with the evolution of the methodology versions. In this section, “the design rationale” of the methodology is presented, including its specific features, and put forward for critical observation.

In section four, the findings of the study and their relationships and contribution to the evolution of the external knowledge base are presented. The section starts with a summary of the results and their relative impact in answering the research question stated in section 1.3. The outline refers also to original papers published during this phase of the research. Results of each research phase are analysed individually.

In section five, original publications are introduced separately. The purpose, contents and main contribution of each publication are proposed and related to the research questions. Also, the author's role in conducting the research and reporting the results is clarified.

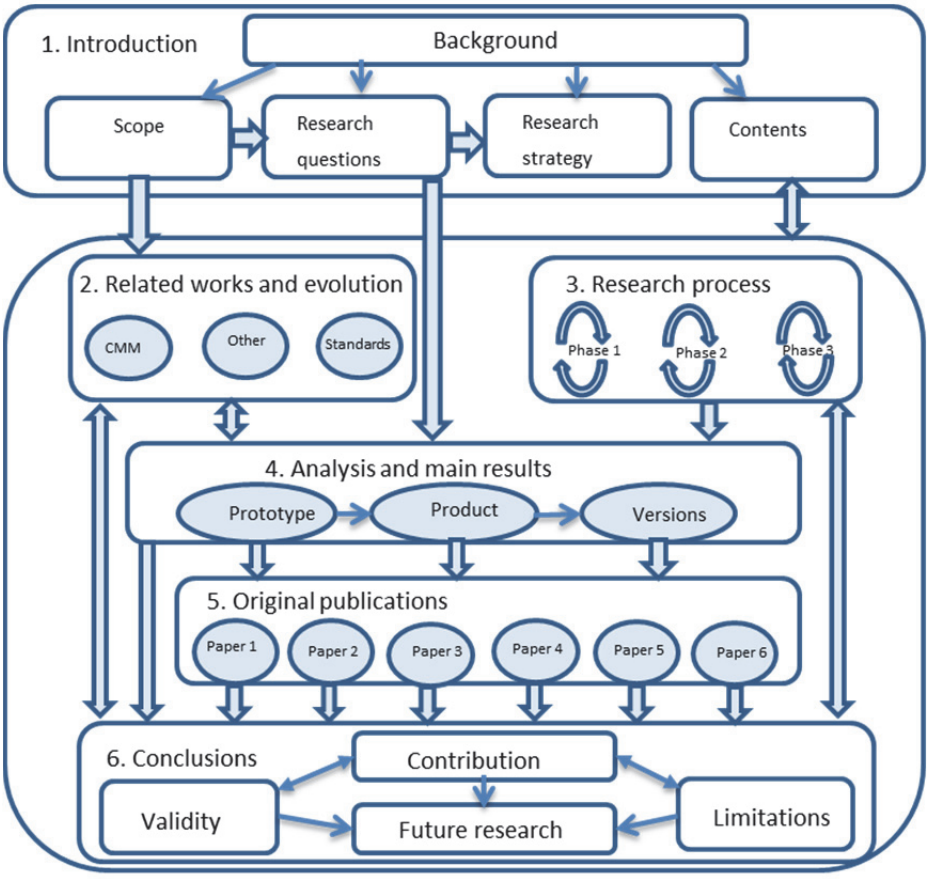

Fig. 5. Contents of the thesis.

In section six, the main contribution of the thesis is presented by answering the research questions one by one. Validation of the research and results is then described using the guidelines defined by Hevner (Hevner & Chatterjee 2010) for the design science research that underlies this thesis. In this section also, the

limitations of the research are identified and the potential for future research outlined.

2 Related works and evolution

The scope of this research is to study and develop software process assessment and improvement approaches, also called software process appraisal (Olson *et al.* 1989; Paulk 2009), as described in section 1.3. In this chapter, all related background approaches and their evolution that influenced or have been combined into the methodology development are discussed. The emphasis is on the approaches that have directly impacted on BOOTSTRAP methodology development or that help in understanding the artefact in its own context. Specifically, we describe those methods, standards and research results that have served as sources of inputs to this research effort, or which are elements of the knowledge base that may potentially receive an additional contribution based on the results of this study.

The chapter contains four main areas of interest. The first section describes Capability Maturity Model (CMM) developments[17], which were one of the triggers and background models for the BOOTSTRAP methodology development. CMM is a de facto reference and reflection model for BOOTSTRAP development and a de facto international standard for software process assessment and improvement. The first versions of the CMM model and Software CMM tackle similar research and practical problems, make quite similar theoretical assumptions and end up with similar technical solutions such as BOOTSTRAP, although they exhibit a number of distinct differences. The second section introduces other assessment and improvement models that have been devised in various parts of the world and discovered during the BOOTSTRAP research. The third section outlines the background standards for reference models of the assessment approaches and guidelines for improvement. The last section is dedicated to the Software Process Improvement and Capability dEtermination (SPICE) initiative, which has packaged the assessment and improvement approaches into the form of requirements that increase the possibilities of benchmarking the results of different assessments.

2.1 Capability Maturity Model developments

The Capability Maturity Model (CMM) resulted from research conducted at the Software Engineering Institute (SEI) at Carnegie-Mellon University in Pittsburgh,

[17] Including CMMI.

Pennsylvania. The SEI was established to support advancement of the software engineering practices adopted by suppliers of the US Department of Defense (DoD) to improve the quality of software-dependent systems. Later on, the results of the work were disseminated to a wider community beyond the military.

2.1.1 Capability Maturity Model

The initial version of the CMM was first published in the shape of a technical report describing the maturity framework (Humphrey & Sweet 1987), general assessment procedure, four questionnaires and algorithms for defining the maturity level and technology support stage (generally called the "SEI model") (Humphrey & Sweet 1987). The original goal of the SEI model was to provide guidelines and procedures for assessing the ability of potential Department of Defense (DoD) contractors to develop software in accordance with DoD standards. The primary objective in developing the structured assessment approach was to provide a standardised method that is documented, publicly available and periodically modified as experience is gained with its use. A further objective was to provide a public process to assist software organisations in identifying areas where they should make improvements in their own capabilities, specifically to be able to secure contracts with the DoD (Humphrey & Sweet 1987).

In complete SEI assessment, a specific questionnaire was used to evaluate the level of experience of the software development personnel, to define the maturity level of the software engineering process, to classify the stage of software technology and to receive an assessment perception of the target organisation[18]. Evaluation of the experience level is based on a questionnaire of nine questions with numeric answers[19] regarding years of experience of the personnel and size of the organisation[20]. The nature of the numeric information collected using these questions is informative and intended to assist assessors in evaluating the relevance of the target organisation in a particular procurement context[21] (Humphrey & Sweet 1987).

[18] Originally "contractor" (Humphrey & Sweet 1987).

[19] The questions are expected to be answered with numbers like "years of experience of software development managers", "percentage of the software development staff who have a bachelor degree or higher in computer science or software engineering", etc. (Humphrey & Sweet 1987: 21).

[20] Or organisational unit.

[21] In general the information can also be used for classification of the target organisations in comparing the results of different assessments when using maturity levels and technology stage (the

The software engineering process maturity level is evaluated using a questionnaire comprising 85 questions with "yes" or "no" answers. The maturity levels are: *Initial*[22], *Repeatable*[23], *Defined*[24], *Managed*[25] and *Optimised*[26][27], *Initial* being the lowest level and *Optimised* the highest. Each maturity level builds on the previous level. The starting point for assessment is that the software engineering process is on the *Initial* level at the beginning and rises when attributes of good software engineering practices are recognised. At the end of the assessment, the real maturity level of the software engineering process of the organisation is defined, based on answers given in the questionnaire[28].

---

subsequent two questionnaire parts of the model) as the means of comparison (Humphrey & Sweet 1987: 5).

[22] At the *Initial* level (level 1), the software engineering environment has ill-defined procedures and controls. The organisation does not consistently apply software engineering management to the process, nor does it use modern tools and technology. Level 1 organisations may have serious cost and schedule problems (Humphrey & Sweet 1987: 5).

[23] At the *Repeatable* level (level 2), the organisation has generally learned to manage costs and schedules, and the process is now repeatable. The organisation uses standard methods and practices for managing software development activities, such as cost estimating, scheduling, requirements changes, code changes and status reviews (Humphrey & Sweet 1987: 5).

[24] At the *Defined* level (level 3), the process is well characterised and reasonably well understood. The organisation defines its process in terms of software engineering standards and methods, and it has made a series of organisational and methodological improvements. These specifically include design and code reviews, training programmes for programmers and review leaders and increased organisational focus on software engineering. A major improvement in this phase is the establishment and staffing of a software engineering process group that focuses on the software engineering process and the adequacy with which it is implemented (Humphrey & Sweet 1987: 5).

[25] At the *Managed* level (level 4), the process is not only understood but also quantified, measured and reasonably well controlled. The organisation typically bases its operating decisions on quantitative process data and conducts extensive analyses of the data gathered during software engineering reviews and tests. Tools are used increasingly to control and manage the design process as well as to support data gathering and analysis. The organisation is learning to project expected errors with reasonable accuracy (Humphrey & Sweet 1987: 6).

[26] At the *Optimised* level (level 5), organisations have not only achieved a high degree of control over their process but also have a major focus on improving and optimising its operation. This includes more sophisticated analyses of the error and cost data gathered during the process as well as the introduction of comprehensive error cause analysis and prevention studies. The data on the process are used iteratively to improve the process and achieve optimum performance (Humphrey & Sweet 1987: 6).

[27] After 1987 the SEI used Optimizing rather than Optimized

[28] The questions have been developed keeping in mind (among other things) that the quality of a software product stems, in large part, from the quality of the process used to create it; the software engineering process is a process that can be managed, measured and progressively improved; the quality of a software process is affected by the technology used to support it; the level of technology used in software engineering should be appropriate to the maturity of the process; and software products developed by contractors for DoD use are acquired under contracts invoking DoD-STD-

The questions cover organisation and resource management and software engineering process and its management. Organisation and resource management deals with functional responsibilities, personnel and other resources and facilities of the target organisation. Software engineering process and its management concern the scope, depth and completeness of the process and how the process is measured, managed and improved. Both of the objects of evaluation are composed into specific topics of interest, which have all been further elaborated into assessment questions. Altogether there are seven interest topics grouped into the maturity levels[29], so that the questions relating to a specific level are considered as a hurdle that must be negotiated in order to pass on to the level. The questions are also classified as standard questions and important questions[30]. The maturity levels are considered to build upon the previous levels and therefore each level has its own questions, which are considered together with the questions of the previous level when evaluating fulfilment of the threshold criteria. The threshold is 80% for all questions and 90% for important questions (Humphrey & Sweet 1987).

The questionnaire includes an additional 16 "yes or no" questions that cover tools and technologies used in the software engineering process and form a method for evaluating the software engineering technology of the organisation. Additionally there are ten follow-up questions for the assessment team to request amplification of responses. The answers to the questions are expected to reflect standard organisational practice. The result of the technology assessment places the target organisation either at stage A (inefficient technology) or stage B (basic technology) (Humphrey & Sweet 1987).

The maturity levels of the SEI model were further elaborated in an article published in *IEEE Software* (Humphrey 1988), and more comprehensive description of the background ideas and main features of the maturity framework and process improvement were published in a book by Humphrey in 1989 (Humphrey 1989), which also gave more details on a framework to define a maturity level for the target organisation. The model attracted both positive and

---

2167/A, Defense System Software Development, as tailored for each contract (Humphrey & Sweet 1987: 5).

[29] In Humphrey and Sweet (Humphrey & Sweet 1987), the levels are called maturity levels and the "overall maturity" of the assessed contractor is called "capability"; " the capability of a contractor to perform software engineering has been divided into three areas", namely organisation and resource management, software engineering process and its management and tools and technology.

[30] The important questions are marked in the questionnaire with an asterisk and are considered to have "greater importance for the indicated maturity level" (Humphrey & Sweet 1987).

negative criticism from Bollinger and McCowan (Bollinger & McGowan 1991), to which Humphrey immediately responded (Humphrey & Curtis 1991).

The assessment results obtained using the SEI model were collected and analysed over four years from 1987 to 1991 and the analysis published in a technical report by Kitson and Masters in 1992 (Kitson & Masters 1992). A summary article also appeared in the *International Conference of Software Engineering* Proceedings in 1993 (Kitson & Masters 1993). The results had been obtained from 59 assessments and 296 projects. Thirteen assessments were conducted as SEI-assisted (Olson *et al.* 1989), encompassing 63 projects, and 46 were self-assessments, encompassing 233 projects.

2.1.2 Capability Maturity Model for Software

Based on the experience with the SEI software maturity framework and the maturity questionnaire, the SEI formalised the concepts and evolved the maturity framework into the Capability Maturity Model for Software (called Software CMM version 1.0), released in August 1991 (Paulk *et al.* 1991). In the model, the maturity levels were further elaborated by introducing concepts of key process areas, key practices and key indicators, which together formed the basis for a new maturity questionnaire to be used in CMM assessments and which were described in detail in a separate report entitled "Key Practices of the Capability Maturity Model" (Weber *et al.* 1991). In the model, software process assessment and software capability evaluation were clearly separated. Software process assessment was meant to be conducted by an organisation and focus on identifying improvement priorities within its own software process (Paulk *et al.* 1991), whereas software capability evaluations were meant to be performed by a third party and focused on identifying contractors who represented the lowest risk for building high-quality software on time and within budget (Paulk *et al.* 1991). The common steps for software process assessment and software capability evaluations were also defined. The maturity questionnaire was published in June 1994 (Zubrow *et al.* 1994), replacing the former questionnaire published in 1987 as the "SEI Model Questionnaire" (Humphrey & Sweet 1987). Therefore, in the assessments performed with Software CMM version 1.0, the older version of the maturity questionnaire was used. Software CMM version 1.0 attained a great success in industry, as 157 SEI assessments had already been conducted by December 1992 and the results provided to the SEI (Kitson & Masters 1993).

Software CMM version 1.1 was published in February 1993 as two technical reports, "Capability Maturity Model for Software, Version 1.1" (Paulk *et al.* 1993) and "Key Practices for the Capability Maturity Model, Version 1.1" (Paulk *et al.* 1993) and published as a book in 1995 (Paulk *et al.* 1995). The new version included few substantive changes, although almost every practice was changed by clarifying its meaning and making the wording of practices more consistent (without substantively changing their content or intent). Additionally, the wording of the goals was changed by omitting subjective expressions (such as "reasonable plans" and other goodness attributes). The purpose was to use the goals in rating satisfaction of the key process areas. According to Paulk (Paulk 2009), the problem of dysfunctional behaviour remained in the model as a result of its targeting the improvement towards maturity level rather than improvement against business objectives. "The Software CMM has now been retired in favour of the CMM Integration (CMMI) model, but has inspired many other standards and frameworks, including the People CMM (Curtis *et al.* 1995), the Systems Engineering CMM (Bate *et al.* 1995) and the Systems Security Engineering CMM (Hefner 1997). It was also was one of the drivers in the development of ISO/IEC 15504 (Process Assessment), notably Part 7 on the assessment of organisational maturity (ISO 2008)" (Paulk 2009: 5).

In fact, the development of Software CMM version 2 had already started during the development of version 1.1, as requests for change that involved radical revisions (for example, adding new key process areas) were deferred to version 2, which was planned for 1997 (Paulk 2009). During version 2 development, SEI was participating actively in the SPICE project and alternatives to include aspects of continuous representation were being investigated (Paulk *et al.* 1995; Paulk 1999). Both the staged and continuous perspectives were recognised to have value and to be conceptually compatible (Paulk 1996). The result of the process was to make the relationships between the two perspectives explicit but leave the staged architecture as the primary representation (Northcutt & Paulk 2010). The relationship was implemented by including institutionalisation from general practices in the continuous model architecture into a maturity level goal for each key process area of the resulting stage model. Additionally, process contents were changed or split and new key process areas of "risk management" and "organisational software asset commonality" were added (Paulk 1997; Paulk 2009). The release of Software CMM version 2 was planned for the end of 1997 but was halted in favour of work on the CMMI in October of that year (Paulk 2009).

2.1.3 Capability Maturity Model Integration

Capability Maturity Model Integration (CMMI) was developed by the CMMI product team, a team of process improvement experts from the government, industry and the SEI, to improve the Software Capability Maturity Model (SW-CMM) released in 1991 (Paulk *et al.* 1991; Weber *et al.* 1991). The SW-CMM was a "roadmap" that described "evolutionary stages" consisting of key practices that guide organisations in improving their software capability. After SW-CMM was released and updated (Paulk *et al.* 1993; Paulk *et al.* 1993), the Enterprise Process Improvement Collaboration (EPIC), a US industry and government collaborative effort, together with SEI, developed and published the Systems Engineering Capability Maturity Model (SE-CMM) (Bate *et al.* 1995) and the International Council on Systems Engineering (INCOSE) developed and published the Systems Engineering Capability Assessment Model (SECAM) (Sheard 1997). Several systems engineering maturity models supported systems engineering process improvement (Sheard 1997; Paulk 2004). The EPIC SE-CMM and the INCOSE SECAM were alternative models and were amalgamated with the Electronic Industries Alliance (EIA) to create EIA/IS-731 (Sheard & Lake 1998). Additional CMMs were also developed, including: the Software Acquisition CMM, the People CMM and the Integrated Product Development CMM. In the creation of the CMMI systems engineering from EIA/IS-731 (Systems Engineering Capability Model), software engineering from SW-CMM (version 2C) and integrated process and product development from the Integrated Process and Product Development IPPD-CMM (version 0.98) were integrated into a single model that incorporated both the staged (CMMI 2002b) and continuous (CMMI 2002a) representations.

CMMI continued to evolve; version 1.2, an upgrade to version 1.1, was published in 2006. The new overall feature was the concept of CMMI "constellations". A constellation can produce one or more related CMMI models and related appraisal and training materials. Version 1.2 contains three constellations. CMMI for Development was the first of these constellations; it is a reference model that covers the development and maintenance activities applied to both products and services (Chrissis *et al.* 2007). The second constellation was CMMI for Acquisition (CMMI 2007), which represents best practices for improving relationships with suppliers through improvements to internal processes. It can be used to increase control of projects, better manage global sourcing of products and services and more successfully acquire solutions that

meet organisational needs. The third constellation of the version 1.2 was CMMI for Services (CMMI 2009), which contained descriptions of best practices for service provider organisations.

Currently CMMI is already published in version 1.3. In this version, the practices for maturity levels 4 and 5 have been revisited, the representation of staged and continuous models made better aligned and modern engineering practices like "agile" included (Phillips 2010). The version follows the line of the previous version (1.2) and includes three constellations, namely CMMI for Development (v.1.3) (Chrissis *et al.* 2011), CMM for Acquisition (v.1.3) (Gallagher *et al.* 2011) and CMM for Services (v.1.3) (Forrester *et al.* 2011). The evolution of CMMI will certainly continue and new versions will appear. One obvious line to be updated will be the inclusion of agile principles (Glazer *et al.* 2008) in all constellations; at present, agile development is included in CMMI Development only.

2.2 Other assessment and improvement models

At the beginning of the BOOTSTRAP project and even before (Kuvaja 1992), many other research and development efforts were directed towards assessment and improvement approaches. Most of them were inspired by the SEI model, and experimented or followed the ideas of the model in their own developments. The most recognised approaches are outlined here. All of them also participated in the SPICE project (Dorling 1993). The general intention was to develop better approaches for software process assessment and improvement, and in the SPICE project specifically to aim for standardisation of the contents and learn from the other approaches.

