Management competences, not tools and techniques: A grounded examination of software project management at WM-data

Jeremy Rose a,*, Keld Pedersen a, Jens Henrik Hosbond a, Pernille Kræmmergaard b

a Department of Computing Science, Aalborg University, Denmark
b Department of Business Studies, Aarhus Schools of Business, Aarhus University, Denmark

Available online 11 February 2007

Abstract

Traditional software project management theory often focuses on desk-based development of software and algorithms, much in line with the traditions of the classical project management and software engineering. This can be described as a tools and techniques perspective, which assumes that software project management success is dependent on having the right instruments available, rather than on the individual qualities of the project manager or the cumulative qualities and skills of the software organisation. Surprisingly, little is known about how (or whether) these tools techniques are used in practice. This study, in contrast, uses a qualitative grounded theory approach to develop the basis for an alternative theoretical perspective: that of competence. A competence approach to understanding software project management places the responsibility for success firmly on the shoulders of the people involved, project members, project leaders, managers. The competence approach is developed through an investigation of the experiences of project managers in a medium sized software development company (WM-data) in Denmark. Starting with a simple model relating project conditions, project management competences and desired project outcomes, we collected data through interviews, focus groups and one large plenary meeting with most of the company’s project managers. Data analysis employed content analysis for concept (variable) development and causal mapping to trace relationships between variables. In this way we were able to build up a picture of the competences project managers use in their daily work at WM-data, which we argue is also partly generalisable to theory. The discrepancy between the two perspectives is discussed, particularly in regard to the current orientation of the software engineering field. The study provides many methodological and theoretical starting points for researchers wishing to develop a more detailed competence perspective of software project managers’ work.

© 2007 Elsevier B.V. All rights reserved.

Keywords: Software project management; Software engineering; Competence

1. Introduction

The task of managing a software project can be an extremely complex one, drawing on many personal, team and organizational resources. Even ‘successful’ software projects are acknowledged to have many problems, such as schedule and budget overrun, and software products with quality deficiencies. Unsuccessful projects can result in substantial waste of resources. The software engineering research tradition rightly concerns itself both directly with techniques for constructing software, and with the management of software projects. Classic software engineering texts such as [1,2] prescribe many techniques for building software and for managing the software process, but hardly ever relate those techniques to examples of real software projects. Texts that do describe real projects, for example [3] often confine themselves to anecdotal examples, without any discernible research process. Lack of discernible research process is often (unfortunately) characteristic in the areas of the literature where there are examples of case histories, for instance in the stories of CMM (Capability Maturity Model) projects [see 4]. This article will argue that a tools and techniques perspective...
dominates the software engineering literature. Moreover, the prescriptive tools and techniques are often the result of desk-based research. It will further develop the theoretical basis for an alternative, competence-oriented perspective to software project management. Competence is here understood as knowledge and skill enacted in practice. Knowledge is information stored and interpreted in the human mind. Skills are obtained through experiences and based on knowledge. Competence is, according to Dreier [5], the ability to transform knowledge and skills into practice in a qualified way. This alternative competence account is methodologically rooted in descriptive qualitative ways of collecting and analyzing data, and of generalizing to theory; also in contrast to the dominant methodological paradigm in the software engineering literature.

The research project described here used interviews and grounded qualitative analysis methods to generate descriptive theory about software project experience in the Danish subsidiary of a multi-national software firm, WM-data. A descriptive theoretical strategy for researching software project management starts with the premise that experienced project managers are the experts in software project management, not researchers. The research objectives are:

1. to identify and describe the project manager (PM) competences valued at WM-data;
2. to identify and analyze approaches to software project management in the software engineering literature;
3. to compare those the results of 1 and 2 and discuss research consequences for the software engineering field.

The qualitative research followed a grounded strategy, where theoretical descriptive categories and relationships are built up from the data, to avoid the possibility that the data collection and analysis could be biased by the way project management is currently described in the software engineering literature. A literature study was conducted subsequently to establish how software project management is understood in the software engineering literatures. The difference between the way that software project management is portrayed in our literature sample, and the way it is conducted at WM-data is also discussed.

2. Research method

The research is located within the interpretive [6,7] and hermeneutic [8,9] traditions where social realities are assumed to be constructed, and research conclusions are reached by studying the interpretations of the research subjects. These interpretations are located in texts (here interview transcriptions) which can be analyzed. Established qualitative analysis methods (here causal mapping and content analysis) help with the researchers task of (1) generating a neutral description of the research subjects’ subjective perceptions, and (2) generalizing that description to theory by establishing categories reflecting patterns in the data. The research is grounded [10,11], in as much as the analysis is primarily based on empirical data collected on the basis of an intuitively derived model, rather than a theoretically derived framework or hypothesis. The purpose of these philosophical and methodological stances is to allow the opinions of experienced practitioners in software project management to be systematically collected, analyzed and categorized, without basing the research process on a particular software engineering research paradigm or tradition. The research instead follows the traditions of qualitative research more common in the information systems literature.