2.2.1 Healthcheck

Healthcheck was developed within British Telecom (BT) in 1989. The purpose was to find a method of assessing software process against current world best practice to enable direct improvement initiatives for process, product and people working in software development (Mackie & Rigby 1993). The objective was to improve the quality of the software and hence to understand where to allocate investment in changes. The idea was to apply an improvement cycle that included steps identified as assess, analyse, modify and execute. Before the development of Healthcheck, extensive trials were conducted in BT using the SEI model

(Humphrey & Sweet 1987). At first the model was found to lack certain key practices in requirements for capture and analysis and for verification, validation and testing, which were deemed important in the telecoms business. Accordingly, checklists were added to the basic model. The additions were developed internally by experts in various software engineering disciplines. However, the outcome was felt to be a failure for many reasons (Mackie & Rigby 1993) and the CMM was abandoned. At end of the experiment, the additions made to the CMM were substantially augmented and reviewed and formed the nucleus of the Healthcheck assessment method (Mackie & Rigby 1993).

Healthcheck assessments consist of a series of one-to-one interviews with people working in a particular unit or project. Each interview lasts one to two hours. All interviews are based on the Healthcheck questionnaire, which consists of nine key process areas, each containing from ten to 16 questions. The key process areas are project management, requirements capture and analysis, metrication, design, configuration management, verification, validation and testing, performance engineering, maintenance and release and implementation. The key process area fulfilments are scored on a percentage scale. A 100% score for a key process area indicates that a unit is, in BT's experts' view, working according to current world best practice. Healthcheck also allows the collection of other data, which may include a limited collection of "people" and "product" related data. The results are presented as radar diagrams using a percentage scale (Mackie & Rigby 1993; Norris *et al.* 1994).

According to Mackie (Mackie & Rigby 1993: 270), "The primary aim of the Healthcheck process is to assess groups of engineers in order to assist in process improvement. It has also been applied to big software programs where a rapid assessment of the state of software engineering across the program is required. This is most important where the development is taking place on geographically different sites." Healthcheck has also been used to assess potential software suppliers. Between 1990 and 1993, approximately 1,000 software engineers were interviewed within Healthcheck assessment in British Telecom (Mackie & Rigby 1993).

2.2.2 Software Technology Diagnostic

Software Technology Diagnostic (STD) was developed in 1989 by the Scottish Development Agency (SDA) as part of their software technology programme. The aim of the programme was to support small and medium-sized organisations

(4–50 staff members) in their efforts to stay competitive (Tully *et al.* 1999). STD is a process assessment approach designed as an integral part of a performance improvement programme, which can be conducted speedily and without major disruption to the group being assessed (Craigmyle & Fletcher 1993). Craigmyle and Fletcher characterised the model as follows: "A key feature of the STD is the gathering of business data so that the assessment results can be judged in the context of the organisation's own business needs" (Craigmyle & Fletcher 1993: 257).

STD follows the classical three-step assessment, including preparation, diagnosis and a feedback clinic, which they call the "improvement programme establishment" (Craigmyle & Fletcher 1993). The preparation step includes two tasks, which are to commit senior management to the improvement and get them involved in the assessment process, and to define the scope of the assessment and improvement (personnel to be interviewed and projects to be assessed). In the diagnosis step, assessors interview the personnel (one assessor per interview) and collect the data using STD evaluation forms based on the reference architecture of STD processes. The assessor then evaluates the data systematically, provides a quantitative percentage rating of the capability and effectiveness of the processes and a qualitative assessment of the priority areas for improvement and prepares an outline action plan. In the feedback clinic, the results of the assessment are communicated to the senior management (Craigmyle & Fletcher 1993).

The STD approach (version 2.5) includes a reference framework of 20 processes allocated to three capability levels (2 to 4) according to CMM (Tully *et al.* 1999). Most of the processes span levels 2 and 3, and only three processes reach level 4. Processes on level 2 have six common practices and between two and 12 key practices, varying between the processes. Processes on level 3 have seven common practices and from one to five key practices, accordingly. Processes reaching level 4 have seven common practices and only one or two key practices. Assessors score the existence of practices on a binary scale ("yes" or "no") and give a percentage rating to management practices and use of technology (Compita 1993b; Compita 1993a).

2.2.3 TickIT

TickIT is a certification scheme, initiated by the UK Department of Trade and Industry (DTI), to regulate third-party certification of software quality management systems under ISO 9001/9000-3, and in that way to improve

professional practice and market confidence in software quality management systems (QMS) audit. TickIT addresses those aspects of an organisation's QMS that are concerned with the specification, design, development, installation and support of software. TickIT is seen as a stepping stone in the development of a QMS in the context of TQM, based on the use of definitive standards. TickIT applies the ISO 9000-3 guidelines to ISO 9001, which details the quality requirements for systems in general. It is advised to use also the ISO 9002 part of the ISO 9000 series of international standards in TickIT certification, particularly in relation to production, installation and servicing activities. TickIT has been one of the approaches influencing the ISO working group for the development of international standards for software process assessment, improvement and capability determination (SPICE). The scheme applies mainly to the UK but has been adopted in other countries as well (Thomson & Mayhew 1994b; Tully *et al.* 1999; TickIT 1995).

2.2.4 Trillium

In 1991, the first version of the Trillium model was elaborated by Bell Canada, Nortel and Research Bell Northern. This model and its application method have been improved based on experience and feedback from suppliers. From 1992 to 1993, version 2 with variations was used extensively and data about assessment were collected from assessment teams and management. In January 1995, version 3.0 was issued to Bell Canada suppliers and placed in the public domain. Trillium has been used worldwide within the above-mentioned group of companies, both for self-assessment and for the assessment of existing and prospective suppliers (Tully *et al.* 1999; April & Coallier 1995b).

Trillium was designed initially to be applied to embedded software systems such as telecommunications systems. It incorporates requirements from the ISO 9000 series, the CMM for Software and the Malcolm Baldrige criteria, with software quality standards from the IEEE and internal Bellcore standards. The telecoms standards bring a telecommunications orientation to the model, which therefore cannot necessarily be adopted as-is, because in some cases, goals of the frameworks are used rather than their detailed requirements and because the model includes process information that is unique to the telecommunications field. However, much of the model can be applied to other segments of the software industry such as Management Information Systems (MIS). Furthermore, a significant percentage of the practices described in the model can be applied

directly to hardware development. The Trillium model covers all aspects of the software development lifecycle, most system and product development and support activities and a significant number of related marketing activities. Altogether, the model serves as proof that the requirements of several of the popular frameworks can be combined, and it provides a template for additional efforts in this area (April & Coallier 1995b; Sheard 1997).

Although the Trillium model is based on the SEI model (CMM version 1.1), the architecture and the concepts of capability and maturity differ from it significantly. The most notable differences are the roadmap-based concept (instead of key process areas), product perspective (instead of software), wider scope of capability (all functions that contribute to the customer's perception of the product, including engineering, marketing, customer support and quality assurance) and strong customer focus (capability from the customer and development perspective). The Trillium model is based on the concept of a roadmap, a set of related practices that focus on an organisational area or need or a specific element within the product development process. Each roadmap represents a significant capability for a software development organisation. Within a given roadmap, the level of the practices is based on their respective degree of maturity. The most fundamental practices are at a lower level, whereas the most advanced ones are at the higher level. The roadmaps are organised into capability areas, each representing a significant capability for a software development organisation. The model thus contains capability areas, roadmaps and practices, where each capability area incorporates one or more roadmaps, and each roadmap comprises one or more practices that span several Trillium levels. Trillium version 3.0 had eight capability areas, 28 roadmaps and 508 practices. More detail on Trillium is given in the next section. In addition to process maturity, it offers the ability to assess the appropriateness of the technology used in the process (Tully *et al.* 1999; April & Coallier 1995b).

2.3 Standards and guidelines

Process assessment in companies relies on comparing their own practices with the best and successful organisations. In practice this means benchmarking their way of working in software development against professional, national or international standards, as well as other organisations in the same market segment. The assessment approach should therefore map to existing engineering standards as well as quality standards. It should also provide an output that can be used easily

to benchmark against "best-in-class" organisations (April & Coallier 1995b). The number of process-related and quality standards and their relationships is continuously developing and even considered to be a quagmire (Sheard 1997; Paulk 2004). Still, certain groups of standards and guidelines to apply them constitute the background of and are referred to in the main software process and assessment approaches. From that point of view, they can be classified as quality, lifecycle and conformity standards.

2.3.1 Quality standards

In the late 1980s, the International Standardisation Organisation (ISO) published the ISO 9000 standard series, to rule the relationships between the purchaser and the supplier within a contractual agreement. ISO 9000 is a family of standards and guidelines that, when first published, contained: *Model for quality assurance in design, development, production, installation, and servicing* (ISO 9001:1987 1987) (for companies and organisations whose activities includes the creation of new products), *Model for quality assurance in production, installation, and servicing* (ISO 9002:1987 1987) (including the same material as ISO 9001 but without covering the creation of new products) and *Model for quality assurance in final inspection and test* (ISO 9003:1987 1987) (covering only the final inspection of the finished product with no concern for how the product was produced). ISO 9001 specified the minimum requirements for the quality system of any organisation to form the basis of a relationship between purchaser and supplier. The requirements covered the entire product lifecycle, including design, development, production, installation and servicing. The standard applied to a large set of products including hardware and software.

The family of ISO 9000 standards was updated in 1994, with the emphasis on quality assurance by using the concept of preventive action. A new version of ISO 9001 was published in 2000 (ISO/IEC 9001:2000(E) 2000), replacing all three previous standards (ISO 9001:1994 1994; ISO 9002:1994 1994; ISO 9003:1994 1994) of the 1994 issue. It also focused on processes and included concepts of process management, process performance measurements and continuous process improvement. As this version represented quite a radical change by shifting emphasis from inspection of the final product to monitoring and optimisation of a company's tasks and activities, the new version in 2008 (ISO/IEC 9001:2008(E) 2008) introduced only clarifications to the requirements of ISO 9001 and improved consistency with other ISO standards.

After more than twenty years' evolution, the ISO 9000 series of standards have come to provide requirements for defining, establishing and maintaining an effective quality assurance system for manufacturing and service industries. The current version of the standard (ISO/IEC 19011:2011(E) 2011), a corrected edition from 2009, forms the core of the current ISO 9000 family, together with ISO 9004 (ISO/IEC 9004:2009(E) 2009; Gallagher *et al.* 2011) and ISO 19011 (Gallagher *et al.* 2011). ISO 9001 is used when seeking to establish a quality management system that provides confidence in an organisation's ability to provide products that fulfil customer needs and expectations. There are five sections in the standard, specifying activities that need to be considered when implementing a quality management system: overall requirements for the quality management system and documentation; management responsibility, focus, policy, planning and objectives; resource management and allocation; product realisation and process management; and measurement, monitoring, analysis and improvement. All except the product realisation section are applicable to all organisation types (ISO 2009 2009).

ISO 9004 is used to extend the benefits obtained from ISO 9001 to all stakeholders in the company. ISO 9001 and ISO 9004 are compatible and can be used separately or in combination to meet or exceed expectations of customers and interested parties (ISO 2009 2009). "Both standards apply a process approach. Processes are recognized as consisting of one or more linked activities that require resources and must be managed to achieve predetermined output. The output of one process may directly be input to the next process and the final product is often the result of a network or system of processes" (ISO 2009 2009). The following quality management principles provide the basis for performance improvement: customer focus, leadership, involvement of people, process approach, system approach to management, continual improvement, factual approach to decision making and mutually beneficial supplier relationships (ISO 9000:2005 2005). ISO 9004 (ISO/IEC 9004:2009(E) 2009) gives guidance on a wider range of objectives of a quality management system than does ISO 9001, particularly in managing for the long-term success of an organisation. ISO 19011 (ISO/IEC 19011:2011(E) 2011) covers the area of auditing of quality and environmental management systems. It provides guidance on the audit programmes, the conduct of internal or external audits and auditor competence. It also provides an overview of how an audit programme should operate and how management system audits should take place. Effective audits ensure that an implemented QMS meets the requirements specified in ISO 9001.

Additionally, a guide dedicated to software (ISO 9000-3:1991 1991) was provided to help in applying the ISO 9001 requirements (ISO 9001:1987 1987) to a software organisation. The guidelines covered contract-based development where two parties require the demonstration of a supplier's capability to develop, supply and maintain the software product. It formed also the basis for TickIT certification. The guidelines suggested controls and methods for producing software that meet a purchaser's requirements, notably by preventing nonconformity at all stages, from development through to maintenance. The guidelines define the overall framework for a quality system and quality activities during lifecycle and supporting functions. They provide quite detailed guidance for software development organisations and set requirements for their performance assessment. The guidelines were updated in 1994 (ISO 9000-3:1994 1994) and 1997 (ISO 9000-3:1997 1997). The current version, published in 2004 (ISO/IEC 90003:2004(E) 2004), has wider scope and provides guidance for organisations in the application of ISO 9001 standard to the acquisition, supply, development, operation and maintenance of computer software and related support services. The application of the guidelines is appropriate to software that is part of a commercial contract with another organisation, a product available for a market sector, used to support the processes of an organisation, embedded in a hardware product or related to software services. Whatever the situation is, the organisation's quality management system should cover both software related and non-software related aspects of the business.

2.3.2 Lifecycle standards

The lifecycle models have their origins in the concept of the waterfall model, a sequential design process, often used in software development processes, in which progress is seen as flowing steadily downwards (like a waterfall) through the phases of conception, initiation, analysis, design, construction, testing, production/implementation and maintenance. The first formal description of the waterfall model was in the article (Royce 1970) by Winston W. Royce, although the concept of waterfall was obviously the result of a misunderstanding (Larman & Basili 2003), as Royce did not use the term "waterfall". Instead, he presented an iterative model that contained the following steps: system requirements, software requirements, analysis, program design, coding, testing and operations (Royce 1970: 330). The lifecycle phases were thereafter named accordingly.

Military organisations, commercial bodies, national standardisation organisations, aeronautics and space organisations and international standardisation organisations then proceeded to develop lifecycle models. Lifecycle standards cover software lifecycle models for software development and systems lifecycle models for embedded and information systems development. In the USA, the Department of Defence published a standard (DOD-STD-2167A:1988 1988) in 1988 for software development that was an update of the previous version from 1985. This document established uniform requirements for software development that are applicable throughout the system lifecycle and encompass software development management, engineering, testing, product evaluations, configuration management and support. The standard was designed to be used with another military standard (DOD-STD-2168:1988 1988) defining requirements for the development, documentation and implementation of a software quality program. These standards were used in building the original SEI model (Humphrey & Sweet 1987). Even earlier, NATO had published their own military standard for quality (AQAP-13:1981 1981), which was followed in the development of assessment models. A new standard for software development and documentation was established from the earlier such standards (DOD-STD-2167A:1988 1988; DOD-STD-7935A:1988 1988; MIL-STD-498:1994 1994). This standard was intended to be an interim standard, to be replaced by an international commercial standard, ISO 12207 (ISO/IEC 12207:1995(E) 1995).

In Europe, the European Space Agency (ESA) developed a comprehensive set of software engineering standards, the first complete version of which was published in 1984 (ESA.BSSC1-1:1984 1984). This was replaced by (ESA.PSS-05-0:1987 1987) in 1987 and (ESA.PSS-05-0:1991 1991) in 1991. The later version (Issue2) was reviewed by users, whose feedback and experiences the standard development group, called the Board for Software Standardisation and Control (BSSC), used to write a set of ten guides, covering both the lifecycle phases (Product Standards) (Mazza 1994) and the management aspects (Procedure Standards) (Jones *et al.* 1996). The standard describes the processes involved in the complete lifecycle of a single software project, from its inception to its retirement. The standard is split into lifecycle models and management activities, thus reflecting an “operations and control” pair. The lifecycle models include: definition of user requirements, definition of software requirements, architectural design, detailed design, transfer and operations and maintenance. The management activities contain: software project management, software configuration management, software verification and validation and software

quality assurance. Altogether there are about 200 mandatory practices, split roughly equally between lifecycle models and management activities. The descriptions contain material that defines what has to be done and gives some advice about how it can be done. The standard does not prescribe any particular methods to support the various phases but equally favours lifecycle models such as waterfall, incremental and evolutionary development (Jones *et al.* 1997).

The further development of the standard reflects the experiences and feedback received. One piece of feedback was that the standard was too heavy to be applied to small projects. In order to solve the problem, ESA published a guide to applying the standard in small projects (ESA.BSSC(96)-1:1996 1996) in 1996. When ISO 12207 standard became the main international lifecycle standard, ESA published within the European Cooperation for Space Standardization (ECSS) a new version based on ESA-PSS-05-0 called ECSS-E-40 (software engineering). The standard first appeared in 1996 (ECSS-E-40A:1996 1996) and was updated in 2003 (ECSS-E-40B-1:2003 2003) and 2005 (ECSS-E-40B-2:2005 2005) and published in the current issue in 2009 (ECSS-E-40C:2009 2009). The standard has six parts: space segment software, ground segment software, software lifecycle, software verification, validation and testing and software development for reuse. The new standard is based on ISO 12207 (ISO/IEC 12207:2008(E) 2008) and tailors it specifically for space projects. The standard is recommended to be used together with quality assurance standard ECSS-Q-80 (Software product assurance) (ECSS-Q-ST-80C:2009 2009), which is also based on ISO 12207. ECSS-Q-80 contains four parts, which are guidelines on: the reuse of pre-developed software, software process assessment and improvement, software dependability and safety methods and techniques, and for software metrication programme definition and implementation (Jones *et al.* 2002).

Today the main international standard for "software life cycle processes" is ISO/IEC 12207 (ISO/IEC 12207:2008(E) 2008), which establishes a common framework for software throughout its lifecycle, from conception through to retirement. It addresses the organisational context of these software processes, both from the system's technical viewpoint and from the enterprise's business viewpoint (Paulk 2004). It is defined at the process rather than the procedure level, including three categories of processes, namely primary (business and technical), supporting and organisational processes (Paulk 2004). Primary processes include: acquisition, supply, development, operation and maintenance. Supporting processes include: documentation, configuration management, quality assurance, joint review, audit, verification, validation and problem resolution, and

organisational processes consisting of management, infrastructure, improvement and training. In the description, each process is divided into activities and each activity into tasks. The description includes 23 processes, 95 activities, 325 tasks and 224 outcomes. The standard has strong TQM orientation and commitment. In addition to the specific and verification quality-directed processes (quality assurance, joint review, audit, verification, validation and improvement), each process has a built-in PDCA quality cycle (Tully *et al.* 1999). The first edition of the standard (ISO/IEC 12207:1995(E) 1995) was published in 1995 and the second and current edition (ISO/IEC 12207:2008(E) 2008) in 2008.

An international system lifecycle standard ISO 15288 was initiated by the group that created the ISO software lifecycle standard ISO/IEC 12207, and augmented by people with systems engineering expertise (Sheard 1997). The standard was intended to become companion in the systems engineering domain to ISO 12207 in the software engineering domain. Work on the standard started in 1994 and the first edition (ISO/IEC 15288:2002(E) 2002; ISO/IEC 15288:2002 2002) was issued in 2002. The standard dealt with the need to apply systems engineering principles to non-technical systems and to put more emphasis on management in systems engineering and its impact on business (Arnold & Lawson 2004). It provided a collection of system lifecycle processes based on system principles and concepts that govern their application. The standard provides support for technical systems, non-technical systems and systems composed of both technical and non-technical system elements. The processes in the standard were grouped into technical, project, enterprise and agreement processes. Technical processes include: definition of stakeholder requirements, requirements analysis, architectural design, implementation, integration, verification, transition, validation, operation, maintenance and disposal. Project processes consist of: project planning, project assessment, project control, decision making, risk management, configuration management and information management. Enterprise processes include: enterprise environment management, investment management, system lifecycle management, resource management and quality management. Agreement processes consist of: acquisition and supply. Each process is defined by a purpose, outcomes and activities. Altogether there are 25 processes, which have 123 outcomes derived from 403 activities. The standard was updated in 2008 (ISO/IEC 15288:2008(E) 2008; Arnold & Lawson 2004).

2.3.3 Conformity standard

In June 1991, the International Standards Group for Software Engineering approved a study period (ISO/IEC :1991 1991) to investigate the need for and requirements of a standard for software process assessment. The international study (ISO/IEC :1992 1992; ISO 9000:1992 1992, ISO/IEC :1992 1992)
International Standards Organization (ISO 9000 Series *et al.* 1992, ISO/IEC:1992 1992) concluded that there is an urgent need for a worldwide standard for process assessment. It found that the development should include trialling to provide usable output in an acceptable timescale and to ensure that the standard fully met the needs of its users. The resulting project was named SPICE (Software Process Improvement and Capability dEtermination). The project aimed to build on the best features of existing software assessment methods such as the Capability Maturity Model (CMM) (Paulk 1993), Trillium (Coallier *et al.* 1993), Software Technology Diagnostic (STD) (Craigmyle & Fletcher 1993) and the BOOTSTRAP method (Koch 1992) developed as part of a European ESPRIT project. In addition to preparing initial drafts of the standard, the project would also conduct application experiments whose results would be disseminated to the international community, assist in revision of the standard and market the overall activity (Dorling 1993).

The SPICE project was accordingly organised into seven working groups aiming to create material to establish a standard for software process assessment and improvement. The working groups had precise targets for generating the seven products that were to comprise the SPICE product suite. The products were:

- Introductory Guide (IG) (a top-level architecture document that describes how various parts of the standard fit together and provides guidance for their selection, use and creation of conformant variants)
- Baseline Practices Guide (BPG) (defines the goals and fundamental activities that are essential for good software engineering, and more precisely describes what activities are required but not how they are implemented)
- Assessment Instrument (AI) (a guide for designing instruments that extract data related to the processes undergoing assessment)
- Process Assessment Guide (PAG) (a document that specifies the assessment method, which in turn defines how to conduct an assessment using the assessment instrument and baseline practices)

- Process Improvement Guide (PIG) (provides guidance on how to use the assessment results for process improvement)
- Process Capability Determination Guide (PCDG) (provides generic guidance on how to use the results of an assessment for capability evaluation), and
- Assessor Training and Qualification Guide (ATQG) (provides generic guidance for the development of programmes designed to train people as assessors using this standard, and procedures for the qualification of assessors who intend to act in third-party situations) (Dorling 1993; Paulk & Konrad 1994a).

The SPICE project developed the initial sets of draft working documents between 1993 and 1995. The cores of the draft documents were used in two main rounds of trial assessments performed by the working group members in their own or customer organisations. In the first round (Woodman & Hunter 1996), so-called "SPICE version 1" was trialled. After that, ISO 12207 ideas were inserted into the draft documents and a second round of trials (Jung *et al.* 2001) was performed using so-called "SPICE version 2" documents. The drafts were then submitted to the normal standards balloting process. "During the course of these ballots, the SPICE Trials pursued a detailed series of investigations, aimed at demonstrating the extent to which the new International Standard met its original requirements, and validating its usefulness to the software development industry. The results of the Trials provided substantial empirical evidence supporting the approach to assessment embedded in the Standard, and also identified significant ideas for improvement" (Rout *et al.* 2007: 1484). The balloting process changed the original architecture considerably. The first version of the standard was released as a Technical Report (Type 2) in 1998, including: Part 1 – Concepts and introductory guide (ISO/IEC 15504-1:1998(E) 1998), Part 2 – A reference model for processes and process capability (ISO/IEC 15504-2:1998(E) 1998), Part 3 – Performing an assessment (ISO/IEC 15504-3:1998(E) 1998), Part 4 – Guide to performing assessment (ISO/IEC 15504-4:1998(E) 1998), Part 5 – An assessment model and indicator guidance (ISO/IEC 15504-5:1998(E) 1998), Part 6 – Guide to competency of assessors (ISO/IEC 15504-6:1998(E) 1998), Part 7 – Guide for use in process improvement (ISO/IEC 15504-7:1998(E) 1998), Part 8 – Guide for use in determining supplier process capability (ISO/IEC 15504-8:1998(E) 1998), and Part 9 – Vocabulary (ISO/IEC 15504-9:1998(E) 1998). Thereafter, the technical reports went through a study, review and revision period before becoming a full standard (Rout *et al.* 2007).

The first versions of the standard were published in five parts between 2003 and 2006. The purpose of the standard was to provide a framework for harmonising different approaches to assessing and improving the software process. It was meant to be used by an organisation as an aid to understanding the state of its own processes for process improvement or determining the suitability of its own processes for a particular requirement or set of requirements, or by an organisation for determining the suitability of another organisation's processes for a particular contract or set of contracts. It was designed also to provide a basis for a common approach to describing the results of process assessment for benchmarking between different but compatible models and methods (ISO/IEC 15504-1:2004 2004).

The first part of the standard was published in 2003, including "Part 2: Performing an assessment" (ISO/IEC 15504-2:2003 2003). The part is normative and sets requirements for process assessment and for process models used as a reference in an assessment. It defines also a measurement framework for evaluating process capability. The measurement framework defines nine process attributes grouped into six process capability levels, making up an ordinal scale of capability that is applicable across all selected processes. "This capability dimension can be applied, in principle, to any process e.g., the processes in ISO/IEC 12207 or ISO/IEC 15288 (System Life Cycle Processes)" (Paulk 2004). It also sets conformance requirements for the assessment process to be followed, the measurement framework to be applied (including capability dimension) and process models to be used as a reference (ISO/IEC 15504-1:2004 2004).

In 2004, three new parts were published containing "Part 1: Concepts and vocabulary" (ISO/IEC 15504-1:2004 2004), "Part 3: Guidance on performing an assessment" (ISO/IEC 15504-3:2004 2004) and "Part 4: Guidance on use for process capability determination" (ISO/IEC 15504-4:2004 2004). In Part 1 all other parts of the standard are introduced. It also contains the consolidated terms and definitions used in the standard. Part 3 is informative and provides guidance on meeting the requirements for performing an assessment as defined in Part 2, describes the capability dimension and rating principles and gives guidance for selecting process reference and assessment models. It also provides guidance for verifying the conformity of process reference models, process assessment models and performed process assessment. Part 4, likewise informative, gives guidance on using process assessment in process improvement and capability determination.

"Part 5: An exemplar process assessment model" (ISO/IEC 15504-5:2006 2006) was added into the ISO 15504 suite of standards in 2006. It was directly based on Technical Report (ISO/IEC 15504-5:1998(E) 1998), entitled "An assessment model and indicator guidance". It contains detailed descriptions of the process dimension, including indicators of process performance, and of the capability dimension, including indicators for process capability. There are also practical hints and guidance on performing an assessment. The exemplar model has been the basis of most of the assessments called "SPICE assessments".

The first versions of the standards focused on software development processes, but were later enhanced to cover six business areas, including: organisation, management, engineering, acquisition, support and operations. The issued standard now includes four additional parts, namely "Part 6: An exemplar system lifecycle process assessment model" (ISO/IEC 15504-6:2008 2008), "Part 7: Assessment of organisational maturity" (ISO/IEC 15504-7:2008 2008) and "Part 9: Target process profiles" (ISO/IEC 15504-9:2011 2011). The missing "Part 8: An exemplar process assessment model for IT service management" (ISO/IEC PDTR 15504-8:2012 2012 (target)) is due to be published in 2012.

3 Research process

The research of this thesis is constructive in nature and was performed in experimental cycles following the approach described in section 1.3. The research was carried out in international collaboration in industrial R&D projects between 1990 and 2004. The projects constituted the context of the research and established natural phases in time, as described in Table 3. The phases included different organisational settings and evolved iteratively, developing experimental frameworks that were adjusted to the research context. The research was done in collaboration with experimental researchers and software professionals who were experts in software development, software quality and/or software process assessment and improvement. This formed the practical settings for conducting the research (see Table 3).

The first phase of the research was performed in an ESPRIT project called BOOTSTRAP (Kuvaja & Koch 1992; Koch 1993) funded by the European Commission[31]. The phase may be called "initial development", as it encompassed the initial development of the methodology. The author of the thesis was a project manager and one of the main contributors in a research team assigned by the Department of Information Processing Science of the University of Oulu, which was a full partner of the project.

The second phase of the research and BOOTSTRAP methodology development was done within the BOOTSTRAP Institute, where the Department of Information Science of the University of Oulu became an Academic member. This phase may be called "methodology professionalisation", as it aimed to achieve a professional assessment and improvement methodology that conformed to the emerging standards of the area and had a supportive mechanism to further develop and sustain the methodology. The author of the thesis represented the Department in the management of the Institute and was a contributing member of the methodology development team in the Institute. The research and development within the Institute was threefold. One part of it was done internally in the Institute and other two parts in the International ISO project, SPICE (Dorling 1993) and the European ESSI ESPRIT project, TAPISTRY (Kuvaja *et al.* 1999b). In the SPICE project the author of this thesis represented the Department via the BOOTSTRAP Institute and became a member of the

[31] The funding for the Finnish partners of the project was provided by the Finnish Technology Development Centre – TEKES. Finland was not yet a member of the European Union but belonged to the European Economic Area.

development team and co-leader of the work package responsible for the development of the process improvement guide (Kuvaja *et al.* 1995b). In the TAPISTRY project, the author was project leader on behalf of the BOOTSTRAP Institute and one of the key developers of the TAPISTRY approach.

The third research and development phase took in place in a European IST project called PROFES (Oivo *et al.* 1999), together with wider European industry. In that project, the methodology was enhanced to include product development lifecycle processes and combined with Goal–Question–Metrics (GQM), quality improvement paradigm (QIP) and Experience Factory (EF) PROFES methodology. Therefore, the phase may be called "methodology enhancement". In the PROFES project, the author was a project leader of the research team of the Department of Information Processing Science in the University of Oulu, which was a full member of the project. The author contributed also as a member of the entire PROFES research team to the results of the project. Other related research projects that have provided enhancements for BOOTSTRAP methodology were: ESPRIT Project Drive-SPI[32], ESSI ESPITI project SPIT[33] and EUREKA/ ITEA project MOOSE[34].

[32] Esprit Project 20474 – Drive-SPI, "Risk Driven Software Process Improvement", funded by the European Commission.
[33] SPIT, "Software Process Improvement Training in Finland", a sub-project of ESPITI, "European Software Process Improvement Training Initiative", funded by the European Commission.
[34] European EUREKA/ITEA/MOOSE Project No. 01002, "Software Engineering Methodologies for Embedded Systems".

Table 2. Overall research process.

Phase		Research context	Research setting	Research cycles
1	1990–1993 BOOTSTRAP project	European R&D project	Interactive collaboration between experimental researchers and software professionals (experts)	Two entire experimental PDCA cycles with internal iterations in industrial settings
2	1993–2004 BOOTSTRAP Institute	Internal methodology development team	Interactive collaboration between researchers, software professionals and BOOTSTRAP assessors	Numerous experimental PDCA cycles with internal iterations, and external experience interchange
		International SPICE project	Experts' role in SPICE and interactive collaboration between researchers, software professionals, BOOTSTRAP assessors and international researchers and experts	Numerous internal experimental PDCA cycles with internal iterations, and internal and external experience interchange
3	1996–1999 BOOTSTRAP Institute and IST cooperation	European TAPISTRY project	Interactive collaboration between researchers and software professionals	Numerous experimental PDCA cycles with internal iterations and experience interchange
		European PROFES project	Interactive collaboration between researchers, software professionals, BOOTSTRAP assessors and international researchers and experts	Numerous PDCA cycles with internal iterations and internal and external experience interchange

Thanks to the settings and memberships of the teams and projects set out in the table, the author of this thesis was able to contribute both to the research and development of BOOTSTRAP methodology and to international standards in the area. The results have been published in international publications, as described in this thesis. Additionally, the development of the research methodology through learning and reflections on the literature form part of the thesis. The phases, their research settings and applied research cycles, outlined in Table 3, will each be described more precisely in the following sections.

3.1 Phase 1 – Initial development

Initial research and development of BOOTSTRAP methodology was conducted in the BOOTSTRAP project of the third research Framework Programme (FP-3) of the European Commission. According to its mission statement, "the project shall

fertilize the grounds for introducing modern software technology into industry. The means are through analysis of the current state of software engineering in industry, identification of the potentials of change and motivation for accepting new contexts for software engineering" (Koch 1993: 387). This set out the motivation for the project and was elaborated more specifically as described in section 1.3 above. The research in the BOOTSTRAP project was experimental and applied constructive approach, implemented in two full-scale PDCA cycles.

The *Plan* step in both cycles started from the project goals and background assumptions, performed state of the art and practice analysis based on the literature and experts' understanding, and produced a design rationale and requirements for a new version of a software process assessment and improvement approach. Researchers and experts of the partner companies worked together in the project team. In the first cycle, the project goals and background assumptions played a major role. In the second cycle, the analysed experiment results and industrial experiences and retrospective comments from the participating and target organisations became more important.

In the *Do* step, the process assessment and improvement approach was constructed in both cycles by following the requirements set out in the *Plan* step. The design rationale helped the building team to make choices and reminded them of the purpose of the specific requirements. The construct contained an assessment instrument in the form of a questionnaire, guidance on how to use the questionnaire, an assessment model as a description of the assessment process, an algorithm to produce and show the assessment results and guidance on how to record the results and collect experiences and lessons learned. The step was mainly performed by the researchers who constructed the instantiation of the methodology based on analysis of the external knowledge base and feedback received previously from the practice and experiments.

In the *Check* step, extensive experimental assessments were performed in numerous industrial settings. In the first cycle, assessment was performed in 20 Bosch GmbH software development units. In the second cycle in total 53 assessments were performed in different European countries by the partner organisations. The starting point for the first cycle was methodology version 1 and for the second cycle methodology version 2. The experiments were initiated and run by the researchers and their results communicated to the experts and to software engineers and the management of the assessment target organisation in common workshops. During the workshops, immediate feedback from and

experiences of the researchers, experts, software professionals and management were collected for analysis in the next step.

In the *Act* step, the performance, results and experiences of the assessment were analysed collaboratively by the research and development team and industrial experts from the partner companies of the project. The analysis yielded corrections to be made to the approach, new ideas and lessons for the next development iteration cycle. The project team was highly professional and had solid relevant industrial background and expertise. This gave a good starting point for publishing the findings from the cycle in academia and learning also from their feedback. At the end of the second cycle, the collected feedback, assessment experiences and lessons learned indicated many immediate changes to the approach, which the development team implemented.

3.2 Phase 2 – Methodology professionalisation

The BOOTSTRAP Institute was a spin-off of the BOOTSTRAP project and established at the moment when several new initiatives in the area of software process assessment and improvement emerged and the markets for professional third-party and second-party assessments started to grow. The Institute was a non-profit European Interest Economic Group (EEIG) established according to European law to bring BOOTSTRAP methodology into the professional market and take care of its further development. The research and methodology development within the Institute was organised into a methodology development team, led by the director of the Institute, which performed both internal development activities and cooperative activities with external partners (e.g. the European Software Institute) and initiatives (e.g. the SPICE project).

The development team continued internal research and development by carrying out experimental PDCA cycles. Design rationales and requirements were set up and constructed (*Plan* and *Do* steps) and then tested by the Institute members and licensees in practical assessments (*Check* step). The new features were communicated twice a year at organised assessors' days and assessors' update (for the new version) training courses. The feedback was collected (*Act* step), together with the assessment results registration, into the BOOTSTRAP data base managed by the Institute. The assessor feedback was then summarised as lessons learned. The data base itself was also used to analyse the validity of the assessments and assessment methodology versions.

The external development was organised so that the internal development team planned (*Plan* step) new features (including a design rationale and requirements) for the methodology, based on its own internal and continuous state of the art and practice analysis and deep interaction with external research projects and initiatives. This work was highly iterative and had internal iterations before the requirements for internal implementation (*Do* step) were released by the development team. Typically, the new plans and features were released for external consideration outside the development team, for example to the SPICE project (the case reported in this thesis) and feedback was collected in order to help in choosing the features for the implementation.

The main external development that impacted on BOOTSTRAP methodology happened in the SPICE project (Paulk & Konrad 1994b). The purpose of the SPICE project was to develop material (in practice the ISO Technical Report) for establishing a new ISO standard for software process assessment (Kuvaja *et al.* 1995a) and improvement (Kuvaja *et al.* 1995b). The BOOTSTRAP Institute participated the project in an expert role for the purpose of defining the contents of the forthcoming standard and at the same time to develop new versions of BOOTSTRAP methodology ((Kuvaja 1999; Hamann *et al.* 2000; Kuvaja *et al.* 1999a) that would conform to ISO 15504. All recognised process assessment and improvement methodology developers participated in the project including CMM (Humphrey & Sweet 1987; Humphrey 1989), Healthcheck (Norris *et al.* 1994), Trillium (Coallier *et al.* 1993; Coallier *et al.* 1994), Quantum (Barford *et al.* 1992), STD (Craigmyle & Fletcher 1993), Software Quality Improvement Method (Thomson & Mayhew 1994a; Thomson & Mayhew 1994b) and TickIT (TickIT 1992; Ould 1992). The author of this thesis acted as work package leader in the development of an improvement guide (Kuvaja *et al.* 1995b) in the SPICE project.

Three members of the development team were also members of the SPICE project, ensuring that the results of the SPICE work were available "online" for the development team. This information was valuable in choosing the features for implementation. The new versions of the methodology were then experimented with by following the same procedure as described in the internal development step (*Check* step) above. In addition, trial results of the SPICE project (*Check* step) were used in collecting assessment results and feedback for the internal analysis. In this case, the collaboration between the experimenters and subjects included also external experimenters and subjects of the SPICE project. The analysis was performed inside the development team and also by utilising the

analysis of the SPICE project in order to ensure that new BOOTSTRAP methodology versions would conform to the standard under development.

3.3 Phase 3 – Methodology enhancement

Since companies differ in size and internal culture, operate in different business areas, are in different stages of attaining maturity and have different goals, there was and still is a need for assessment to fit for different purposes. This was the motivation to start the research and development aimed at producing different versions of BOOTSTRAP methodology. The BOOTSTRAP development team and the SPICE project defined assessment modes to include internal assessment, second-party assessment or third-party assessment. In BOOTSTRAP methodology, internal assessment is performed either by an internal software process assessment team as professional assessment or by software engineers themselves as self-assessment. The professional internal assessment team performing a BOOTSTRAP assessment is led by a certified lead assessor and includes trained and certified process assessors, assessment facilitators and an assessment sponsor. The self-assessment is done by the software engineers themselves using a lightweight version of an assessment and improvement methodology, preferably supported by tools and templates.

The first step towards developing a specific version of BOOTSTRAP methodology focused on self-assessment. The methodology research and development team in the BOOTSTRAP Institute, together with the European Software Institute (ESI) (Doiz 1997), developed a self-assessment version of the methodology implemented in a software tool called BootCheck. The next research and development steps were performed in TAPISTRY (1996–1997), SPAM[35] (1996–1998) and SPIRE[36] (1997–1998) projects, which all were "sister projects" in the ESSI ESPRIT programme. The projects had similar aims to develop a self-assessment based approach to deploy software process improvement for SMEs in the European software industry. In parallel to the projects, BOOTSTRAP methodology was further enhanced in the PROFES (1997–1999) project.

[35] In the SPAM project, BOOTSTRAP methodology was used as a starting point for the development of a software portability assessment approach.

[36] In the SPIRE project (Sanders & SPIRE Partners 1998), the purpose was to encourage small to medium-sized enterprises to perform self-assessment based software process improvement.