2.1. Data collection

Initial data was collected through ten semi-structured interviews lasting 1 1/2–2 h with experienced project managers at WM-data (Denmark) during the period September–December 2004. Interviewees were chosen to reflect the variety of types of projects that the company runs. The interviews were structured around the initial research model, and questions were designed to investigate what the characteristics of the project situations were, what skills and competences the managers used in running their projects, and how those skills and competences could lead to successful project outcomes. However, the participants were encouraged to tell narratives of successful and less successful project episodes and their own role in these episodes, without the interviewers insisting on any particular focus or limiting the discussion to particular concepts. The interviews were transcribed and analyzed, validated with the interviewees, and the developing analysis was tested and refined in two 3-h focus meetings with the interviewees. Provisional research conclusions were fed back to the company in a plenary session with all the company’s project managers and relevant senior managers present (comprising about 75 people in all), and the resulting discussion annotated.

2.2. Data analysis

Data collection and analysis follows broadly the strategy outlined in [12] for elicitation of tacit knowledge. However, we employed two complementary analysis methods which are widely used in grounded data analysis: causal mapping [13] and content analysis [14,15]. We used content analysis (with the analysis tool HyperText) in order to elicit and structure variables (concepts and categories), such as different kinds of project conditions or management competences. The project management competences discussed in detail in this paper were identified as hierarchically layered analysis codes (for example ‘team management_project staffing_matching skills to project needs’). The codes were identified iteratively through successive passes of the text (with two coders comparing results), and frequency counts helped identify much-discussed topics. However, content analysis is poor at establishing causal relationships (here understood as the connections that project managers
themselves make between project circumstances, how they manage and project outcomes, rather than a positivist theoretical description of causal relationships between independent and dependent variables). Instead we documented these links in causal maps, initially with one for each interview, but later with summary maps. These relationships are important for establishing the context, motivation, and perceived outcomes of project management competences. The analysis products were also translated back into more conventional textual and diagrammatic forms so that project managers could easily review them. The two analysis methods, spread between different analysts, help to triangulate the analysis, and avoid the effects of potential researcher bias.

2.3. Generalisation to theory

The preliminary theoretical model developed in this research rests mainly on the identification and analysis of concepts through content analysis, with concept mapping used as triangulation to help establish competence concepts and perceived relationships between competences and successful project outcomes. Reliability and validity are here dependent on the careful observation of patterns in the data, rather on the number of occurrences or sample size. The consequent theory identifies important competence concepts in the data, and establishes a conceptual organisation of those concepts. It thus provides a descriptive theory that can be used for ordering and understanding experience, rather than a prescriptive theory intended to establish causal relationships capable of predicting outcomes in the future.

2.4. Literature study

The literature study was conducted after the data analysis in order to add perspective without interfering with the research design’s grounded strategy. Ninety six contributions were identified from the Web of Science database using the keywords ‘software project management’. Table 1 shows a frequency count for the number of times journals were represented in the sample, excluding journals that were only represented once. The journal analysis confirms that the literature base is located in the software engineering tradition.

All the abstracts were analyzed and full-text versions of 26 key contributions were obtained.

2.5. Research design

The research design is given in Fig. 1.

2.6. Initial research model

The initial research model is given in Fig. 2. The intuitive starting point for the model is the understanding that software projects are different, and these differences can be expressed as different conditions, for example, size: a project can be small or large.

Different combinations of project conditions necessitate the use of different project management competences (defined as skills + knowledge refined in practice). Software
size estimation could be such a competence. These perceived competences can contribute, in project managers’ perceptions, to favorable project outcomes, such as good software quality. Though the analysis generates a wider understanding of causal relationships in software project management, this article focuses on categorization and description of organizational and individual project management competences.

3. Project management in the software engineering literature

3.1. The shape of the field

The software project management literature sample was analyzed in relation to two aspects: the research method (process), and the research product or outcome. In the absence of a suitable analysis model, the categories were developed in a grounded manner, as part of the analysis.

3.2. Research methods for project management in the SE literature

Categories for research method are explained in Table 2. The 96 articles in the sample were allocated to one of these categories according to their predominating research method and the results are displayed in Fig. 3.

The analysis shows that software engineering researchers display a heavy concentration on the development of software tools in their research into software project management. Taken together, researchers working with metrics, algorithms, software, tools and simulations represent half the contributions (SD, CSC and LE). These researchers do not seek to ground their work in direct research experience with software project management; this direct involvement occurs mainly as they try their tools and techniques out in practice situations. Normally they report that the tool or technique is helpful, but there are no examples of sustained interactions with practice situations leading to widespread adaptation of the advocated tools and techniques. The researchers working with existing project data have a kind of indirect experience of real project situations, but the data is not designed or collected with research purposes in mind, and researchers may lack the context knowledge necessary to interpret limited data collected for different purposes in meaningful ways. Practitioners and consultants report their hands-on experience, but normally without declaring a research process or theoretical orientation that could justify their conclusions. Only about a fifth of the articles used some kind of field study method (predominantly surveys and case studies) which put them in contact with live software project management.

3.3. Research outcome/product for project management in the SE literature

Another way of understanding a body of research is to understand what type of conclusion, outcome or product the research intends to supply. The iteratively developed categories for the analysis of research product or outcome are explained in Table 3.

When allocated to one of these categories for research product or outcome the literature sample displays the following distribution (Fig. 4).