The motivation for the TAPISTRY project was a desire to understand that, although a quantity of research results, tools, methods and recommended strategies was available, it was quite difficult for any small to medium-sized enterprises to choose an improvement approach and apply it in their company (Kuvaja *et al.* 1999b). The need for the project became even more urgent as it was recognised that small software development organisations are able to improve their software processes as well as large organisations (Paulish 1993; Damele *et al.* 1995) if they get the right and reliable information at the time when it is needed. Small to medium-sized enterprises were found to have the following weaknesses, preventing them from improving their processes:

- no substantive expertise
- scarce personnel and economic resources, and
- no knowledge of how to start or what experts to hire (Kuvaja *et al.* 1999b).

Accordingly, the TAPISTRY project was set up to develop an approach focusing on small and medium-sized enterprises that would be able to:

- inform SMEs of how process improvement techniques can be applied in practice in their settings
- provide SMEs with practical and implementable process improvement guidance specific to their current processes and company objectives
- demonstrate to SMEs that their software quality problems are not unique and can be solved without buying expensive tools or implementing heavy processes, which require additional staff to administer
- allow SMEs to compare their practices and policies with other SMEs, and/or
- provide SMEs with specific advice and consulting services by local and international experts in software process improvement without the SME incurring large consulting costs (ESSI 1998).

The TAPISTRY project was conducted under the guidance of the BOOTSTRAP development team. The partners of the project were members of the BOOTSTRAP Institute or BOOTSTRAP licensee organisations. The research process followed the PDCA cycle described in section 1.3, with tens of experiments carried out by project members around Europe. The development team made a plan (*Plan* step) to construct a workshop for educating small to medium-sized organisations about software process improvement. The workshop was iteratively constructed (*Do* step) and experimented with (*Check* step) within the project. The experiments were carried out in the customer organisations of the

project partners and lessons learned were collected from the workshop participants and instructors. The lessons were analysed and immediate corrections made to the TAPISTRY approach. Additionally, the project team organised public TAPISTRY workshops within the ESPITI programme (validation cycle) and feedback from the participants of those workshops was collected and published (Kuvaja *et al.* 1999b). In internal development, the researchers, TAPISTRY workshop instructors and software professionals and representatives of partner organisations of the project worked interactively together.

The European IST project called PROFES[37] (Oivo *et al.* 1999) was a three-year project starting in 1997 and ending in 1999. In the PROFES project, the idea was to start from product improvement and discover how it would be possible by improving the process that produces it. For that purpose, BOOTSTRAP methodology was enhanced to include product development lifecycle processes. The main research effort in PROFES was, however, focused on discovering how to combine Goal–Question–Metrics, Quality Improvement Paradigm and Experience Factory approaches with the BOOTSTRAP methodology in such a way as to produce results conformant with the emerging new standard developed in the SPICE project. The goal was also to end up with a new product-focused software process improvement approach that would contain the approaches mentioned above as interchangeable elements, which might be replaced with another similar approach if needed in practical settings. Fortunately, all the approaches were goal-driven in nature and the members had personally participated in the development of the background approaches and paradigms.

The PROFES project was a free-standing IST project funded through the Fourth Research Framework Programme by the European Commission. The project was organised into work packages according to the Quality Improvement Paradigm. The project followed the PDCA cycle, the BOOTSTRAP methodology being used in the role of measurement within the GQM approach (*Plan* step). Then it was adopted and combined with the goals defined by the GQM paradigm (*Do* step), and tested with GQM for defining process capability in several experimental assessments (*Check* step) performed in partner companies (Dräger Medical Electronics, Ericsson Finland and Tokheim (earlier Schlumberger RPS)). The feedback was collected in iterative cycles and after several cycles the result was reported (*Act* step) as PROFES methodology. The researchers

[37] An ESPRIT Project No. 23239, PROFES (PROduct Focused improvement of Embedded Software processes), funded by the European Commission during 1997–1999.

(experimenters) had a background in BOOTSTRAP methodology, GQM, QIP and/or Experience Factory. The representatives of the industrial partners worked in close cooperation with the researchers and other personnel from the same companies provided feedback on the experiments.

In the PROFES project, the author was a project leader of the research team of the Department of Information Processing Science in the University of Oulu, which was a full member of the project. The author was also work package leader of the "assessment" work package and a member of the entire PROFES research team developing the PROFES methodology. Specifically, the author contributed to the development of the enhancements to BOOTSTRAP methodology to include product development lifecycle processes and close connection to the GQM and QI paradigms.

4 Analysis and main results

In this chapter, the research conducted and main results are analysed through the research phases and applied research approach, as described in chapter 3. The research effort is iterative and the target artefact, BOOTSTRAP methodology, is under continuous evolution. Each phase produces its own results, which contribute to the research questions as to their own. Therefore, answers to research problems will emerge from each research phase, building partly on the results of the previous phase but also responding to the new specific requirements of the phase. The strength of answers is evaluated and presented in Table 3, using a relative ascendant scale from one to three expressed by the number of "+" signs. This indicates how the research and results are focused on solving the research questions. The outline of the research results and their relative allocation to the research questions and presentation in the original publication for each phase are presented in the table.

Table 3. Research results outline.

Phase	Research results	Research problems	Articles
1990–1993 BOOTSTRAP project	BOOTSTRAP methodology versions 1.0 and 2.2	Q1 ++ Q2 + Q3 +++ Q4 +	1.
1993–2004 BOOTSTRAP Institute	BOOTSTRAP methodology version 2.3	Q1 +++ Q2 +++ Q3 +++ Q4 ++	2. 3.
	BOOTSTRAP methodology versions 3.0 and 3.1	Q1 +++ Q2 +++ Q3 +++ Q4 ++	4.
1996–1999 BOOTSTRAP Institute and IST cooperation	Guided workshop for self-assessment and improvement supported with BootCheck tool	Q1 + Q2 +++ Q3 +++ Q4 +	5.
	BOOTSTRAP methodology version 3.2 and element in PROFES methodology	Q1 +++ Q2 +++ Q3 +++ Q4 +++	6.

The research process was performed in four evolutional phases, as described in the previous chapter (Fig. 3). In the first phase, the initial research and development of the BOOTSTRAP methodology resulted in a research prototype, which was then further developed in the subsequent phases. The second research and development phase started with the establishment of a spin-off organisation from the project of the first phase. The spin-off was the BOOTSTRAP Institute, which identified and organised the research and development for bringing BOOTSTRAP methodology into markets worldwide. In the third phase, the methodology was enhanced to fit different assessment modes, scopes and goals. Finally, in phase four, the research was aggregated to a proposal for a new approach called Empirical Explorative Research. In this chapter, the research phases are renamed according to the main results from the viewpoint of BOOTSTRAP methodology evolution. The contents and main results of each phase are analysed and presented in the following sections.

4.1 Phase 1 – Research prototype

The first versions (1, 2.1 and 2.2) of BOOTSTRAP methodology were developed in the BOOTSTRAP project. The specific goals of the project included the following targets:

- to develop a software assessment methodology
- to validate the assessment methodology by applying it to a number of companies to define their process capability and organisational maturity profiles
- to develop an improvement methodology, including setting the targets and generating improvement plans, and
- to validate the improvement methodology by applying it to a number of companies (Kuvaja *et al.* 1994: 18).

As described in the previous chapter, the research process comprised two main cycles. The first cycle started from the motivation and overall goals of the project. In the planning step, it was decided to apply CMM methodology as much as possible and enhance it with the main features of ISO 9001 and ISO 9000 part 3 requirements. Starting from the design rationale, BOOTSTRAP questionnaire version 1.0 was constructed and tested in practice by using the CMM approach in defining the maturity levels of the target software unit. In total, more than 20 assessments were conducted in ten different Bosch GmBH software units. The

feedback from the target software units indicated that it was not enough to show the maturity level only as an integer indicating the overall level of the organisation, as derived through the project level assessment, but more precise result was needed. The assessors' feedback focused mainly on categorising the answers to the questionnaire using the binary scale ("yes" or "no"). The researcher also faced the problem of finding a way to compare the results of the experimental assessments with one another and the results of original CMM assessments.

The second main research cycle started from the overall project goals, feedback from the first research cycle and revisiting the background approaches indicated by the design rationale of the first main cycle. A request to apply ISO 9000 principles and, from the target organisations, to provide more precise results for the maturity led to the preparation of separate questionnaires for SPU and project level. The binary scale having proved lacking, the design rationale divided the "yes" answer into "a bit yes", "quite a lot yes" and "fully yes". The statements of the design rationale were incorporated into the new construct of the model. The result was intermediate version 2.1 of the model, including separate questionnaires for SPU and projects, and a four-point (scoring) scale for answering the questionnaires. The model was immediately tested to get quick feedback for planning and help in deciding whether to proceed towards a version that would completely satisfy the requests. Positive feedback from the quick experiments confirmed that two separate questionnaires and the new four-point scale worked quite well in practice. Specifically, the two levels of questionnaires aligned the approach with the ISO 9000 certification schema. Additionally, the four-point scale was found to be used also in the Rocky Mountain Questionnaire (which allowed a variety of response types: binary, four-point scale and six-point scale).

The planning phase was repeated, taking into account the positive feedback to the initially implemented changes. In light of the project's demands to align the methodology towards European needs and continuous improvement, requests were met to complete implementation of ISO 9001 by following ISO 9000 part 3 guidance and to apply the European Space Agency lifecycle standard. The assessment experiences reported by NASA (Bush 1991) and new version of CMM (version 1.1) prompted a revisiting of the questionnaire contents and maturity level definitions. All these things together set the design rationale for improvements to the approach.

Accordingly, BOOTSTRAP version 2.2, a draft version for the experiments, contained a new process structure, changed questions, new questions added into each process area and new naming conventions. The process structure was divided into organisation (Q), methodology (M) and technology (T), which became known as process areas. New questions were added in line with the NASA experiences. The processes started to be called "attributes", as they together constituted the maturity of the target organisation and the questions measuring process took on the form of statements postulating the way working. The methodology was then tested in 40 assessments performed in 11 SPUs in external companies and, after minor internal iterations, completed to form version 2.22.

At that point, one important piece of feedback from the target companies and retrospectives of the assessments remained unresolved. The main feedback received was a request to show the maturity levels for both the SPU and the projects separately and in more detail than just an integer number indicating the maturity level. Specifically, the managers wanted to know whether the maturity level was only just on that level or already moving towards the next level. Analysis of the assessment results and assessors' experiences indicated that each partner company had used the same questionnaire but applied its own method of performing the assessments and interviews, interpreting the questions and scoring the findings, and rating and presenting the results.

One result of the project was organising the IPSS[38]-Europe International Conference on Lean Software Development on October 22nd to 23rd with tutorials on October 21st in 1992 in Stuttgart, Germany. The conference may have been the first in the world touching on the topic of "lean software production". The conference issued the following statement: "Lean Software Development hits the Software Community while it is still struggling to understand and control the software development process. What should we learn from concepts of lean production? What are the specific rules of software development not to be violated? How should we describe Lean Software Development?" The conference presented the initial ideas of the questions and statements and offered a good basis for the BOOTSTRAP methodology development team to evaluate their results within the bigger picture.

[38] IPSS: Information Processing Systems and Software Sub-programmes, European Commission, DG XIII: Telecommunications, Information Industries and Innovation.

4.2 Phase 2 – Commercial methodology

The BOOTSTRAP project ended in February 1993 at the same time as the first meeting of the SPICE project was held. At the end of the project, it was agreed to establish an institute to continue the methodology development work and bring it to the commercial markets, as high demand was recognised, particularly in Europe. The key developers (including the author of this thesis) continued research and development of the methodology during the interim period of the establishment of the Institute. The research and development team took BOOTSTRAP methodology version 2.22 and feedback collected at the end of the BOOTSTRAP project as a starting point for planning the new version of the methodology. The resulting design rationale included requirements to present the assessment results in more detail, to create a common way to perform the assessments, to establish a mechanism to collect and present the assessment results and to support assessors in performing the assessments.

The result of the implementation of the requests was the first commercial version (2.3) of the BOOTSTRAP methodology. It comprised a division of the maturity levels into quartiles, an algorithm (Messnarz 1994) to produce ratings based on scoring of the questionnaire, presentation of the capability of the processes in the form of an attribute profile, initial computer-based tools for supporting the scoring, rating and collecting the assessment results and common templates for performing the assessments. The methodology was taken immediately into commercial use by the partner companies of the BOOTSTRAP project and founding members designate of the Institute. The methodology content was also introduced worldwide in several publications (Kuvaja & Bicego 1993; Haase *et al.* 1994; Kugler & Messnarz 1994). This was the research and development status at the moment when the BOOTSTRAP Institute was initiated.

The BOOTSTRAP Institute, the owner of the Bootstrap methodology, was established (April, 1994) as an independent organisation and as a spin-off from the BOOTSTRAP project. The Institute was a non-profit organisation dedicated to the continuous development of the methodology. The main task of the Institute was to provide fair and equal access to the methodology while allowing all interested parties to participate in its evolution. In order to protect the intellectual property rights of the methodology, the BOOTSTRAP trademark was registered in Europe and the USA. Later, the Institute set out the following objectives for BOOTSTRAP methodology development in the future, which were to:

- keep the methodology consistent and preserve its unique features (see unique features more precisely in sections 3.4 to 3.6)
- keep the methodology up to date with main international standards such as ISO 9000, results of the SPICE project, new CMM versions, etc.
- keep the BOOTSTRAP data base up to date
- maintain supportive computer-based tools
- ensure that all BOOTSTRAP assessments would be performed according to BOOTSTRAP methodology principles
- control the use of the methodology, following the terms of the licence agreements, and
- guarantee that the methodology would keep its position as the leading European software process assessment and improvement methodology on the market (Kuvaja 1995b).

The objectives were already applied in the development of the first commercial version of BOOTSTRAP methodology. Further development of the methodology was affected by the SPICE project (Dorling 1993; Kuvaja *et al.* 1995a) in which the BOOTSTRAP methodology development team participated intensively (see Paulk & Konrad 1994a). That work started just after completion of the BOOTSTRAP project. The overall goals of the new methodology version development were to follow the above-mentioned objectives, and the specific goals were to ensure full conformity between BOOTSTRAP methodology and the emerging ISO standard for software process assessment and improvement, better known as SPICE (ISO 15504). Development started with these goals and a thorough revision of BOOTSTRAP methodology version 2.3, SPICE version 1.0 requirements and ISO 12207 lifecycle standard requirements. The result of combining these design rationales was BOOTSTRAP methodology version 3.0, an updated and documented assessment (reference and capability) model with SPICE conformant process and capability dimensions. The process dimension documents the BOOTSTRAP processes and practices. The new BOOTSTRAP processes were developed by carefully revising the earlier BOOTSTRAP process model and updating it to include requirements from IS0 12207, the SPICE 1.0 process model, ESA PSS-005, IS0 9001 and IS0 9000-3. The capability dimension was also aligned to the SPICE capability concepts, including six levels from 0 to 5 (where 0 represents the lowest and 5 the highest capability level). The main change was that all capability levels were applied to each process of the process dimension, thus permitting a common improvement strategy. In addition

to standard SPICE results, BOOTSTRAP kept its feature for generating synthetic profiles using quartiles within the capability levels, as in its previous versions. Additionally, BOOTSTRAP 3.0 kept its original features for goal-oriented improvement and technology support evaluation. The BOOTSTRAP Institute participated in the SPICE version 1.0 trials by performing full experimental assessments using the BOOTSTRAP version 3.0 methodology and reporting the results to the SPICE project. The feedback received from SPICE formed the basis for further BOOTSTRAP methodology development.

The SPICE project continued development of the (forthcoming ISO) standard requirements and published version 2.0 of the work results. The version also now applied ISO 12207 requirements for the lifecycle processes, which thus became aligned with BOOTSTRAP version 3.0. Additionally, new process descriptions and updates for the previous versions were added. Collaboration with the SPICE project (1993–2004) provided the Institute with a good basis for further development and validation of the methodology. As a result, BOOTSTRAP methodology version 3.1 was developed, which included all the features that the SPICE project had identified. The version became equivalent to the CMM-I Continuous model while still keeping BOOTSTRAP methodology specific features. The new SPICE requirements (SPICE version 2) were also trialled, as was the new version of BOOTSTRAP methodology.

During the collaboration with the SPICE project, BOOTSTRAP methodology contributed greatly to the SPICE results. The following specific features from the original BOOTSTRAP methodology were incorporated into the contents of SPICE:

- define the scope of the assessment according to the goals of the assessment target organisation
- use a four-point scoring scale
- present capability separately for each process, and
- present the assessment results as capability profiles.

What was adapted from the SPICE project to BOOTSTRAP methodology was mainly the definition of the processes for the assessment (reference) model.

4.3 Phase 3 – Different versions

In parallel with the developments emanating from SPICE, the BOOTSTRAP methodology was quite actively used, especially in Europe, for software process

assessment and improvement. Experience with the methodology showed that European organisations had quite different needs for assessment, depending on how advanced they were in software process improvement. The extreme ends were, on one hand large, leading-edge organisations with a fairly mature process and a long history of process assessment and improvement, and on the other hand organisations that did not have any process management experience and were not yet aware of its potential benefits. Therefore, there was a need for a simple self-assessment approach for beginners that could create awareness and motivate them to continue to more serious software process improvement. In addition, the advanced organisations needed more sophisticated and professional software process assessment and improvement approaches (Bicego & Kuvaja 1996).

As companies differ in size and internal culture, operate in different business areas, are in a different stage of achieving maturity and have different goals, there was and still is a need for assessment fit for different purposes. This, along with the findings above, was the motivation for starting the research and development aimed at producing different versions of BOOTSTRAP methodology. The BOOTSTRAP development team and the SPICE project defined assessment modes to include internal assessment, second-party assessment or third-party assessment. In BOOTSTRAP methodology, internal assessment is performed either by an internal software process assessment team as professional assessment or by software engineers themselves as self-assessment. The professional internal assessment team performing a BOOTSTRAP assessment is led by a certified lead assessor and includes trained and certified process assessors, assessment facilitators and an assessment sponsor. The self-assessment is performed by the software engineers themselves, using a lightweight version of an assessment and improvement methodology, preferably supported by tools and templates.

As described in section 3.3 specific version of BOOTSTRAP methodology focused on self-assessment was developed in collaboration between the BOOTSTRAP Institute and the European Software Institute. The result was implemented in a software tool, called BootCheck (Doiz 1997), which included tool and template support and a lightweight version of the methodology for self-assessment. Its role was to give an idea of software process assessment and improvement, and provide “rough” overall capability profiles for the assessed processes. The next steps were performed in TAPISTRY (1996–1997), SPAM (1996–1998) and SPIRE (1997–1998) projects, which all aimed to develop a self-assessment based software process improvement approach for SMEs. In the SPAM project, BOOTSTRAP methodology was used as a starting point for the

development of a software portability assessment approach. In the SPIRE project (Sanders & SPIRE Partners 1998), the purpose was to encourage small to medium-sized enterprises to perform self-assessment based software process improvement. In parallel to the projects, the BOOTSTRAP methodology was further enhanced in the PROFES (1997–1999) project, where it was combined with GQM, QIP and Experience Factory approaches and enhanced with product management and lifecycle processes.

The motivation for the TAPISTRY project was the understanding that, although research results, tools, methods and recommended strategies were available in some number, it was quite difficult for any small to medium-sized enterprises to choose an improvement approach and to apply it in their company (Kuvaja *et al.* 1999b). The need for the project became even more urgent as it was recognised that small software development organisations might improve their software processes just as much as large organisations (Paulish 1993; Damele *et al.* 1995) if they got the right, reliable information about software process improvement at the time when it was needed. Small to medium-sized enterprises found they had certain weaknesses that prohibited them from improving their processes. One thing hindering them was that they could not afford to maintain substantial expertise of software process improvement within their companies, but instead had to hire external expertise. They also suffered a scarcity of resources, both human and financial, just when they urgently needed them. Another problem was that they did not know how to start the improvement and which experts to hire (Kuvaja *et al.* 1999b).

Accordingly, the TAPISTRY project's approach with regard to the small and medium-sized enterprises aimed to:

- inform SMEs of how process improvement techniques can be applied in practice in their settings
- provide SMEs with practical and implementable process improvement guidance specific to their current processes and company objectives
- demonstrate to SMEs that their software quality problems are not unique and can be solved without buying expensive tools or implementing heavy processes, which require additional staff to administer
- allow SMEs to compare their practices and policies with other SMEs, and/or
- provide SMEs with specific advice and consulting services from local and international experts in software process improvement, without the SME incurring large consulting costs (ESSI 1998).

The motivation and project objectives set the design rationale for the research and development.

The design rationale was fulfilled by applying a learning-by-doing paradigm (Merrill *et al.* 1995), high consulting expertise from the project partners and adoption of a downscaled model of the BOOTSTRAP assessment methodology, implemented in the BootCheck tool. The result was a two-day workshop, including a step-by-step self-assessment and improvement exercise focused on the participants' own organisations and tutored by the workshop instructors, who were software process improvement experts. During the workshop, the participants were instructed to perform a self-assessment using the BootCheck tool, analyse the results and prepare an improvement plan for their own organisation, the "doing" part of the approach. The purpose was to urge the participants to learn why it is important to do software process improvement and what the value of it for their own organisation might be, what the main problems in their own organisation are (in order to understand where to start software process improvement) and what the software process improvement activities needed in their own organisation might be (documented in the form of the software process improvement plan).

The workshop suite includes also supportive material for the participants and a guide and training course for the instructors and coaches. The TAPISTRY approach was validated through a large number of experimental workshops and instructor training sessions in Finland, Germany, the UK, Norway and Iceland (Kuvaja *et al.* 1999b). Thereafter, the TAPISTRY approach was licensed by the BOOTSTRAP Institute and has been used internally in licensee companies for internal SPI training, in SPI courses of higher education in universities and in commercial consultancy for guiding a company towards taking the first steps in SPI.

The purpose of the PROFES project was to develop a product quality improvement-driven software process improvement methodology. The motivation came from the practice of software process improvement, where the aim was to improve the quality of the product that the process produced and not just the quality of the process. It was understood that this viewpoint was not emphasised in any available process improvement approach, which tended instead to rely on the assumption that the better the process quality is, the better the quality of the product it produces. In the PROFES project, the logic started from the opposite direction. First the quality of the product was analysed, then the targets for its improvement were set, the processes were assessed and only then were the

required process improvements made. From that starting point, the design rationale for the research and development was set to develop a methodology by integrating software process assessment, software measurement and organisational learning guided by the relationships between product and process characteristics (Bicego *et al.* 1997).

The resulting PROFES methodology combines and enhances the strengths of goal-oriented measurement, process assessment, product and process modelling and Experience Factory. PROFES goal-oriented measurement methodology GQM (Goal–Question–Metric) and BOOTSTRAP (ISO 15504 compliant) process assessment and improvement methodology provided the framework for analysing the status, setting the targets and monitoring progress. The BOOTSTRAP assessment model was enhanced to fulfil the requirements stated for embedded systems development. The enhancements included product management, development lifecycle and related processes. ISO 9126 standard was also used as a background reference for the product quality characteristics but the actual product measurements are based on the GQM approach. Process modelling was needed to describe the software development processes. Furthermore, PROFES introduced a method for establishing product process dependencies (PPD), which are a core element of the method and so far unique in the world of software process improvement. PPDs connect the product characteristics into the process characteristics (Hamann *et al.* 1998). Additionally, the entire structure of the method followed the Quality Improvement Paradigm and applied organisational learning according to the Experience Factory approach. The resulting approach included the following six steps:

- characterise the process improvement environment (product, processes)
- set goals for product improvement
- plan process changes and implementation
- execute product development project according to plans
- analyse data and findings, and
- package results for reuse (PROFES project team 2000).

The BOOTSTRAP methodology is mainly included in the first step, but has a role also in all the subsequent steps except execution. The initial versions of the methodology were tested at full scale in three industrial organisations, which offered real-life experimental environments for the methodology development and validation.

5 Introduction to original publications

In this chapter, the original publications, their main results and the author's role and contribution are presented. The chapter starts by introducing the publications, their origin and the author's involvement in them. The contents and main results of each publication are presented in separate sections, as well as the author's role in and contribution to their compilation.

5.1 Publications and author's contribution

The works included in the thesis were published between 1994 and 2004 in journals and peer-reviewed international conference proceedings. The publications present the results of work done in many international and national research and developments projects and initiatives.

The starting point for the author's work in the area was to join an ESPRIT research and development project called BOOTSTRAP from 1992 to 1993 in the role of project manager of the University of Oulu research group and active member of the methodology research and development team. During that activity, BOOTSTRAP methodology version 2 and its final version 2.2 were developed.

Subsequently, the members of the project established the BOOTSTRAP Institute as a new European Economic Interest Group (EEIG), a non-profit organisation. The purpose of the Institute was to ensure continuous development of BOOTSTRAP methodology and to bring it to the market and support and manage its use in practical software process improvement in industry. The author of this thesis was a board member of the Institute with full responsibility for and participation in the methodology development (1994–2004). BOOTSTRAP methodology version 2.3 was the first release by the Institute.

In 1993, the BOOTSTRAP methodology development team joined an international project called Software Process Improvement and Capability dEtermination (SPICE). The author became a member of the project with special responsibility for and expertise in all the process assessment and improvement research from the BOOTSTRAP methodology development viewpoint. In 1994 the author was also nominated as product manager of Process Improvement Guide development in the project, a position he held until 1998.

BOOTSTRAP methodology development continued further in an ESPRIT project called Product Focused improvement of Embedded Software processes (PROFES) with the purpose of combining the methodology with Goal–

Questions–Metrics (GQM) and enhancing BOOTSTRAP with product development assessment and improvement features. The author was project manager of the University of Oulu project team and one of the key developers of the methodology.

In parallel with the PROFES project, the BOOTSTRAP Institute, together with the European Software Institute (ESI), developed a self-assessment tool called BootCheck (Doiz 1997). The tool was based on a downscaled version of the BOOTSTRAP methodology and helped users to perform a self-assessment with interactive guidance. The author of the thesis managed the development activity within the BOOTSTRAP Institute and participated as a key developer in devising the downscaled version of the methodology.

BootCheck and the downscaled version of BOOTSTRAP methodology were then further developed in an ESSI ESPRIT project called a Tailored Application of Software Process Improvement Techniques for Small Enterprises (TAPISTRY) during 1996–1997, where they were adopted into a new workshop-based assessment and improvement approach. The author of the thesis was project manager of the project and one of the main contributors to the development of the methodology.

A number of articles and conference papers have been published arising out of this work. Six of them have been chosen here to present different viewpoints of the BOOTSTRAP assessment and improvement methodology and reflect the different development expectations and experimental cycles that were part of continuous international development in the area. The contents and roles of the publications and the author's role and contribution are summarised in Table 4 and elaborated in more detail in the following sections of the chapter.

Table 4. Original publications and author's contribution.

Publication	Source	Contribution
1. Kuvaja P., Bicego A. (1994): "BOOTSTRAP – a European Assessment Methodology", Software Quality Journal	BOOTSTRAP Project, BOOTSTRAP Institute	member of the project, one of the main developers of the methodology, main author of the article
2. Similä J., Kuvaja P., Krzanik L. (1995): "BOOTSTRAP: A Software Process Assessment and Improvement Methodology", International Journal of Software Engineering and Knowledge Engineering	BOOTSTRAP Institute, SPICE Project, BOOTSTRAP book writing	member of the institute, one of the main developers of the methodology, member of the development team, member of the project, main author of the book, co-author of the article
3. Bicego A., Kuvaja P. (1996): "Software process maturity and certification", Journal of Systems Architecture	BOOTSTRAP Institute, SPICE Project	member of the institute, member of the development team, member of the project, co-editor of work package in the project, co-author of the article
4. Kuvaja P. (1999): "BOOTSTRAP 3.0 – A SPICE Conformant Software Process Assessment Methodology", Software Quality Journal	BOOTSTRAP Institute, SPICE Project	member of the institute, one of the main developers of the methodology, co-editor of work package in SPICE project, author of the article
5. Kuvaja P., Bicego A., Palo J. (1999): "TAPISTRY – A Software Process Improvement Approach Tailored for Small Enterprises", Software Quality Journal	BOOTSTRAP Institute, TAPISTRY Project	director of the Institute, one of the main developers of the methodology, member of the project, main author of the article
6. Maansaari J., Kuvaja P., Taramaa J., Seppänen V. (1999): "Definition of an Embedded Systems Process Frame to Enhance ISO 15504 Conformant Assessments", International Conference on Product Focused Software Process Improvement – PROFES	PROFES Project	member of the project, one of the main developers of the methodology, co-author of the paper

5.2 BOOTSTRAP – a European assessment methodology

This article was published in *Software Quality Journal* in 1994, presenting the origin and motivations, development, contents and early validation data of a European software process assessment and improvement methodology called BOOTSTRAP. The methodology was developed in a three-year ESPRIT project

called BOOTSTRAP[39], by keeping the original CMM (also known as an SEI model) as the main background model and extending it with features based on the guidelines from ISO 9000 quality standards and the ESA (European Space Agency) lifecycle model (ESA standard No. PSS-005). The extensions were made in order to fit the methodology to the European context and to obtain more detailed capability profiles, in addition to maturity levels separately for organisations and projects. The resulting methodology version (version 2.2) includes assessment process description, questionnaires and an algorithm for maturity and capability determination, tool and data base support for assessment data collection and benchmarking.

BOOTSTRAP methodology originated in the European context, where approximately 80% of all software was found to be developed in small to medium-sized enterprises (SMEs) (Koch 1992) and 70% in non-IT sectors of the economy (industry). Therefore, the ability to produce software efficiently, in a timely manner and with consistently high quality was becoming increasingly important to enable industries across Europe to maintain and enhance their competitiveness. The BOOTSTRAP project was consequently identified and accepted as one of the pathfinders for the European Systems and Software Initiative (ESSI) (Koch 1992).

BOOTSTRAP methodology development was especially focused on the needs of the European software industry. Therefore, the specific goals of the development were that BOOTSTRAP methodology should be suitable for software process assessment-based improvement in both SMEs and large companies and should take into account the selection of software development methods and lifecycle models, as well as processes and practices, and especially adopt international software standards applied in Europe.

The first version of BOOTSTRAP methodology (version 1.0) was developed in the BOOTSTRAP project by taking the CMM (version 1.0) as the basic reference (capability and maturity levels and process definitions) and extending it with the features of ISO 9000 quality standards and the European Space Agency lifecycle model. The CMM model was applied in such a way that the results of a BOOTSTRAP assessment could be comparable to CMM results. The adopted ISO 9000 standards included guidelines for a company-wide quality system and a

[39] The BOOTSTRAP project was ESPRIT project Number 5441, funded by the European Commission, which preceded and had a role in preparing for the European System and Software Initiative – ESSI programme.

distinction between organisation, methodology and technology. The ISO 9000 quality system certification schema gave us the idea of assessing the Software Producing Unit (SPU) and its projects separately. This separation is not included in the CMM model. Applying the lifecycle model defined in ESA PSS-005 enhanced the process suite of the process reference model used in CMM[40], especially with the addition of support processes and the contents of the ordinary lifecycle processes common in Europe. That made the BOOTSTRAP reference model more complete and applicable to the European context. Experiments were done within the BOOTSTRAP project on the methodology by performing assessments in Bosch software development units in Europe, the aim being to define the maturity levels of the assessed organisations (as in the original CMM assessments), gain experience of the workability of the methodology in practice and collect feedback about the validity of the methodology version used.

Based on the experiences and feedback collected during the first experimental cycle, a second version of BOOTSTRAP methodology was developed. The second version included an algorithmic way to allow processes to attain capability spanning more than one capability level, capability and maturity levels to include quartiles inside the levels, and individual questions in the questionnaires offering answers on a scale of five values, represented most commonly with such adjectives as “absent”, “weak”, “fair”, “extensive” and “non-applicable”. All of them were totally new features for software process assessment methodologies developed at that time. The questionnaire version 2.1 included enhancements and updates to process coverage based on new material published about CMM version 1.1, the “Rocky Mountain Questionnaire”, SQA process assessment by NASA (Bush 1990) and HP’s SQPA approach (Grady 1992). The questionnaire was tested internally in the project and then version 2.2 was devised to include classification schemas both for projects and SPUs in order to help in systematic assessment results collection and comparison. At the end of the BOOTSTRAP project, the methodology version was 2.2.

The results at the end of BOOTSTRAP project and the experiences of its use were published in the article. The maturity distribution profiles presented included results of all experimental assessments carried out during the project but now included in the BOOTSTRAP data base and reported in the format of the new BOOTSTRAP data base output. The feedback presented in the article was

[40] The processes used in the CMM are based on US Department of Defense lifecycle model standard DoD-STD 2167A.

collected during the second experimental cycle of the project and relates to methodology version 2.2. The results represented a strong incentive to develop the methodology further for commercial use.

5.3 BOOTSTRAP: A Software process assessment and improvement methodology

This article was published in the International Journal of Software Engineering and Knowledge Engineering in 1995, presenting BOOTSTRAP software process assessment and improvement methodology version 2.3. The article noted that the next steps in methodology development were to enter into collaboration with the European Software Institute (ESI) and to join the international SPICE project[41], as well as including new specific application areas such as embedded systems for methodology use. The article was mainly based on the work done at the end of the BOOTSTRAP project, when the research results were further developed for its deployment[42] in the industry, and BOOTSTRAP methodology version 2.3 was developed. One part of that effort was the establishment of the BOOTSTRAP data base, for which a classification schema for the assessments had to be devised. The purpose was to collect all BOOTSTRAP assessment results into the data base by offering the possibility of benchmarking the results of a single assessment against the data base. At the same time, the data in the data base standardised and validated the methodology contents.

The BOOTSTRAP project was concluded in 1993. Thereafter its members founded BOOTSTRAP Institute[43] to complete the methodology for commercial markets and establish and maintain the BOOTSTRAP data base. The main objectives of this new legal entity were to keep the methodology up to date, spread and support its use, manage automatic data collection and data base services and train the assessors in order to guarantee that all the assessments fulfilled the same quality standards. Within the BOOTSTRAP Institute, version 2.3 of the methodology was developed and published, to include the BOOTSTRAP assessor training and accreditation schema, a methodology licensing policy, questionnaire version 2.3, computer-aided tools for scoring and

[41] The author joined the SPICE project at its beginning in January 1994.
[42] One part of that effort was writing a book: Software Process Assessment and Improvement – The BOOTSTRAP Approach" (Kuvaja *et al.* 1994), which documented the main results of the project for exploitation purposes.
[43] The statute of the BOOTSTRAP Institute was signed on April 4th 1994.

data collection during assessment, results presentation profiles and data transfer for the BOOTSTRAP data base that would enable benchmarking the assessment results against the data base. Templates for on-site assessment meetings and reports were included, as well as a description of the assessment process and guidelines for process improvement and the generation of an improvement action plan. The article focuses particularly on software process improvement, which was explicitly the primary purpose of BOOTSTRAP assessment.

5.4 Software process maturity and certification

This article was published in the Journal of Systems Architecture in 1996. The concepts of process maturity and certification were introduced through representing the main maturity models and certification schemas of that time and their subsequent development potentials in the SPICE project and BOOTSTRAP methodology. The concept of a quality system and its certification was presented according to the ISO 9000 suite of standards, leading on to a discussion of the concept of process maturity by way of a short description of the objectives and main technical characteristics of the Capability Maturity Model (CMM). An introduction to SPICE followed, explaining the background and main objectives of the emerging standard for software process assessment, process improvement and capability determination. The article concluded by announcing the new SPICE conformant version of BOOTSTRAP methodology.

The article considered process maturity and quality certification as a means to improve customer satisfaction in terms of better product and service quality, timeliness of delivery and cost-effective production that results in lower costs. It took the software producer viewpoint, in which improved software quality means better predicted and controlled software project costs, time schedule and results and enhanced ability to manage the risks of development. Process maturity was analysed through the main capability maturity model CMM and quality system certification through the ISO 9000 suite of quality standards. CMM was seen as a de facto international standard for software process assessments, originally intended as “Contractor Software Engineering Capability Assessment (CSECA)” that was used briefly and then replaced by “Software Capability Evaluation (SCE) (Humphrey & Sweet 1987), and has been used in the same sense as ISO 9000 for

quality system certification in cases where the intended target maturity level was assigned[44].

The article introduced the foundation for combining international quality standards and certification with process capability evaluation via the initiative to develop an international standard for software process assessment and improvement. The developed software process assessment standard was intended to be applicable for both software process capability determination and software process improvement. The article discussed further how BOOTSTRAP methodology[45] used the CMM[46] approach for process assessment in combination with ISO 9000 quality requirements for goal-oriented process improvement purposes. This was seen as the main basis for further adoption of the requirements of the forthcoming international standard for software process assessment to be prepared and tested in the SPICE project. Conformance of BOOTSTRAP methodology version 3.0, then under development, with the initial requirements stated in the first results[47] of the SPICE project was also demonstrated. The article concluded by stressing the importance of quality improvement, especially in small to medium-sized enterprises, by means of assessment, certification and the application of international quality and process-related standards.

5.5 BOOTSTRAP 3.0 – A SPICE[48] Conformant Software Process Assessment Methodology

This article was published in Software Quality Journal in 1999. It introduced BOOTSTRAP methodology version 3.0, which was compliant with the definitions and process suite defined in the SPICE project, called SPICE version 1.0. The SPICE result was a basis for preparing the emerging ISO 15504 standard for software process assessment. The core of the new methodology version consisted of an assessment model and method. The assessment model of the

[44] When CMM was used in the USA for evaluating the maturity levels of the subcontractors of the Department of Defense, the target maturity level of the contractor organisation was set at level 3 in bidding situations.
[45] Version 2.3, see (Bicego & Kuvaja 1996: 618).
[46] CMM versions 1.0 and 1.1.
[47] So-called SPICE version 1.0.
[48] Software Process Improvement and Capability dEtermination – SPICE, an international project set up by ISO JTC1, Technical Committee 7 (Software engineering) Working Group 10 (Software process assessment) to develop initial working draft material for the forthcoming standard ISO 15504 "Information technology – Software process assessment".

version 3.0 was updated to align with the ISO 12207 and 15504 reference model requirements. In addition to the process and capability dimensions, it contained a technology dimension based on the earlier BOOTSTRAP questionnaire version 2.3. The process dimension contained 33 processes organised into six clusters: Organisation, Lifecycle-Dependent, Management, Support, Customer–Supplier and Process-Related. The capability dimension consisted of six levels, each comprising one or more process attributes for scoring, adopted from ISO 15504 and included as quartiles in capability and maturity ratings. BOOTSTRAP assessment was conducted at SPU and project levels, as was the earlier BOOTSTRAP methodology version. The BOOTSTRAP Institute had already released the methodology version in 1997 and it was used in SPICE version 1.0 trials as one of the complete assessment methodologies.

5.6 TAPISTRY – A Software Process Improvement Approach Tailored for Small Enterprises

This article was published in *Software Quality Journal* in 1999 and introduced the TAPISTRY approach that resulted from an ESSI ESPRIT project (No. 24238), called TAPISTRY[49], a tutored process improvement approach tailored for small enterprises[50]. The article described how the TAPISTRY approach was developed, used and validated by experiments. In the TAPISTRY project, a downscaled model of the BOOTSTRAP assessment methodology, called BootCheck[51], was adopted and a workshop-based assessment and improvement method developed, together constituting a process improvement approach for small to medium-sized enterprises. In TAPISTRY workshops, the participants were tutored in self-assessment and improvement planning by software process improvement experts. The resulted TAPISTRY approach was validated through the experiments performed during TAPISTRY project.