Table 2
Research method analysis categories

<table>
<thead>
<tr>
<th>Research method</th>
<th>Description</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>Field study</td>
<td>FS Examination of real-life software projects, typically, field experiments, case studies, surveys, interviews</td>
<td>[16–18]</td>
</tr>
<tr>
<td>Project data analysis</td>
<td>PDA Analysis of existing project data (typically collected for the purposes of managing a project, rather than for research purposes)</td>
<td>[19,20]</td>
</tr>
<tr>
<td>Laboratory experiment</td>
<td>LE Investigation of laboratory subjects’ interaction with a software management tool, often a simulation</td>
<td>[21,22]</td>
</tr>
<tr>
<td>Computer science conceptual</td>
<td>CSC Desk-based development of metrics, algorithms, models and simulations intended to support the software management process</td>
<td>[23–25]</td>
</tr>
<tr>
<td>Software development</td>
<td>SD Development of a software product intended to support the software management process</td>
<td>[26–28]</td>
</tr>
<tr>
<td>Theoretical</td>
<td>TD Theoretical discussion or meta-discussion other than computer science</td>
<td>[29,30]</td>
</tr>
<tr>
<td>No research process</td>
<td>NRP No discernible or reported research method (often practitioners’ reports of their experiences)</td>
<td>[31,32]</td>
</tr>
</tbody>
</table>
The analysis confirms that the field is largely prescriptive and tool and technique oriented. Researchers focus their attention on the development of tools and techniques; often intended for use by software project managers. There is widespread use of mathematical models and solution algorithms. The tools and techniques include: refinements to techniques for estimation [38,46,47], risk management [20,30,48,49], quality assurance [43,50], knowledge reuse [51], project management audits [52], inspections [53], version control [26], metric use [24,32,35,54,55], testing [56], scheduling [44], and workload management [57]. They aim almost exclusively to support the management of the software process, rather than other elements of management reported below in our analysis of WM-data’s practice. The approach is largely mathematical and algorithm-based, employing Petri nets [58], genetic algorithms [59], fuzzy logic [44], entropy calculation [41], Kepner–Tregoe program, [60], bi-directional simulation [23], data mining [25], Monte Carlo simulation [61] and graph grammars [62]. There is a concentration on software support for process management: expert simulator [43], Unix file system [26], co-operative work support system [63], decision support system [44], project management support system [27], risk projection [60], Object Database Support for Software Project-Management Environment [64], and knowledge-based tools [54]. A significant strand in project management education is advanced through the development of simulations [21,65,66], and simulation techniques also play a significant role and understanding project management [61] and as a management tool [67]. Taken together, this emphasis is represented by the categories software system mathematical model and technique comparison (44% of contributions in our analysis). Other portions of the literature (represented by our categories prescriptive conclusions and conceptual model) also reflect the prevailing desire to support the software management process with advice and experience-organizing models (see Table 3).

The dominant paradigm in the literature (at least on the basis of this sample of articles) can therefore be described as desk-based, and tool and technique oriented. The descriptive strand of the software project management literature is much less in evidence. Descriptive work has as its primary object the goal of understanding practice, rather than that of supporting it. Descriptive theory represents reasonably secure understandings of practice which are transferable from one situation to another. Though descriptive work made up 31% of our literature sample, only 10% resulted in descriptive theory. Theory generation implies appropriate and rigorous research method, so descriptions of practice without suitable method result in descriptive conclusions which are not generalisable beyond the experience reported. Descriptive work in this literature is often used for other purposes than the generation of theory, for instance the derivation of ‘lessons leaned’ or the illustration of the authors’ thesis. Laboratory experiments can be used to understand (for instance) decision making behaviour, but often incorporate so many methodological

<table>
<thead>
<tr>
<th>Research outcome/product analysis categories</th>
<th>Description</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>Conceptual model CM</td>
<td>CM</td>
<td>[33,34]</td>
</tr>
<tr>
<td>Descriptive conclusions DC</td>
<td>DC</td>
<td>[31,35]</td>
</tr>
<tr>
<td>Descriptive theory DT</td>
<td>DT</td>
<td>[36,37]</td>
</tr>
<tr>
<td>Literature summary LS</td>
<td>LS</td>
<td>[38]</td>
</tr>
<tr>
<td>Prescriptive conclusions PC</td>
<td>PC</td>
<td>[16,39]</td>
</tr>
<tr>
<td>Predictive theory: causal model PTC</td>
<td>PTC</td>
<td>[20,40]</td>
</tr>
<tr>
<td>Mathematical model MM</td>
<td>MM</td>
<td>[41,42]</td>
</tr>
<tr>
<td>Software system SS</td>
<td>SS</td>
<td>[43,44]</td>
</tr>
<tr>
<td>Technique comparison TC</td>
<td>TC</td>
<td>[45]</td>
</tr>
</tbody>
</table>