[49] Tailored Application of Software Process Improvement Techniques for Small Enterprises.
[50] Fewer than 60 staff members.
[51] Developed in cooperation between the BOOTSTRAP Institute and European Software Institute – ESI. Free versions of the tool can be downloaded from the following web addresses: bootstrap-institute.com and esi.es. See also (Doiz 1997).

5.7 Definition of an embedded systems process frame to enhance ISO 15504 conformant assessments

This publication is a conference paper published in the Proceedings of the International Conference on Product Focused Software Process Improvement in 1999. The paper outlined the backgrounds for new features of BOOTSTRAP methodology to be used in embedded systems assessment. It explained how development of products that include embedded systems involve aspects that are not found in traditional software development and how general software process assessment methodologies neglect features specific to embedded systems. The paper proposed a bridge between the development of embedded systems and general software process assessment standards, by defining a process frame for the development of embedded systems and by identifying key activities for each process of the frame. The findings of the paper were based on an analysis of three industrial organisations in the PROFES project[52], in addition to an extensive literature survey. They formed a basis for enhancing ISO 15504 (SPICE) conformant assessment methodologies to cover the assessment of the embedded systems development process.

[52] Esprit project PROFES (EP 23239) funded by the EU.

6 Conclusions

In this chapter, the main results of the research are summarised, validation of the results reported and the limitations and further research possibilities discussed.

6.1 Main contributions

The research reported in this thesis was iterative and conducted as part of international research projects and an institute from 1990 to 2004. The research process was divided into three chronological phases (Tables 2 and 3). In the first phase, the research was carried out in a European R&D project (BOOTSTRAP), the main result of which was a research prototype of the BOOTSTRAP methodology. The main outcomes of the project were reported in the attached paper (Kuvaja & Bicego 1994) (Paper 1), which was published in *Software Quality Journal*. Additional results of the project were published in a book by the author of this thesis and others (Kuvaja *et al.* 1994) and in the first international conference on Lean Software Development[53]. The book describes the project, introduces the initial ideas of the BOOTSTRAP methodology and reports the main experiences of the methodology development. An additional result was the establishment of the BOOTSTRAP Institute as a spin-off from the project.

The second phase of the research was conducted by the BOOTSTRAP Institute internally and by participating in the SPICE project. The contribution contains two main professional versions of the methodology. Based on the first version, the BOOTSTRAP Institute registered BOOTSTRAP as a trademark in Europe and the USA in order to protect the specific intellectual property rights of the methodology in the markets. The results were reported in attached Paper 2 (Simila *et al.* 1995), Paper 3 (Bicego & Kuvaja 1996) and Paper 4 (Kuvaja 1999), which were published in *International Journal of Software and Knowledge Engineering* and *Software Quality Journal*. Additional publications of the research performed during the phase that supported the BOOTSTRAP methodology development were Kuvaja 1995a, Kuvaja *et al.* 1995a, and Messnarz & Kuvaja 1996 and Kuvaja *et al.* 1995b).

[53] Esprit BOOTSTRAP, IPSS–Europe International Conference, Lean Software Development, Commission of European Communities, DG XIII: Telecommunications Information Industries and Innovation, IPSS: Information Processing Systems and Software Sub-programmes, Steinbeis-Zentrum Europäischer Technologietransfer, October 21st–23rd 1992, Stuttgart, Germany.

The third phase of the research was accommodated in two European research projects with a view to enhancing the methodology to support software process improvement in small enterprises and embedded product development. The results were published as an article in *Software Quality Journal*, attached Paper 5 (Kuvaja *et al.* 1999b), and in the proceedings of the first PROFES conference, attached Paper 6 (Kuvaja *et al.* 1999a). Other publications resulting from the same research efforts and contributing equally to methodology enhancements were Taramaa *et al.* 1998, Oivo *et al.* 1999, Kuvaja 1999 and Hamann *et al.* 2000. The research efforts and results contributed iteratively to the research questions set out for this research in section 1.3. Therefore, the specific contributions will be outlined here for each question separately.

> *Research question 1 (Q1): What should a software process assessment methodology include, such that it* ***supports professional software process assessment****?"*

BOOTSTRAP methodology was developed in a series of projects, as described above. Each project contributed by providing answers to research question Q1. The main result of the BOOTSTRAP project was the research prototype of the methodology documented in Paper 1. The description contains overall goals of the methodology and a detailed description of the assessment process, including the roles and activities involved in the steps that constitute it, defines the assessment questionnaire and reference model behind it, explains the scoring principles, outlines an algorithm that is used for rating the evaluations and gives an overview of the presentation of results using the CMM maturity levels with quartiles. The model supports assessment to be performed in both SPU and project levels, each having its own questionnaire with comparable contents. Performing assessment at two levels gives an opportunity to compare the results between the target SPU and its projects. This provides a means for evaluating how far the projects are compliant with the organisation procedures and whether the defined procedures are known and considered applicable and effective in practice. The questions are answered by choosing from a list of adjectives, namely "absent", "weak", "fair", "extensive" and "non-applicable". "Absent" means "no"; "weak", "fair" and "extensive" are three different degrees of a "yes" answer, replacing CMM's binary scoring; and "non-applicable" leaves the question out of rating. The assessment results include maturity presentation as a maturity tree that is read in top-down order. The top of the tree indicates overall maturity, then moving downward and hierarchically, the maturity of the process

areas and sub-process areas, ending with single process capability ratings, in which the capability can span two to three capability levels[54]. Another main presentation form is the capability profile including all assessed processes. All these features are unique but especially notable among them are the new scoring principles and capability profiles, which were adopted by the SPICE work and are now part of international standard ISO 15504. The validation data from 37 assessments[55], including 37 SPUs and 90 projects, and lessons learned are included in the presentation of results.

Answers to research question 1 in the second phase of the research were published in three papers (Papers 2, 3 and 4). The main results of the phase were two commercial versions of the BOOTSTRAP methodology developed under the management and with the support of the BOOTSTRAP Institute. The first commercial version (Paper 2) includes a comprehensive description of the contents of the methodology and three case studies of typical assessment target organisations and their goals for the assessment. In the development of this version, concerns about objectivity, reliability and repeatability, which were received as feedback on the research prototype, were taken into account. The validation of the assessment methodology data contained overall results of 63 assessments and feedback from the assessed organisations. Intermediate results were published (Paper 3) at the moment when the research had already started in collaboration with the SPICE project. The publication explains the contents of the quality standards and their role as the backbone of the reference model of the methodology. It contains also the first outline of the SPICE conformant BOOTSTRAP methodology and its objectives. The second commercial version of the BOOTSTRAP methodology (Paper 4) contains the assessment method, process model, capability levels, scoring, principles of ratings and results presentation and process improvement guidelines. The assessment method and reference model are ISO 15504 conformant. The results include enhancements in the process dimension, with goals, base practices and work product indicators, and in the capability dimension, with management practices, practice

[54] Trillium had the same idea later in version 3.0 (April & Coallier 1995a).

[55] This number compares quite well with, for example, CMM validation. See Humphrey's comments in "A Critical Look": "…the SEI has conducted 16 SEI-assisted assessments and supported 37 self-assessments. Because at least 50 people are involved in each assessment, several thousand US software professionals have been directly involved in this work" (Humphrey & Curtis 1991: 45). The same number of software engineers in Europe can be considered to have been involved in BOOTSTRAP methodology research prototype validation, as the number of assessments includes 37 SPUs and their 90 projects.

performance characteristics and resource and infrastructure characteristics. It includes also a technology dimension as its own third dimension and refines the assessment results by subdividing the levels into quartiles. Validation of the new methodology version was done as a part of the SPICE trials and own internal assessment by the member organisations of the Institute and BOOTSTRAP licensee organisations. The main contribution of the new BOOTSTRAP version was to be ISO 15504 conformant, while still keeping its original and unique features, rather than simply copying the contents of the standard.

In the third phase of the research, the BOOTSTRAP methodology got new special features in order to address requests from the industry. First, the methodology was downscaled for self-assessment and SME purposes. The research effort was performed in the TAPISTRY project. The results (Paper 5) include a description of the BootCheck assessment model (Doiz 1997), which was incorporated into the TAPISTRY approach arising out of a collaboration between the European Software Institute (ESI) and the BOOTSTRAP Institute. BootCheck contains a classification schema, an assessment model and an assessment method that are implemented and supported by a computer-based tool. The schema is used for classifying the assessed organisations for benchmarking purposes. The assessment model contains a process model, capability levels, scoring and principles of ratings and results presentation. The assessment method is a self-assessment, supported by the BootCheck tool. The process model is a downscaled version of the complete BOOTSTRAP 3.0 process model, which includes 19 processes out of the total of 35 processes. The model was validated based on the feedback provided by 61 beta test organisations in Europe and India (Doiz 1997). The process capability dimension is equal to capability levels defined in the original BOOTSTRAP assessment methodology, but contains levels from 0 to 3 only. In TAPISTRY, the BootCheck assessment method was rounded off with a two-day tutored workshop on the step-by-step self-assessment and improvement planning procedure. Workshop participants received tuition in evaluating the software processes of their own organisations using BootCheck self-assessment, analysing the results and developing a process improvement plan specific to their organisation. The workshop was assisted with a portfolio of process improvement templates and a standard workshop procedure of five steps with fixed contents. The TAPISTRY workshop was validated by systematically collecting feedback from the participants during the TAPISTRY project and then when the approach was taken into commercial use. A summary of the feedback is included in the results presentation.

As part of the research performed in the PROFES project, specific requirements for assessing embedded product development were defined (Paper 6). This was an enhancement of the BOOTSTRAP methodology to embedded systems development. The results contained a process frame for the development of embedded systems and identification of the key activities for each process of the frame. The findings were based on an analysis of three industrial partner organisations of the PROFES project. The resulting process frame set out the requirements for defining product development lifecycle processes, including product requirements specification, product design, system design and implementation, systems integration and testing, production and installation, customer support and product improvement. The requirements were implemented in the BOOTSTRAP methodology, where a new process area was added, called "product development lifecycle processes", and existing management processes were updated to include product management processes. The result was BOOTSTRAP methodology version 3.2, which includes features quite similar to those of SE-CMM, while being ISO 15504 conformant at the same time. The version can be kept as a pathfinder for later updates in ISO 15504 to cover product development. Validation of the features was done in the assessments and expert reviews during the PROFES project[56].

> *Research question 2 (Q2): What should a software process assessment methodology include, such that it* ***supports continuous software process improvement****?*

One cornerstone of the BOOTSTRAP methodology is total quality management (TQM) (Paper 1), which takes ideas of continuous improvement, a Plan–Do–Check–Act scheme, a process-oriented approach for the improvement of customer satisfaction and productivity and time-to-market as the basic quality dimensions. Improvement guidelines were devised along the same lines (Paper 1) describing how to define and select gradual changes to improve the software production process and formulate them into action plans. The first phase of the assessment is assessment preparation, in which the people to be interviewed are informed about the assessment, the target of the assessment selected, the assessment team identified and a confidentiality agreement signed. The purpose of the phase, part of a continuous improvement process, is to get the target organisation committed to improvement. After each interview, the process ratings are analysed

[56] See for example in (Taramaa *et al.* 1998: 909).

immediately and verified with the interviewees. Verified data are a basic requirement for developing the improvement plan. The main goal of this phase is to ensure that the people interviewed generally agree on the basic data collected and to check for any incompleteness or inconsistencies in the results. The verified assessment results are presented in the form of maturity trees, capability profiles, strengths and weaknesses profiles and benchmarking against the BOOTSTRAP data base, and evaluated against company goals and business needs. The resulting improvement plan (Paper 2) includes descriptions of the small short- and long-term actions that the target organisation is recommended to take before implementing them as projects. The final assessment results including the improvement plan are then presented to the target organisation with the aim of obtaining commitment for the improvement proposals. This basic support for continuous improvement is aligned with SPICE principles (Paper 3), further updated (Paper 4) to maintain that alignment. The idea of continuous improvement was disseminated via the workshops (Paper 5), in which participants learn by doing an assessment process themselves with the help of professional tutoring. The ideas (Paper 6) were incorporated into a new version of BOOTSTRAP methodology, which became applicable as part of the PROFES product-driven process improvement approach that follows the Quality Improvement Paradigm (QIP) to achieve continuous improvement.

> *Research question 3 (Q3): What should a software process assessment methodology include, such that it* ***supports the requirements of relevant standards and their evolution****?*

The first versions of the BOOTSTRAP assessment methodology including research prototype versions (Paper 1) and the first commercial version (Paper 2 and Paper 3) were developed by applying the original SEI model as the main de facto standard. The application includes two aspects, namely process dimension and capability dimension, both of which are needed to answer research question 3. The process dimension of the SEI model was documented in the form of a maturity evaluation questionnaire. The background standard of the questionnaire was DoD standard number 2167A for software development, dating from 1988. BOOTSTRAP development took that questionnaire as a starting point (Paper 1) and enhanced it with new questions based on the guidelines from ISO 9000 quality standards and ESA (European Space Agency) process model standards. Enhancements were made in order to fit the methodology into the European context. ISO standards were applied in two different ways (Paper 3). First, the

requirements of the quality standard were applied in the quality-related section of the questionnaires. Second, the requirements for a quality system were adopted, with the result that the assessment was performed both on SPU and project levels, as the ISO certification schemes required. Both SPU and project have specific questionnaires with comparable contents, which permit comparison of the maturity and capabilities between the two (Paper 2). Assessment performed in this way is the main advantage of the BOOTSTRAP methodology when compared to most CMM-based assessment methodologies, where the result of assessment of the organisation is derived only from project-level assessments. Thanks to the influence of ESA's classical software lifecycle model on the contents of the lifecycle questions, the questionnaires could be enhanced with support processes. In the development of the ISO 15504 conformant methodology (Paper 4), the BOOTSTRAP methodology development team made the decision to apply ISO 12207 lifecycle standard just after the first SPICE trials and became ISO 15504 conformant before the standardisation working group, SPICE, turned to the same decision.

The original SEI model was also de facto the standard for software process assessment approaches and for the capability dimension. The maturity levels of the SEI model were applied as such in the BOOTSTRAP methodology in the beginning (Paper 1) but were developed later to include quartiles within the levels to get more detailed process capability profiles and maturity levels for SPU and its projects. This was also found to support continuous improvement, as smaller advancements in capability and maturity can be recognised and communicated (Paper 2 and Paper 3). New capability concepts, though aligned with CMM, were developed and trialled in the SPICE project. A brainstorming meeting in the SPICE project, lasting several days and hosted by the Software Engineering Institute in Pittsburgh in 1994, used the "taming the lion" metaphor[57] and ended up with six capability levels, which are now part of the ISO 15504 standard. The current version of BOOTSTRAP methodology (Papers 4, 5 and 6) applies ISO 15504 requirements while keeping its original quartiles within the capability levels and defining maturity for both the SPU and its projects. Also ISO has followed the idea and in 2008 published requirements for determining organisational maturity as part 7 (ISO/IEC 15504-7:2008 2008) of ISO 15504.

[57] The more professional the performer is the more structured and managed is the process of taming. The lions were considered to be as "wild" as the processes without management.

*Research question 4 (Q4): What should a software process assessment methodology include, such that it **supports different assessments settings**?*

The purpose of BOOTSTRAP software process assessment is software process improvement, which starts from the company goals and business needs. This principle was already stated at the beginning of the BOOTSTRAP project (Paper 1) and pertains to all types of BOOTSTRAP assessments (Papers 4, 5 and 6). In addition, BOOTSTRAP assessment can be applied when a company considers, for example, IS0 9001 certification as the first step in software process improvement (Paper 3). The scope of the assessment is addressed in BOOTSTRAP in a unique way by assessing the organisation (SPU) level and project level separately, according to ISO 9001 requirements (Paper 2). Professional BOOTSTRAP assessment prefers that the assessments be performed with assistance from an external assessor, although the purpose is software process improvement (Paper 3). The recommendation is based on experiences from the BOOTSTRAP project, where self-assessments did not have the same effectiveness as assisted assessments and results were less reliable. In second-party assessment, the methodology was seen to be applied more rigorously and without any personal tendency to influence the results. The scope of the assessment can be a full assessment that includes all processes found to exist in the target SPU and its assessed projects (Paper 2 and Paper 4). Assessment can also be focused on certain processes or parts of the SPU and projects. Typically this is done in reassessments for monitoring the improvement progress.

6.2 Validity of the research

The research reported in this thesis contributes to understanding of the phenomenon called software process assessment and improvement, as explained in the previous sections (1.2, 1.3). The contribution development has applied a design science[58] paradigm, together with Deming's cycle and the GQM paradigm. Design science sets the research in a wider perspective and therefore confirms the validity of the research. It also covers the general requirements of internal and external validity, construct validity and reliability set originally by Yin (Yin 2009)

[58] Design science research is a research paradigm in which a designer answers questions relevant to human problems via the creation of innovative artefacts, thereby contributing new knowledge to the body of scientific evidence. The designed artefacts are both useful and fundamental in understanding that problem (Hevner A Chatterjee S 2010: 5).

for case studies in social sciences and recommended to be applied also in empirical software engineering by Wohlin (Wohlin *et al.* 2003). As the design science approach is directly aligned with the research questions of this thesis (see Section 1.3), it fits better for validation of the results here.

According to Hevner, research that follows design science principles brings into sharp focus knowledge and understanding of a design problem and its solution in the building and application of an artefact (Hevner A & Chatterjee S 2010: 5). The artefact constructed in this research is the BOOTSTRAP software process assessment and improvement methodology. The overall design problem was described in section 1.3 and the solution is described in more detail in the research papers represented in chapters 3 to 5.

The paradigm includes seven guidelines on performing design science research, as presented in Table 5.

Table 5. Design science research guidelines (Hevner A & Chatterjee S 2010: 12).

Guideline	Description
1: Design as an *artefact*	Design science research must produce a viable artefact in the form of a construct, a model, a method or an instantiation.
2: Problem relevance	The objective of design science research is to develop technology-based solutions to important and relevant business problems.
3: Design *evaluation*	The utility, quality, and efficacy of a design artefact must be rigorously demonstrated via well-executed evaluation methods.
4: Research contributions	Effective design science research must provide clear and verifiable contributions in the areas of the design artefact, design foundations, and/or design methodologies.
5: Research rigor	Design science research relies upon the application of rigorous methods in both the construction and evaluation of the design artefact.
6: Design as a *search process*	The search for an effective artefact requires utilizing available means to reach desired ends while satisfying laws in the problem environment.
7: Communication of research	Design science research must be presented effectively to both technology-oriented and management-oriented audiences.

These guidelines are used here to demonstrate the validity of the research in a structured manner.

Design as an artefact

As described in chapters 3 and 4, the artefact called BOOTSTRAP assessment and improvement methodology was developed in a series of research projects that

logically formed three phases, each producing a new instantiation or version of the methodology. The instantiations are research prototype, commercial product and specific product versions. Each project had iterative development and research cycles, as described in section 1.4. Within each research cycle, the evolution of the end result starts with a problem statement, is followed by the definition of the design rationale, produces a construct for evaluation, experiments with the construct in practice, evaluates its performance and collects feedback and experiences, has a short corrective action cycle after evaluation and ends with a new instantiation to be communicated to the external knowledge base and immediate use and system (application and problem) environment (Hevner & Chatterjee 2010: 17). The environment in this research specifically contains industrial base line projects involved in the experiments, user communities (e.g. BOOTSTRAP assessors), collaborative development and research projects (e.g. SPICE) or institutes (e.g. ESI) and other external industry interested in the artefact or having similar initiatives, developments or use interests or experiences.

Problem relevance

The initial problem of this research comes from the practical problems of software quality, development productivity and meeting the demands of the marketplace. The overall research problem and the starting point for the research were explained in section 1.3 in order to provide a rationale for the research. Each reported project had its own research problems aligned with the overall problems of this research or parts thereof. Each research cycle within the projects then started with its own specific or more detailed problems that set the goal for the cycle, as described in section 1.4.

Design evaluation

Each research cycle concluded with an experiment in a real industrial environment and immediate feedback from practitioners and the developers was collected and analysed. This and the problem setting follow the idea of the relevance cycle defined by Hevner (Hevner & Chatterjee 2010: 16–17) for design science research. Based on the analysis, corrective actions were devised and carried out. Problems that could not be corrected immediately were fed into the second research and development cycle. They might even initiate a new research project and be communicated, with the experiences, to the external knowledge

base and the immediate use and system environment. It is remarkable that in this research both methodology developers and practitioners from the industry worked in close cooperation and the experiments were done in practical situations in industry. This made it possible to achieve close cooperation and get rich feedback immediately. Based on the feedback, the BOOTSTRAP methodology was given new features (e.g. process capability), which are now part of an international standard (ISO 15504).

Research contributions

Each new version of the artefact that was a result of a research phase and that was communicated to the external knowledge base and immediate use and system environment was subjected to verification in a practical environment. The research prototype version of the BOOTSTRAP methodology at the end of the BOOTSTRAP project had been used in 37 assessments including 37 SPUs and 90 projects, as reported in Paper 1. This number is comparable, for example, with CMM validation such as reported by Humphrey in "A Critical Look" as follows: "…the SEI has conducted 16 SEI-assisted assessments and supported 37 self-assessments. Because at least 50 people are involved in each assessment, several thousand US software professionals have been directly involved in this work" (Humphrey & Curtis 1991: 45). About the same number of software engineers can be considered to have been involved in the BOOTSTRAP research prototype validation. In addition, the first reported assessment results of the use of the first commercial version of the methodology included data from 63 assessments (Paper 2). The further versions of the methodology were validated as part of international SPICE trials and internally in TAPISTRY and PROFES projects, as described in the previous section.

Research rigour

As described in section 1.4 the research and development of the artefact in this research happened iteratively by applying Deming's cycle. The final artefact is the result of an evolution including different versions of the BOOTSTRAP methodology. Each version was extensively verified in practical situations and can be used for software process assessment and improvement when the context suits the version purpose. For example, TAPISTRY is a very good means of

introducing software process improvement for the first time to practitioners or students in higher software engineering education.

Design as a search process

The research was conducted in numerous different research and development projects, involving many practitioners and researchers with high expertise in specific topics of the research area. The research and development started in the BOOTSTRAP project in the European context, and with that expertise entered into a worldwide research and development initiative, the SPICE project. The final phase of the research then focused on special purpose assessments requiring special expertise in the focused area in the TAPISTRY and PROFES projects, as described in chapters 3 and 4.

Communication of the research

The BOOTSTRAP project has published actively in the international and European contexts, for both academia and industry. At the end of the project, a spin-off called the BOOTSTRAP Institute was established, a book describing the methodology published and its print-run of 2,000 copies sold. Additionally, the first international conference on Lean Software Development was held, as described above. The BOOTSTRAP Institute has actively participated in European research projects and the SPICE project, which in turn has made a substantial contribution to knowledge. The SPICE project has also started to organise an international conference on software process assessment (SPICE Conference).

6.3 Limitations of the study

The main limitation of this thesis is that it was written during a long time period, starting in 2001 and finishing 11 years later. This required that the evolution of the knowledge base, including background models and standards, and developments in the immediate use and system environment be taken into account and relate to the results of the study. The immediate knowledge base and environment have evolved greatly since 2000 and the results of the research have been applied in industry in practical assessments. The impact of the results can be recognised in the contents and development of international standards, including

the last updates to ISO/IEC 15504-6:2008 (2008), ISO/IEC 15504-7:2008 (2008) and ISO/IEC 15504-9:2011 (2011). The results of the study have been cited copiously in international research and have had a significant impact on the evolution of the external knowledge base. Solving the problems of evolution was the motivation for the thesis work to illuminate the unpublished ideas and rationales behind the publications. Chapter 2 describes the course of evolution of the external knowledge base and the immediate use and system environment and sheds light on the developments behind the methodology, which have never been published before. The author of the thesis was one of the key developers of BOOTSTRAP methodology development in all phases of the research and has had access to the authentic results produced. Therefore, the thesis exhibits precise knowledge of the way in which BOOTSTRAP methodology emerged and its relation to the evolution of the general area. BOOTSTRAP methodology is still in use and has good potential to be applied in improving software, systems and services development in the future. The thesis provides a starting point to continue into that direction.

Other limitations of the thesis are inherent in the area of software process assessment improvement itself. Critics have raised their own concerns about this field, for example Bollinger and McCowan (Bollinger & McGowan 1991), Gray and Smith (Gray & Smith 1998) and Dekleva and Drehmer (Dekleva & Drehmer 1997). Bollinger and McCowan directed their criticism specifically towards the maturity level definition of CMM. Others have criticisms of a more general nature, relating to all assessment and improvement approaches. Dekleva and Drehmer have gone deeper into the topic; their statistical analysis of the contents of the CMM questionnaires, using Rasch theory, raised concerns similar to those of Bollinger and McCowan. Gray and Smith claim that, although the ideal aim of software process assessment and improvement is to furnish software developers with a means to rate their software development and thereby empower them to acquire a credible recognition of the degree of goodness of their practices and hence continuously to improve the practices, the truth is different. According to them, the assessment leads only to identification of those processes that need to be improved. This result is then fed into the process improvement plans, which they say do not necessarily guarantee any improvements to the work of practical software engineers, as was the original goal.

6.4 Future research

The results of this thesis offer a sound basis for further study, including detailed results of analysis and experiences of software process assessment and improvement approach development and its relation to the current state of the art. In specific the actual improvements gained using the assessment approaches would be worth to be investigated in longitudinal quantitative studies. Some unsolved problems also remain, suggesting possibilities for future research, notably relating to the new developments in software engineering. The current software process assessments are based on the requirements of international standard ISO 15504, which specify that software process assessment should be carried out by comparing the performance of the target organisation against the suite of international standards for software, system and service development lifecycles. It also includes comprehensive guidance on doing the assessments. Nevertheless, the standards have not confronted and therefore do not encompass the rapid development of the area, including for example agile and lean thinking. Some assessment approaches have been proposed by Sidky for agile process assessment (Sidky *et al.* 2007; Smith & Sidky 2009), but they have the same limitations as the ISO 15504 conformant assessment approaches. The concept of capability, despite its long history, needs to be rethought, as it is now based on adaptation of management practices (Gray & Smith 1998), which do not apply in the agile way of working. The contents of the maturity levels should also be rethought. The next step following agile has been quite widely accepted to be the adoption of lean principles, which operates at organisational level, whereas agile remains at project level. This fits well with the BOOTSTRAP assessment principle of performing it at both organisational and project level. The thesis provides an excellent opportunity to follow the research line started by SEI and apply the assessment in an agile and lean environment. The first efforts have already started in the Finnish Cloud programme for the development of agile and lean transformation research (Mandic *et al.* 2010).

References

Allen RH & Sriram RD (2000) The role of standards in innovation, Technological Forecasting and Social Change 64(2): 171–181.

April A & Coallier F (1995a) Trillium V3.0: A Model for the Assessment of Telecom Software System Development Capability, 2nd International SPICE Symposium ASQRI. : 79–88.

April A & Coallier F (1995b) Trillium: a model for the assessment of telecom software system development and maintenance capability. Proceedings of the Second IEEE International Software Engineering Standards Symposium 1995 - ISESS'95, 'Experience and Practice', IEEE: 175–183.

AQAP-13:1981 (1981) AQAP-13, NATO Software Quality Control System Requirements. AQAP-13:1981.

Arnold S & Lawson HW (2004) Viewing systems from a business management perspective: The ISO/IEC 15288 standard, Systems engineering 7(3): 229–242.

Austin RD (1996) Measuring and managing performance in organizations, Dorset House Publishing, New York.

Barford A, May G, Miller C, Ould M, Tait S, Thomas M & Watss J (1992) Quantum A measurement-based framework for the assurance of software quality.

Basili V & Green S (1994) Software process evolution at the SEL, IEEE Software 11(4): 58–66.

Basili VR (1992) Software modeling and measurement: the Goal/Question/Metric paradigm, Computer Science, Research Works (CS-TR-2956, UMIACS-TR-92-96).

Basili VR & Weiss DM (1984) A methodology for collecting valid software engineering data, IEEE Transactions on Software Engineering (6): 728–738.

Bate R, Kuhn D, Wells C, Armitage J & Clark G (1995) A systems engineering capability maturity model, Version 1.1, CMU/SEI-95-MM-003.

Bicego A & Kuvaja P (1996) Software process maturity and certification, Journal of Systems Architecture 42(8): 611–620.

Bicego A, Kuvaja P, Van Latum F, Oivo M, Rodenbach E & Ruhe G (1997) PROFES: Announcing the Marriage Between Process Assessment and Measurement, International Conference of Software Engineering - ICSE'97, Poster.

Bollinger TB & McGowan C (1991) A critical look at software capability evaluations, IEEE Software 8(4): 25–41.

Brooks Jr FP (1987) No silver bullet: essence and accidents of software engineering, Computer 20(4): 10–19.

Bush M (1990) Improving software quality: The use of formal inspections at the Jet Propulsion Laboratory. The 12th International Conference on Software Engineering - ICSE´90, IEEE: 196–199.

Bush MW (1991) Process assessments in NASA. Proceedings of the 13th international conference on Software engineering - ICSE´91, IEEE Computer Society Press: 299–304.

Chrissis M, Konrad M & Shrum S (2007) CMMI® Guidelines for Process Integration and Product Development, Boston, USA, Addison-Wesley.
Chrissis MB, Konrad M & Shrum S (2011) CMMI® for Development: Guidelines for Process Integration and Product Improvement, Addison-Wesley Professional.
CMMI PT (2002a) Capability Maturity Model Integration, Version 1.1, CMMI for Systems Engineering and Software Engineering (CMMI-SE/SW, V1.1), Continuous Representation, CMU/SEI-2002-TR-001, ESC-TR-2002–001.
CMMI PT (2002b) Capability Maturity Model Integration, Version 1.1 - CMMI for Systems Engineering and Software Engineering (CMMI-SE/SW, V1.1), Staged Representation, CMU/SEI-2002-TR-00, ESC-TR-2002-002.
CMMI PT (2007) CMMI for acquisition, version 1.2, CMU/SEI-2007-TR-017.
CMMI PT (2009) CMMI for services, version 1.2, CMU/SEI-2009-TR-001.
Coallier F, Gammage N & Graydon A (1994) TRILLIUM-Telecom Software Product Development Capability Assessment Model, Bell Canada.
Coallier F, Gammage N & Graydon A (1993) Trillium-Software Process Self-assessment Capability Assessment (4.0), Bell Canada, Bell Northern Research, Northern Telecom PI QOOS PI QOOOS(4.0).
Compita (1993a) Software Development Diagnostic Practitioner's Guideline, Version 2.5, Compita.
Compita (1993b) Software Technology Diagnostic Preparation Guideline, Version 2.5, Compita.
Craigmyle M & Fletcher I (1993) Improving IT effectiveness through software process assessment, Software Quality Journal 2(4): 257–264.
Curtis B, Hefley WE & Miller S (1995) Overview of the People Capability Maturity Model, Overview of the People Capability Maturity Model, CMU/SEI-95-MM-01.
Cusumano MA (1991) Japan's Software Factories: A Challenge to US Management , Oxford University Press, USA.
Damele G, Bazzana G & Maiocchi M (1995) Quantifying the benefits of software process improvement in Italtel Linea UT exhange, ISS Conference.
Dekleva S & Drehmer D (1997) Measuring software engineering evolution: A Rasch calibration. Information Systems Research 8(1): 95–104.
Deming WE (1981) On the management of statistical techniques for quality and productivity , W. Edwards Deming.
Deming WE (1982) Quality, productivity, and competitive position, Massachusetts Institute of Technology, Center for Advanced Engineering Study Cambridge, MA, USA.
Deming WE (1986) Out of the Crisis, USA, MIT press.
Dertouzos M, Lester RK & Solow R (1989) Made in America: Regaining the competitive edge, Cambridge: MIT Press 81: 93.
DOD-STD-2167A:1988 (1988) DOD-STD-2167A, Defence System Software Development, DOD-STD-2167A.
DOD-STD-2168:1988 (1988) DOD-STD-2168:1988, Defence system Software Quality Program DOD-STD-2168:1988.

DOD-STD-7935A:1988 (1988) DOD-STD-7935A,DOD Automated Information Systems (AIS) documentation standards,. DOD-STD-7935A.
Doiz I (1997) ESI news, Software Process- Improvement and Practice 3(1): 62–63.
Dorling A (1993) SPICE: Software process improvement and capability determination, Software Quality Journal 2(4): 209–224.
ECSS-E-40A:1996 (1996) Space engineering, Software, ECSS-E-40A.
ECSS-E-40B-1:2003 (2003) Space engineering, Software - Part 1B: Principles and requirements, ECSS-E-40B -Part 1.
ECSS-E-40B-2:2005 (2005) Space engineering, Software - Part2B: Document requirements definitions (DRDs), ECSS-E-40B-Part 2.
ECSS-E-40C:2009 (2009) Space engineering, Software, ECSS-E-40C.
ECSS-Q-ST-80C:2009 (2009) Space product assurance - Software product assurance, ECSS-Q-ST-80C.
ESA 1991 (February 1991) ESA -Software Engineering Standards ESA PSS-05-0, Issue 2, Paris, France, European Space Agency.
ESA.BSSC(96)-1:1996 (1996) ESA.BSSC(96)-1 Issue 1, A Guide to applying the ESA software engineering standards to small software projects, ESA.BSSC(96), Issue1.
ESA.BSSC1-1:1984 (1984) ESA.BSSC-1-1 Issue 1, ESA software engineering standards Issue 1, ESA.BSSC1-1, Issue1.
ESA.PSS-05-0:1987 (1987) ESA.PSS-05-0 Issue 2, ESA software engineering standards Issue 1, ESA.PSS-05-0, Issue1.
ESA.PSS-05-0:1991 (1991) ESA.PSS-05-0 Issue 2, ESA software engineering standards Issue 2, ESA.PSS-05-0, Issue2.
ESSI (1998) ESSI Training Action, 24238 TAPISTRY - Tailored Application of Software Process Improvement Techniques for Small Enterprises.
Fenton NE (1991) Software Metrics - A Rigorous Approach, London, Chapman & Hall.
Forrester E, Buteau B & Shrum S (2011) CMMI® for Services: Guidelines for Superior Service, Addison-Wesley Professional.
Gallagher BP, Phillips M, Richter KJ & Shrum SL (2011) CMMI for Acquisition: Guidelines for Improving the Acquisition of Products and Services, Addison-Wesley Professional.
Glazer H, Dalton J, Anderson D, Konrad M & Shrum S (2008) CMMI or Agile: Why not Embrace Both! CMU/SEI-2008-TN-003.
Gray E & Smith W (1998) On the limitations of software process assessment and the recognition of a required re-orientation for global process improvement, Software Quality Journal 7(1): 21–34.
Haase V, Messnarz R, Koch G, Kugler HJ & Decrinis P (1994) Bootstrap: fine-tuning process assessment, IEEE Software 11(4): 25–35.
Hamann D, Derks P & Kuvaja P (2000) Using ISO 15504 Compliant Assessment Combined with Goal-oriented Measurement for Process Improvement at Dräger Medical Technology, Proceedings of International SPICE Conference, Limerick, Ireland.

Hefner R (1997) Lessons learned with the systems security engineering capability maturity model, Proceedings of the 19th international conference on Software engineering , ACM: 566–567.
Hevner A & Chatterjee S (2010) Design science research in information systems - Theory and practice, New York, USA, Springer.
Humphrey WS (1988) Characterizing the software process: a maturity framework, IEEE Software 5(2): 73–79.
Humphrey WS (1989) Managing the software process, Boston, MA, USA, Addison-Wesley Longman Publishing Co., Inc.
Humphrey WS (1993) Introduction to software process improvement, Research Showcase, Carnegie Mellon University, Software Engineering Institute, Paper 180.
Humphrey WS & Curtis B (1991) Comments on a critical look'[software capability evaluations], IEEE Software 8(4): 42–46.
Humphrey WS, Snyder TR & Willis RR (1991) Software process improvement at Hughes Aircraft, IEEE Software 8(4): 11–23.
Humphrey WS & Sweet WL (1987) A Method for Assessing the Software Engineering Capability of Contractors, Software Engineering Institute CMU/SEI-87-TR-23, ADA187320.
ISO 2009 (2009) Selection and use of the ISO 9000 family of standards, ISO.
ISO 9000:1987 (1987) Quality management systems - Fundamentals and vocabulary, ISO 9000:1987 - First edition.
ISO 9000:1992 (1992) International Quality Standards: An Overview for American Business, UK, Commerce Clearing House.
ISO 9000:2005 (2005) Quality management systems - Fundamentals and vocabulary, ISO 9000:2005, Third edition.
ISO 9000–3:1991 (1991) Quality management and quality assurance standards -- Part 3: Guidelines for the application of ISO 9001 to the development, supply and maintenance of software.
ISO 9000-3:1994 (1994) Quality management and quality assurance standards -- Part 3: Guidelines for the application of ISO 9001 to the development, supply and maintenance of software.
ISO 9000-3:1997 (1997) Quality management and quality assurance standards -- Part 3: Guidelines for the application of ISO 9001:1994 to the development, supply and maintenance of software.
ISO 9001:1987 (1987) ISO 9001 - Model for quality assurance in design, development, production, installation, and servicing. ISO 9001:1987.
ISO 9001:1994 (1994) ISO 9001 - Model for quality assurance in design, development, production, installation, and servicing. ISO 9001:1994.
ISO 9002:1987 (1987) ISO 9002 - Model for quality assurance in production, installation, and servicing. ISO 9002:1987.
ISO 9002:1994 (1994) ISO 9002 - Model for quality assurance in production, installation, and servicing. ISO 9002:1994.