**Table 3**

**Research outcome/product analysis categories**

The analysis confirms that the field is largely prescriptive and tool and technique oriented. Researchers focus their attention on the development of tools and techniques; often intended for use by software project managers. There is widespread use of mathematical models and solution algorithms. The tools and techniques include: refinements to techniques for estimation [38,46,47], risk management [20,30,48,49], quality assurance [43,50], knowledge reuse [51], project management audits [52], inspections [53], version control [26], metric use [24,32,35,54,55], testing [56], scheduling [44], and workload management [57]. They aim almost exclusively to support the management of the software process, rather than other elements of management reported below in our analysis of WM-data’s practice. The approach is largely mathematical and algorithm-based, employing Petri nets [58], genetic algorithms [59], fuzzy logic [44], entropy calculation [41], Kepner–Tregoe program, [60], bi-directional simulation [23], data mining [25], Monte Carlo simulation [61] and graph grammars [62]. There is a concentration on software support for process management: expert simulator [43], Unix file system [26], co-operative work support system [63], decision support system [44], project management support system [27], risk projection [60], Object Database Support for Software Project-Management Environment [64], and knowledge-based tools [54]. A significant strand in project management education is advanced through the development of simulations [21,65,66], and simulation techniques also play a significant role and understanding project management [61] and as a management tool [67]. Taken together, this emphasis is represented by the categories software system mathematical model and technique comparison (44% of contributions in our analysis). Other portions of the literature (represented by our categories prescriptive conclusions and conceptual model) also reflect the prevailing desire to support the software management process with advice and experience-organizing models (see Table 3).

The dominant paradigm in the literature (at least on the basis of this sample of articles) can therefore be described as desk-based, and tool and technique oriented. The descriptive strand of the software project management literature is much less in evidence. Descriptive work has as its primary object the goal of understanding practice, rather than that of supporting it. Descriptive theory represents reasonably secure understandings of practice which are transferable from one situation to another. Though descriptive work made up 31% of our literature sample, only 10% resulted in descriptive theory. Theory generation implies appropriate and rigorous research method, so descriptions of practice without suitable method result in descriptive conclusions which are not generalisable beyond the experience reported. Descriptive work in this literature is often used for other purposes than the generation of theory, for instance the derivation of ‘lessons leaned’ or the illustration of the authors’ thesis. Laboratory experiments can be used to understand (for instance) decision making behaviour, but often incorporate so many methodological
difficulties (sampling of subjects, missing context, unrealisti-
cric software parameters) that they often tell us more about
the experiment itself than the practice the experiments are
supposed to represent. It’s also possible to use a practice-
close method (for example field studies) for another
purpose than the development of descriptive theory, for
example to develop prescriptive advice for practitioners.
Live practice studies that utilize reliable method and to
develop descriptive theories are rare in our literature
sample. Excellent, but sparse examples are Keil’s studies
of project escalation [36,68,69].

4. WM-data

Scandinavian societies are some of the most highly
developed in the world, in terms of quality of life indica-
tors, economic indicators and technology development.
The society model is based on high taxes and well-devel-
oped social welfare, which nevertheless provide the basis
for flexible, innovative and competitive industry, particu-
larly in technology sectors. WM-data is one of the largest
Scandinavian suppliers of IT-based business solutions, with
7000 employees in 83 locations in Scandinavia and north-
eastern Europe. There are five sites in Denmark, where our
study was concentrated. The company’s core asset is under-
stood as its ability to run medium to large projects for gov-
ernment agencies or private corporations, rather than a
particular expertise in a technology or application area.
Like most software houses, WM-data has a matrix struc-
ture, which is organized around projects. There is a large
portfolio of system development projects of varying sizes
across many application areas and technologies: the pro-
jects are normally quite independent of each other and
developers are expected to be able to switch between
technologies and application areas. The overall develop-
ment process and technical architecture for a project are
typically tailored to fit the customer’s needs. WM-data’s
project managers are almost invariable systems developers
themselves, they have typically worked on a variety of
house projects before being a project manager themselves.
Many projects have a technical project manager (as well as
a project manager) – this role is designed to allow the pro-
ject manager to focus on broader leadership issues instead
of being buried in technical details. Many project managers
have themselves been technical project managers as a link
in their career development.

5. Data analysis and theory generation process

Qualitative content analysis involves the coding of texts
(here the interviews) and the consequent isolation of ‘pat-
terns that exist in the empirical world under study’ [70].
This is an inductive form of reasoning involving ‘immer-
sion in the details and specifics of the data to discover
import categories, dimensions and interrelationships…by
exploring genuinely open questions’ [70]. Codes were de-
veloped to reflect the competences that the project managers
(implicitly or explicitly) discussed; thus each code reflects
some aspect of a competence they understood as important
in their work. Texts were independently coded by two
researchers to ensure concept validity, but the resulting
codes were iteratively discussed to establish both consis-
tency and the emerging hierarchical ordering. The resultant
categories were iteratively grouped by theme – i.e. similar
themes were grouped together with a headline theme, en-
abling the three level hierarchical coding scheme that
can be seen in Appendix A. A coding example is given in
Table 4. Simple frequency counts of codes helped to estab-

---

Table 4
Use of hierarchical coding scheme with explanations

<table>
<thead>
<tr>
<th>Text</th>
<th>Code</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Interviewer: ‘Can we start with you explaining what you think characterizes a good project?’: Bent: ‘Well…yes, I guess I can try. Well there are many things. A good project…oh… I really like to establish a good project culture…so, for example…we agree how we do things, we agree that here in the project we help each other if we come to have a problem…oh…oh that we know that there is always someone…we… I just always say that, in my projects, we give people permission to test the thin ice…but that there is always a safety net, because we are so big. We can always find people to pull them in again’</td>
<td>Team management_culture</td>
<td>Bent discusses how he manages his team by instilling a particular culture</td>
</tr>
<tr>
<td>Interviewer: ‘Yes. What do you mean by thin ice?’ Bent: ‘well really I mean that…oh…that could be something they do not really know much about’ Interviewer: ‘OK’ Bent: ‘so the responsibility in the end, and that’s also what permeates our project culture, that you can have just as much responsibility that you…oh…that you well…that you want. But, but this also demands that you always take it (indecipherable). My starting point is simply that it’s necessary that people take responsibility’</td>
<td>Team management_culture Mutual support</td>
<td>Bent explains a particular facet of the culture that he tries to instill</td>
</tr>
<tr>
<td></td>
<td>Team management_culture risk taking</td>
<td>Bent explains a further aspect of the desired project culture</td>
</tr>
<tr>
<td></td>
<td>Team management_culture Responsibility seeking</td>
<td></td>
</tr>
</tbody>
</table>
lish prevalent trends in the data patterns, and are the basis for selecting topics for description in the next section. Larger numbers of related codes (grouped under six major headings) plus frequent occurrence of codes in the text demonstrate thematic patterns in the text, which is the basis for generalization to theory.

The first level of the resultant coding scheme provides the basis for headings under which the findings are presented and the eventual categories in the theoretical model. Causal mapping was used for triangulation of the concepts isolated through content analysis. Causality is relevant to the discussion of competences because an important component of competence is that it leads to desirable outcomes. The maps establish causal relations between concepts with the direction of causality represented by arrows. A researcher not involved in content analysis mapped all the interviews (an example is shown in Appendix B). At the centre of each map lies a category named ‘project success,’ with many arrows pointing at it. The other bubbles and arrows map the chains of causality that project managers report. The maps were used for triangulation in two ways. Since the bubbles also represent categories or variables, these were cross-checked with the results of the content analysis to identify coding omissions and inconsistencies. However, clumps or clusters of bubbles with related themes, linked by causal relationships also help establish the major themes indicated in the following section. The list in Appendix A represents the finalized list of 352 codes and its hierarchical ordering into the six major categories described below.

Analysis in this depth also allows for the inclusion of contextual factors that lie not so much in the content of what is said, but the way it is expressed. An integrated picture is also available to researchers which is not available to any single project manager, and additional conclusions can be induced (in this case the sections on uncertainty management) which are only implicitly referred to by project managers. Results of the analysis were fed back to the participants in practitioner-friendly forms for validation and improvement though focus group meetings with the interviewees. Their comments were annotated and fed back into further analysis iterations. The presentation of the research results includes rich descriptions of the observed patterns in the next section (supported by quotations which ground the observations in the words of the participants), and a competence model in the following section.

Holistic perspective ‘the whole phenomenon under study is understood as a complex system which is more than the sum of its parts’ [70] ‘patterns that exist in the empirical world under study’ [70] hermeneutic circle.

6. Findings: Project management at WM-DATA

This section reports the results of the analysis of project management work at WM-data under seven headings (reflecting the major categories in the content analysis).

6.1. Technical management

Understanding the technical dimension of essentially technical work is an important competence at WM-data. Larger projects have a technical project manager as second in command to the project manager. Although we did not ask to interview technical project managers, many of the project managers had experience in this role. Project managers at WM-data are promoted through the company, so they have a technical education and experience as system designers and programmers. Technological understanding becomes especially important where the project involves a new or unfamiliar technology (here the manager might be chosen because of familiarity with a similar technology). In smaller projects, the PM will contribute to the technical work and need to have at least one technical area of competence. In a predominantly technical culture, technical skills are an important way of earning the necessary respect to manage other technical workers

“You can be respected for many things, but (technical understanding) is one of them...it also helps underpin many of the things you say, it’s enormously important...it helps avoid being too much of a dictator – you can argue from technical insight” Søren

In larger projects, it is not necessarily the PM’s responsibility to solve technical problems, though many of them like to do so, but any technical problem that remains unsolved and threatens project success rapidly becomes the PM’s problem. The PM has a responsibility for the technical overview – in a project with many different technical components and many specialists in those components someone must understand how the technologies fit together. They expect to be sparring partners for their team members and will often be responsible for explaining technical matters to their customer. Many of their other competences are partly dependent on understanding technical challenges (for example distributing tasks to the right team members), and they display and use a consistently technical vocabulary – also in their conversations with researchers.

6.2. Process management

Process management concerns the most traditional of software project management skills, which also reflect the scope of standard project management techniques. These skills are particularly important on larger, long running projects such as some of WM-data’s government contracts. Managing the process is about the choice of software development model, the planning, delegation and monitoring of tasks and roles, and hitting the project deadlines.

“You must focus on the task” Søren

“anyway we have some tools, you know planning tools, so what I do first is to check that I have enough resources, in the big picture. After that I simply list all the tasks. I identify
them from the development model, although I know well enough that some other things which I couldn’t identify will turn up later, but that’s just the way it is. I can’t see every task. However its important to get as many as humanly possible at the beginning, and then I simply go in and decide who is good at what or who needs to be trained up in a particular area of the system if we need to share knowledge or transfer knowledge or who can do what. These are the three criteria I chiefly use to distribute tasks.” Morten

“handbooks' common project coding standards, often defined in
improving skills. A stable process framework implies
tion with sharing smart ways of doing things and
code, whereas code reviews are used to combine evalua-
tion is necessary but not sufficient:

“you need a good plan, and updates on the way, check that
you follow up well – I was over there every week and talk-
ing through the tasks they were working on, even when
there weren’t any more customer meetings – and monitor
man-hours. Good plans, good follow up and updates along
the way” Niels