ISO 9003:1987 (1987) ISO 9003 - Model for quality assurance in final inspection and test. ISO 9003:1987.
ISO 9003:1994 (1994) ISO 9003 - Model for quality assurance in final inspection and test. ISO 9003:1994.
ISO/IEC:1991 (1991) ISO/IEC JTC1/SC7 N872, Proposal for a Study Period on Process Management, N872.
ISO/IEC:1992 (1992) ISO/IEC JTC1/SC7 N944R: The Need and Requirements for a Software Process Assessment Standard, Study report, Issue 2.0., N944R, Issue 2.0.
ISO/IEC 12207:1995(E) (1995) ISO/IEC 12207, Information technology - Software life cycle processes. ISO/IEC 12207, First edition.
ISO/IEC 12207:2008(E) (2008) ISO/IEC 12207, Information technology - Software life cycle processes. ISO/IEC 12207, Second edition.
ISO/IEC 15288:2002 (2002) ISO/IEC JTC1/SC7 WG7 NO617, Systems Engineering - Guide for ISO/IEC 15288 (System Life Cycle Processes), ISO/IEC PDTR 19760.
ISO/IEC 15288:2002(E) (2002) Systems and software engineering - System Life Cycle Processes, First edition. ISO/IEC 15288:2002(E).
ISO/IEC 15288:2008(E) (2008) Systems and Software Engineering - System Life Cycle Processes, Second edition. ISO/IEC 15288:2008(E).
ISO/IEC 15504-1:1998(E) (1998) Information technology - Software assessment - Part 1: Concepts and introductory guide, WG10N221.
ISO/IEC 15504-1:2004 (2004) Information technology - Process assessment - Part 1: Concepts and vocabulary. ISO/IEC 15504-1:2004, First edition.
ISO/IEC 15504-2:1998(E) (1998) Information technology - Software assessment - Part 2: A reference model for processes and process capability, WG10N222.
ISO/IEC 15504-2:2003 (2003) Information technology - Process assessment - Part 2: Performing an assessment. ISO/IEC 15504-2:2003, First edition.
ISO/IEC 15504-3:1998(E) (1998) Information technology - Software assessment - Part 3: Performing an assessment, WG10N223.
ISO/IEC 15504-3:2004 (2004) Information technology - Process assessment - Part 3: Guidance on performing an assessment. ISO/IEC 15504-3:2004, First edition.
ISO/IEC 15504-4:1998(E) (1998) Information technology - Software assessment - Part 4: Guide to performing assessments, WG10N224.
ISO/IEC 15504-4:2004 (2004) Information technology - Process assessment - Part 4: Guidance on use for process improvement and process capability determination, ISO/IEC 15504-4:2004, First edition.
ISO/IEC 15504-5:1998(E) (1998) Information technology - Software assessment - Part 5: An assessment model and indicator guidance, WG10N195.
ISO/IEC 15504-5:2006 (2006) Information technology - Process assessment - Part 5: An exemplar Process assessment model, ISO/IEC 15504-5:2006, First edition.
ISO/IEC 15504-6:1998(E) (1998) Information technology - Software assessment - Part 6: Guide to competency assessors, WG10N226.

ISO/IEC 15504-6:2008 (2008) Information technology - Process assessment - Part 6: An exemplar system life cycle process assessment model. ISO/IEC 15504-6:2008, First edition.
ISO/IEC 15504-7:1998(E) (1998) Information technology - Software assessment - Part 7: Guide for use in process improvement, WG10N227.
ISO/IEC 15504-7:2008 (2008) Information technology - Process assessment - Part 7: Assessment of organisational maturity, ISO/IEC 15504-7:2008, First edition.
ISO/IEC 15504-8:1998(E) (1998) Information Technology - Software assessment - Part 8: Guide for use in determining supplier process capability, WG10N228.
ISO/IEC 15504-9:1998(E) (1998) Information technology - Software assessment - Part 9: Vocabulary, WG10N229.
ISO/IEC 15504-9:2011 (2011) Information technology - Process assessment - Part 9: Target process profiles, ISO/IEC 15504-9:2011, First edition.
ISO/IEC 19011:2011(E) (2011) Guidelines for auditing management systems, ISO/IEC 19011:2011(E), Second edition.
ISO/IEC 90003:2004(E) (2004) Software engineering - Guidelines for the application of ISO 9001:2000 to computer software, ISO/IEC 90093:2004(E).
ISO/IEC 9001:2000(E) (2000) Quality management systems - Requirements, ISO/IEC 9001:2000(E).
ISO/IEC 9001:2008(E) (2008) Quality management systems - Requirements, ISO/IEC 9001:2008(E), Fourth edition, Corrected version 2009-07-15.
ISO/IEC 9004:2009(E) (2009) Managing for the sustained success of an organisation - A quality management approach, ISO/IEC 9004:2009(E), Third edition.
ISO/IEC PDTR 15504-8:2012 (2012 (target)) Information technology - Process assessment - Part 8: An exemplar process assessment model for IT service management, ISO/IEC PDTR 15504-8 (under development).
ISO/IEC-JTC1/SC7/WG7/SG1 (1992) The Need and Requirements for a Software Process Assessment Standard, Study Report, N944R, Issue 2.0.
Järvinen P (1999) On research methods, Finland, Opinpaja.
Jones M, Gomez E, Mantineo A & Mortensen U (2002) Introducing ECSS Software-Engineering Standards within ESA, ESA bulletin: 132–139.
Jones M, Mazza C, Fairclough J, Melton B, de Pablo D, Scheffer A, Stevens R & Alvisi G (1996) Software engineering guides, Prentice Hall International (UK) Ltd.
Jones M, Mortensen U & Fairclough J (1997) The ESA software engineering standards: Past, present and future, Third IEEE International Software Engineering Standards Symposium and Forum 1997 - ISESS 97 'Emerging International Standards', IEEE:119–126.
Jung HW, Hunter R, Goldenson DR & El-Emam K (2001) Findings from Phase 2 of the SPICE Trials. Software Process: Improvement and Practice 6(4): 205–242.
Juran JM (1986a) Planning for quality, Juran Institute.
Juran JM (1986b) Universal Approach to Managing for Quality: The Quality Trilogy, Quality Progress, August 1986: 19–24.

Juran JM (1992) Juran on quality by design: the new steps for planning quality into goods and services, Free Press.

Kellner MI (1989) Software process modeling: value and experience, Carnegie Mellon University, Software Engineering Institute.

Kitson DH & Masters SM (1993) An analysis of SEI software process assessment results: 1987--1991, Proceedings of the 15th international conference on Software Engineering - ICSE '93, Los Alamitos, CA, USA, IEEE Computer Society Press: 68–77.

Kitson DH & Masters SM (1992) An Analysis of SEI Software Process Assessment Results: 1987 - 1991, CMU/SEI-92-TR-24, ESC-TR-92-024.

Koch G (1992) The BOOTSTRAP Initiative: Reported Benefits for the Industry, Proceedings of the Esprit BOOTSTRAP Conference on Lean Software Development, Stuttgart, Germany.

Koch G (1993) Process assessment: the 'BOOTSTRAP' approach, Information and Software Technology 35(6): 387–403.

Kugler H- & Messnarz R (1994) From the software process to software quality: BOOTSTRAP and ISO 9000. Proceedings of the First Asia-Pacific Software Engineering Conference, 1994: 174–182.

Kuvaja P (1992) Effects of case implementation on the productivity and maturity level of software development--a case study in a large industrial company, Annual Review in Automatic Programming 16: 161–170.

Kuvaja P (1995a) BOOTSTRAP: A software process assessment and improvement methodology, Objective Software Quality: 31–48.

Kuvaja P (1995b) BOOTSTRAP: European service for assessment and improvement of software development organisations. In Anonymous Milano, Italia, A.I.C.A Associazione Italiana per l'Informatica ed il Calcolo Automatico Sezione di Milano: 21–33.

Kuvaja P (1999) BOOTSTRAP 3.0—A SPICE Conformant Software Process Assessment Methodology, Software Quality Journal 8(1): 7–19.

Kuvaja P & Bicego A (1993) BOOTSTRAP: Europe's assessment method, IEEE Software 10(3): 93–95.

Kuvaja P & Bicego A (1994) BOOTSTRAP - A European assessment methodology, Software Quality Journal 3(3): 117–127.

Kuvaja P, Bicego A & Dorling A (1995a) SPICE: The Software Process Assessment Model, ESI-ISCN´95 - Practical Improvement of Software Process and Products, Volume 1: 87–100.

Kuvaja P, Bicego A & Jansen P (1995b) Overview of SPICE Process Improvement, Proceedings of Software Quality Management, South Africa: Section 13.1.

Kuvaja P & Koch G (1992) Maturity of maintenance [software], Proceedings of International Conference on Software Maintenance 1992, IEEE: 259–260.

Kuvaja P, Maansaari J, Seppänen V & Taramaa J (1999a) Specific requirements for assessing embedded product development, International Conference on Product Focused Software Process Improvement: 68–85.

Kuvaja P, Palo J & Bicego A (1999b) TAPISTRY—A Software Process Improvement Approach Tailored for Small Enterprises, Software Quality Journal 8(2): 149–156.
Kuvaja P, Similä J, Krzanik L, Bicego A, Saukkonen S & Koch G (1994) Software process assessment and improvement: the BOOTSTRAP approach, Blackwell, Oxford.
Kuvaja P (1999) BOOTSTRAP 3.0 A SPICE Conformant Software Process Assessment Methodology, Software Quality Journal 8(1): 7–19.
Larman C & Basili VR (2003) Iterative and incremental developments. a brief history, Computer 36(6): 47–56.
Mackie C & Rigby P (1993) Practical experience in assessing the health of the software process, Software Quality Journal 2(4): 265–275.
Mandic V, Oivo M, Rodrıguez P, Kuvaja P, Kaikkonen H & Turhan B (2010) What is flowing in lean software development, Proceedings of the 1st International Conference on Lean Enterprise Software and Systems (LESS 2010). 65: 72–84.
March ST & Smith GF (1995) Design and natural science research on information technology. Decision Support Systems 15(4): 251–266.
Maupetit C, Kuvaja P, Palo J, Belli M & Isokaanta M (1995) The DriveSPI project - a risk-driven approach for software process improvement, Proceedings of 8th International Conference on Software Engineering and its Applications.
Mazza F (1994) Software engineering standards, Prentice-Hall, Inc.
McFeeley B (1996) IDEAL: A User's Guide for Software Process Improvement, CMU/SEI-96-HB-001.
Merrill DC, Reiser BJ, Merrill SK & Landes S (1995) Tutoring: Guided learning by doing, Cognition and Instruction 13(3): 315–372.
Messnarz R (1994) Design of a Quantitative Quality Evaluation System, Doctoral thesis, Technical University of Graz, Austria.
Messnarz R & Kuvaja P (1996) Practical experience with the establishment of improvement plans, Proceedings of the ISCN'96/SP'96 Congress on SPI, December 1996, Brighton, UK.
MIL-STD-498:1994 (1994) MIL-STD-498, Military standard - Software Development and Documentation, MIL-STD-498:1994.
Moran TP & Carrol JM (1996) Overview of Design Rationale,. In Moran TP & Carrol JM (eds) Design Rationale: Concepts, Techniques, and Use, USA, Lawrence Erlbaum Associates: 1–19.
Norris M, Rigby P & Stockman S (1994) Life after ISO 9001: British Telecom's approach to software quality, Communications Magazine, IEEE 32(10): 58–63.
Northcutt DDM & Paulk MC (2010) Statistical Sampling for Process Assessments, Practical Statistical Process Control for Software Metrics 9(4): 19–28.
Oivo M, Birk A, Komi-Sirviö S, Kuvaja P & Van Solingen R (1999) Establishing product process dependencies in SPI. Proceedings of the Fourth Annual European Software Engineering Process Group Conference, European SEPG´99, Amsterdam, The Netherlands.
Olson TG, Humphrey WS & Kitson D (1989) Conducting SEI-assisted software process assessments, Carnegie Mellon University, Software Engineering Institute, Paper 107.

Ould MA (1992) Software quality improvement through process assessment-a view from the UK, Software Quality Improvement Through Process Assessment, IEEE Colloquium on : 2/1–2/8.

Paulish D (1993) Case studies of software process improvement methods, CMI/SEI-93-TR-26.

Paulk M (1993) Capability maturity model for software, Wiley Online Library.

Paulk M (1997) Software Capability Maturity Model, Version 2, Draft Technical Report, Software Engineering Institute, Carnegie Mellon University, Pittsburgh, Pa.

Paulk MC (1996) Process improvement and organizational capability: Generalizing the CMM, Proceedings of the ASQC's 50th Annual Quality Congress and Exposition, Chicago, IL: 92–97.

Paulk MC (1999) Analyzing the Conceptual Relationship Between ISO/IEC 15504 (Software Process Assessment) and the Capability Maturity Model for Software, Proceedings of the Ninth International Conference on Software Quality, Cambridge, MA, 4–6 Oct 1999, pp. 293–303.

Paulk MC (2004) Surviving the quagmire of process models, integrated models, and standards, Proceedings of the ASQ Annual Quality Congress, Toronto, 24–27 May 2004.

Paulk MC (2009) A History of the Capability Maturity Model for Software, ASQ Software Quality Professional 12: 5–19.

Paulk MC, Curtis B & Chrissis MB (1991) Capability maturity model for software, Carnegie Mellon University, Software Engineering Institute, CMU/SEI-91-TR-24, August 1991.

Paulk MC, Curtis B, Chrissis MB & Weber C (1993) Capability Maturity Model for Software, version 1.1. Capability Maturity Model for Software, Version 1, SEI CMU/SEI-93-TR-24.

Paulk MC & Konrad MD (1994a) ISO seeks to harmonize numerous global efforts in software process management, IEEE Computer 27(4): 68–70.

Paulk MC & Konrad MD (1994b) An overview of ISO's SPICE Project, American Programmer (February): 16–20.

Paulk MC, Konrad MD & Garcia SM (1995) CMM versus SPICE architectures, IEEE Computer Society Technical Council on Software Engineering, Software Process Newsletter (3): 7–11.

Paulk M, Weber C, Garcia S, Chrissis M & Bush M (1993) Key Practices of the Capability Maturity Model for Software, version 1.1, Software Engineering Institute Technical Report CMU/SEI-93-TR-25 ESC-TR-93-178.

Paulk MC, Weber CV, Curtis B & Chrissis MB (1995) The capability maturity model: guidelines for improving the software process. Boston, MA, USA, Addison-Wesley Longman Publishing Co., Inc.

Phillips M (2010) CMMI Representations: Past and Future, Carnegie Mellon University, Software Engineering Institute.

PROFES project team (2000) PROFES - User manual. Stuttgart, Fraunhofer IRB Verlag.

Rombach HD & Verlage M (1993) How to assess a software process modeling formalism from a project member's point of view, Second International Conference on Continuous Software Process Improvement, 1993: 147–158.

Rout TP, El Emam K, Fusani M, Goldenson D & Jung HW (2007) SPICE in retrospect: Developing a standard for process assessment, Journal of Systems and Software 80(9): 1483–1493.

Royce WW (1970) Managing the Development of Large Software Systems, IEEE WESCON: 328–338.

Sanders M & SPIRE Partners (1998) The SPIRE Handbook: Better Faster Cheaper Software Development in Small Organisations, Centre for Software Engineering Limited, Ireland.

Sheard SA (1997) The frameworks quagmire, a brief look. People 829(830): 1012–1016.

Sheard SA & Lake JG (1998) Systems engineering standards and models compared. Proceedings of the Eighth International Symposium on Systems Engineering, Vancouver, Canada, Citeseer: 589–605.

Sidky A, Arthur J & Bohner S (2007) A disciplined approach to adopting agile practices: the agile adoption framework. Innovations in systems and software engineering 3(3): 203–216.

Simila J, Kuvaja P & Krzanik L (1995) BOOTSTRAP: a software process assessment and improvement methodology,. International Journal of Software and Knowledge Engineering 5(4): 559–584.

Smith G & Sidky A (2009) Becoming Agile... In an Imperfect World , Manning Publications.

Taramaa J, Khurana M, Kuvaja P, Lehtonen J, Oivo M & Seppanen V (1998) Product-based software process improvement for embedded systems, Proceedings of 24th Euromicro Conference, 1998, IEEE, Volume 2: 905–912.

Thomson HE & Mayhew P (1994a) A practical approach for software process improvement, Software Quality Management, edited by Ross M.et al., CMP 1: 149–164.

Thomson HE & Mayhew P (1994b) The software process: a perspective on improvement. The Computer Journal 37(8): 683–690.

TickIT (1995) Guide to Software Quality Management System Construction and Certification Using ISO 9001, Issue 3.0, DTI, CSA, UK

TickIT A (1992) Guide to software quality management system construction and certification using EN29001, DTI, CSA, UK.

Tully C, Kuvaja P & Messnarz R (1999) Software process analysis and improvement: a catalogue and comparison of models. In Messnarz R & Tully C (eds) Better Software Practice for Business Benefit - Principles and Experience. Los Alamitos, California, IEEE, Computer Society: 51–106.

Weber CV, Paulk MC, Wise CJ & Withey JV (1991) Key Practices of the Capability Maturity Model. Key practices of the capability maturity model CMU/SEI-91-TR-25, ESD-TR-91-25.

Wohlin C, Höst M & Henningsson K (2003) Empirical Research Methods in Software Engineering, Empirical methods and studies in software engineering: experiences from ESERNET 2765: 7.

Womack JP, Jones DT & Roos D (1991) The Machine That Changed the World: The Story of Lean Production, Harper Perennial.

Woodman I & Hunter R (1996) Analysis of Assessment Data from Phase 1 of the SPICE trials, Software Process Newsletter (6): 1–6.

Yin RK (2009) Case study research: Design and methods, Sage publications, INC.

Zubrow D, Hayes W, Siegel J & Goldenson D (1994) Maturity Questionnaire, CMU/SEI-94-SR-7.

Original publications

I Kuvaja P & Bicego A (1994) BOOTSTRAP – A European assessment methodology. Software Quality Journal 3(3): 117–127.

II Similä J, Kuvaja P & Krzanik L (1995) Bootstrap: a software process assessment and improvement methodology. International Journal of Software and Knowledge Engineering 5(4): 559–584.

III Bicego A & Kuvaja P (1996) Software process maturity and certification. Journal of Systems Architecture 42(8): 611–620.

IV Kuvaja P (1999) BOOTSTRAP 3.0 – A SPICE conformant software process assessment methodology. Software Quality Journal 8(1): 7–19.

V Kuvaja P, Palo J & Bicego A (1999) TAPISTRY – A Software process improvement approach tailored for small enterprises. Software Quality Journal 8(2): 149–156.

VI Kuvaja P, Maansaari J, Seppänen V & Taramaa J (1999) Specific requirements for assessing embedded product development. In: Oivo M & Kuvaja P (eds) Proceedings of International Conference on Product Focused Software Process Improvement – Profes '99. VTT Symposium Series 195: 68–85.

Original publications are not included in the electronic version of the dissertation.

589. Riipinen, Katja-Anneli (2011) Genetic variation and evolution among industrially important *Lactobacillus* bacteriophages

590. Lampila, Petri (2011) Populations and communities in human modified forest landscapes

591. Liukkunen, Kari (2011) Change process towards ICT supported teaching and learning

592. Segerståhl, Katarina (2011) Cross-platform functionality in practice : Exploring the influence of system composition on user experiences of personal exercise monitoring

593. Tiikkaja, Marjo (2012) Value creation in collaboration between software suppliers and customers: suppliers' perspective

594. Rousu, Timo (2012) Liquid chromatography–mass spectrometry in drug metabolism studies

595. Kangas, Teija (2012) Theoretical study of the oxidation of a pure and alloyed copper surface

596. Härkönen, Laura (2012) Seasonal variation in the life histories of a viviparous ectoparasite, the deer ked

597. Niinimäki, Sirpa (2012) Reconstructing physical activity from human skeletal remains : Potentials and restrictions in the use of musculoskeletal stress markers

598. Mandić, Vladimir (2012) Measurement-based value alignment and reasoning about organizational goals and strategies : Studies with the ICT industry

599. Leiviskä, Katja (2012) Why information systems and software engineering students enter and leave their study programme : A factor model and process theory

600. Siira, Tuula (2012) Value Creation by Enterprise Systems Value Added Resellers : The Case of PLM Systems VARs

601. Kontula, Jukka (2012) New venture creation in software business : A contextually embedded entrepreneur's perspective

602. Juntunen, Kaisu (2012) Tieto- ja viestintätekniikan soveltamiseen perustuvat toimintaprosessien uudistukset terveydenhuollossa : Sosio-teknis-taloudellinen näkökulma

603. Seppä, Karri (2012) Quantifying regional variation in the survival of cancer patients

Book orders:
Granum: Virtual book store
http://granum.uta.fi/granum/

UNIVERSITY OF OULU P.O.B. 7500 FI-90014 UNIVERSITY OF OULU FINLAND

ACTA UNIVERSITATIS OULUENSIS

ISBN 978-952-62-0029-3 (Paperback)
ISBN 978-952-62-0030-9 (PDF)
ISSN 0355-3191 (Print)
ISSN 1796-220X (Online)

UNIVERSITY of OULU
OULUN YLIOPISTO