Monitoring and regular planning updates are equally
important. This means that PM’s stress the importance of
frequent checkpoints (milestones) in the project cycle. Con-
ected with the idea of checkpoints is also the reflective
process of evaluation and review

“we plan reviews after every version release…we stop and
discuss what went well and what went badly and where we
can economize…” Bent

Module and system tests focus on well-functioning
code, whereas code reviews are used to combine evalu-
ation with sharing smart ways of doing things and
improving skills. A stable process framework implies
common project coding standards, often defined in
‘handbooks’

“It’s important, especially when there’s a big turnover on
a project and new team members; they must learn all our
standards. We need a fair number of handbooks so that
they can have a fighting chance of learning how we do
the various things…we’ve handbooks for how you build
a module, a window, a printout, and the development
handbooks, then we have one that defines windows stan-
dards so that we can have a reasonably well-defined user
interface...something like 30 handbooks” Morten

Risk management is an important tool for anticipating
and planning for process difficulties; but if the planning
goes wrong then there are really only two alternatives:
move the deadline or reduce functionality

“The available mechanisms are either to make more time
by moving the delivery date, or to scale down what’s being
built” Ole

6.3. Team management

The subject that occupied most space in the PM’s discus-
sions was the management of their team

“a good project is first and foremost a well-functioning
group, that is a group that get along with each other” Morten

Staffing the project is extremely important; both in terms
of covering the necessary technical skills, but also in terms
of building a team that can work together, with comple-
mentary social skills

“if you’ve got a poor team, then you’ve got a poor
project” Niels

“you need a good team composition, with some different
profiles...technical competence profiles that must be able
to work together, and a good personal chemistry...it’s like
a football team, someone in goal, some defenders and
some attackers and some that can play in several
positions” Torben

Staffing is not always something that PM’s have full
control of, since they are normally dependent on who is
free in the organisation at the right time, and upon their
own political power, influence and negotiating skills to
compete for the organization’s resources. Staffing can be
particularly critical for small projects, where the loss of a
key team member can have serious consequences. In longer
projects the PM can expect to lose some team members
and have to replace them. Whereas team management can
partly be seen as the allocation of people to tasks.

“get the right people allocated to the right
tasks” Torben
PM’s must also consider knowledge transfer (should the module designer also be the module programmer because they understand the task better, or should there be a specialist designer and a specialist programmer who must communicate in some way), skills development (who needs to improve their skills in a particular area), intra-team communication (who will interact with the person in the corner who has irreplaceable technical skills, but poor social skills) and, in longer projects career trajectories (the project members must maintain and develop skills that will make them attractive to other projects). External contractors can fill holes in the team’s skills portfolio, but how will they become part of the team? It’s also important that the PM really delegates, not just the task, but also the responsibility for the task.

“You should be good at delegating... to say to people: here’s your task. It’s your responsibility to carry out the task and its not my intention to check every day if you’re doing it properly, however I do expect that you turn up when there are problems, and then I’m always ready to help you.” Morten

PM’s assume the responsibility for creating the correct project atmosphere. One PM described this as project culture – he was responsible for creating the WM culture in his projects. Team spirit cannot be taken for granted but must be monitored and developed. This might take the form of mutual engagement and respect.

“We treat each other properly, with respect” Bent

Shared responsibility

“Then it’s a question of creating a collective will, so the group feels that it has a collective responsibility for achieving the task” Morten

Or collective problem solving

“The project is the place where we help each other, when we have a problem” Bent

It is particularly important that team members do not sit on their own problems for long stretches of time (because this consumes valuable resources), but seek help from colleagues and the PM. PM’s try to foster a culture of openness, (particularly in their own accessibility) and a certain amount of shared decision making. The PM must be the chief motivator and commitment developer, and understand the sources of motivation in individual team members; whereas one is motivated by contact with users, another is excited by a technical challenge, and a third needs constant small successes or wins to stay motivated. The PM provides feedback so that the team and the team members can assess their personal progress and the project’s progress. However the PM is also the one who drives the team members.

“There should be someone who knows how to wield the whip” Kristian

The PM is also the person that protects them, perhaps from outside interference from the customer or other managers, perhaps when they have other problems at home, perhaps when the pressure is on and stress levels are high. In shorter projects this kind of culture must be established quickly, on longer ones it must be maintained through many years of co-operation and changing personnel. Social problems such as bad personal relationship with a colleague must be talked through and dealt with, rather than being left to fester. The ideal relationship in the project is one of mutual trust, rather than a control relationship between manager and staff, and the PM needs to instill a certain confidence in the team’s professionalism and ability to overcome challenges. This was described by Niels as ‘competent ignorance’ – not that the PM or the team had an instant answer to every problem, but they had the necessary level of professionalism to find the answers and solve the problems.

PM’s acknowledge that they have the principle responsibility for developing and maintaining a good team culture, and that they must lead by example.

6.4. Customer management

The customer is an essential part of the equation of software development. The customer is an organisation, but is made up of individuals: the customer project manager, steering committee representatives, experts, managers, accountants, users. In some development situations the customer is known from previous projects; however the relationship must be maintained and difficulties overcome. In other projects the customer is new, and a working relationship must be built up. Every project customer is also a potential repeat customer, because if the relationship goes well, there can be further orders and further projects.

“That personal contact with the customer, the importance of it working and that there is trust and suchlike, well that’s where I think the biggest challenge lies and I think its really fascinating and very important. Because you can follow up well, and report well and all the other things, but if the chemistry isn’t there, and you don’t have a good co-operation and a good relationship to the customer, I don’t think you can build that long-term co-operation that we’re looking for” Michael

A good working relationship is built upon trust, mutual understanding, and mutual respect. It is essentially a social relationship where realistic shared expectations are generated, those expectations are reliably fulfilled, problems are solved, and interest conflicts are defused or resolved. Superficial negative characterizations of the customer (‘they don’t know anything about technology’) and easy
blame allocation are to be avoided. PM’s see themselves as principally responsibly for establishing and maintaining the customer relationship.

“T’ve got these personal relationships (with the customer). Much of WM’s long-standing relationship over there goes through me. The relationship is personified in some way through me.”  Michael

Customer and developer organisations have different discipline backgrounds and culture, so communication can be difficult.

“you can sit and nod at each other and tell each other we agree, but the picture inside your head can be very different from that in the customer’s head. However if you have a trust relationship and know each other better…”  Torben

These difficulties need to be overcome if the project is to go well, normally, according to the PM’s, through a reasonable degree of openness and honesty. It is part of the projects remit to understand the customer’s work process and the users’ needs. It is particularly important that project agreements are understood in the same way, for example with the outcomes of a meeting. A deeper level of involvement than fulfilling a contract is desirable.

“we’re more than just a software supplier! I spend a lot of time communicating that we would like to be seen as more than just a software supplier. We’d like seen as involved, and as seeing things through. Maybe not exactly as a business partner, but in any case with a better co-operation that just a software supplier”  Michael

The project manager is a sparring partner, a technical consultant, a work process consultant, a social acquaintance in the pub and maybe even a personal friend. That engagement generates respect, and respect is a key factor in overcoming the inevitable problems that turn up in any project.

When there is a crisis, and there’s always some kind of crisis in a project, problems with a delivery or a schedule or something, you can hopefully deal with it in a professional way ‘we should do this and this and this’ it doesn’t help to stand and shout and insult each other...we must plan and be constructive, and you can only do that if you basically respect each other...and that respect must be developed before the problem occurs”  Niels

An important element of the PM’s role can be to educate the customer. Customers may need help with understanding the development model, with developing their requirements, understanding their own work process and so on. Naïve customers may have unrealistic financial and contract expectations, and must learn that when they change their mind, they need to pay for the resulting extra work. The PM cannot be dominated by the customer and must learn to stand up to constant requests for alterations and unreasonable demands. As in any relationship, harsh words can be exchanged at stressful times, and the PM must learn to distinguish personal matters from project problems, but to resolve both. One key is good information – if the customer has regular progress reports which also honestly indicate potential and developing difficulties, they are much better positioned to resolve conflicts before they become crises and can understand why a delivery may be delayed.

6.5. Business management

Software development companies must earn money to survive, and, as in all other companies in the commercial sector, this objective occupies a reasonable central place in every manager’s worldview.

“when we do something, we do it in order to earn money. That’s absolutely normal, and if you think it’s strange I’d like to ask if you know of a company that aims not to earn money”  Michael

A software company has a portfolio of projects, and some of these can lose money, but only if other projects can compensate for these losses by generating additional profits. This means that every project manager will have a more senior manager looking over their shoulder at the project’s bottom line. Profitability is essential. It is also at the centre of most of the PM’s resource allocation decisions. All software development problems can eventually be solved if enough resources are thrown at them; this is not the PM’s difficulty. The difficulty is that there is a budget and a deadline, and the project cannot consume more resources than the customer can be persuaded to pay for. The difference between the resources consumed and the resources paid for is the project’s profit – or loss. Successful project managers generate the company’s profits. Project profitability is a delicate balancing act which at first sight can seem impossible, but experienced PM’s manage it in spite of the difficulties.

“on the one hand you have the customer complaining that you are just too expensive...on the other hand a manager is looking at your profitability and telling you that it should really be better...and I must try to improve both things...get the customer to experience at they are getting more for their money, and nevertheless earn more on the bottom line...that can actually be done!”  Bent

The project’s starting position is often complicated by the way it has been sold; software development is normally tendered in a competitive environment and the salesman must make their bid financially attractive to the customer. This can result in financial and resource pressure right from the project’s outset, and in extreme cases can mean that it is sim-
6.6. Personal (self) management

We describe under this heading some competences that relate particularly to the personal skills of the project leader, and which are typically used orthogonally across many situations and across many of the competence areas already described.

PM’s are decision makers

“one of the most important leadership skills is the ability to take decisions. . . that you’re never afraid to take a decision. It could well be the wrong decision, but its important for the project members that they know that the project leader makes the decisions and stands by the consequences” Bent

Its important that decisions are taken in a timely manner, otherwise the PM risks becoming a bottleneck which is hindering the project. Despite something of an espoused focus on individual responsibility and communal decision-making in WM-data’s culture, there are also situations (perhaps with conflicting opinions or many unknowns) where the project manager simply has to take command

“occasionally I have to cut through the crap and say ‘that’s enough – now it’s going to be this way.’ They have to accept this because I have chosen to take a tough decision. On the few occasions I do this I’m also careful to give my reasons...and most people quietly accept it.” Morten

Decision making in the face of uncertainty also involves a form of risk taking – described by one PM as leaping off a building. PM’s are also responsible for overview, vision and direction for the project and its team

“You must obviously be able to create an overview of the situation – even if it seems chaotic. You must also be able to re-create that overview when things get out of hand and you feel you are losing it” Morten

On occasion they must display good change management skills (the example we came across concerned a long standing project that migrated from a fixed price contract to an hourly rate contract). They need good communication skills, because they must bridge the gap between different disciplines and differing types of discourse (customer manager, user, internal manager and programmer) and be good at expectation management. They must prioritise their own time, and also the various challenges that they are confronted with. They must also be able to manage their own stress, (which stems from many responsibilities and tasks) whilst remaining open to others and approachable, because they must learn about their team members’ and their customers’ problems in order to be able to solve them. They must further be able to project a certain re-assuring calm and confidence, even under difficult circumstances. Their own inter-personal attitudes and behaviours are very important for setting an example to team members, and for generating trust in the customer relationship.

“that’s the key to everything – that people know they can trust me” Michael

These attitudes and behaviours generate the respect which later becomes a kind of currency which can later be used to wield problem-solving influence. When things do go wrong, PM’s need good conflict resolution skills, and since most projects face some kind of crisis during their life time, they occasionally need crisis management skills.
6.7. Uncertainty management

Whereas the preceding competences are largely the result of analysis of interviews with project managers, the final competence reported in this paper is derived from analysis of the cumulative burden of management skills that software project leadership must encompass. What happens if your project concerns a cutting edge (un-tried) technology, has a new development method, a team you do not know and did not choose, a new and inexperienced customer, too few development hours in the contract and you yourself are an inexperienced project manager? You must be a reasonably inexperienced project manager otherwise you would have found a way to duck this project in favor of one that looked more manageable. This hypothetical example has so many obvious difficulties attached to it that a mature software firm such as WM-data might decline it or resolve some of the difficulties (for example by hiring external consultants well-versed in the new technology) at a more senior management level rather than leaving the PM to struggle with them all. Nevertheless, many software development projects, including many of those at WM-data, display self-reinforcing multiple major uncertainties which make them difficult to manage. In our hypothetical example: the new technology throws up development hiccups which delay a delivery date, which makes it difficult to establish a trust relationship with the inexperienced customer, which means that the customer becomes reluctant to pay for requirements changes, which negatively impacts the project budget, which means that internal managers start complaining about the project’s poor profitability and are reluctant to hire the external consultants that could fix the technical hiccups. The project manager’s life is difficult here, not because the problems are unknown or unexpected, but because the many uncertainties negatively impact each other, setting up a self reinforcing cycle of management difficulties. Whereas each uncertainty leads to a potential problem that can be tackled, the combination of uncertainties leads to multiple related problems that are much harder to manage. An extra effort to resolve one problem simply results in neglect of other problems or a new problem. The project manager insists that the project team work around the clock to meet the deadline, but this results in an exhausted team and uses up many expensive person hours, negatively affecting team morale and the project budget. The PM uses up much of his (or her) hard won personal capital to achieve this end and has less personal authority to help solve the next problem. Living and working with multiple uncertainties requires a particular set of PM competences which concern good anticipation, good prioritization, good expectation management, good personal stress management, the ability to spot the crucial link in the chain of problems (which can be attacked), and multi tasking – simultaneously working on multiple solutions to multiple problems. With many projects, especially newer projects without a long established history and tradition, uncertainty management is the key project management skill.

7. Seven competences for software project management at WM-data

In the pyramid at Fig. 5 we sketch the project management competences for software project management resulting from the analysis described above.

1. Technical management competence concerns the ability to manage in a technical software environment.
2. Process management competence concerns the ability to master traditional project skills in the planning and the management of resources.
3. Team management competence concerns the ability to form a satisfactory project team and manage relationships with the project members.
4. Customer management competence concerns the ability to develop and maintain a satisfactory relationship with the customer.
5. Business management competence concerns the ability to achieve a satisfactory financial result from the project.
6. Personal management competence relates to the personal skills of the project managers which can be used across different contexts.
7. Uncertainty management concerns the ability to manage complex interrelated problem sets resulting from multiple uncertainties.

The pyramid form suggests an impressionistic relationship between competences that is not justified by any particular research outcome (though it is partly based on concept frequency in the textual analysis).

Though the competence pyramid is primarily developed from conversations with project managers, only the sixth competence, personal management relates directly to competence that can only be located in the project manager. Other competences can also be located in the organisation. For instance, whereas the project manager will seldom have an unhindered choice of who will be in their team, it is the organisation’s responsibility to see that a compe-
tent team is assembled, which matches the project requirements. Most project management competences are not person specific; expertise that a particular PM does not possess can usually be compensated for by the deployment of other organizational resources. A good structural reminder of this at WM-data is that many projects are assigned a technical project manager: an additional leadership figure with a specific role and responsibility. The relationship between the project leader’s competences and wider organizational competences is an interesting one which we were not able to develop in this research.

The competence perspective presented in this analysis, derived from qualitative analysis, is presented as a descriptive theory, which we shall later argue is at least partly generalisable. In other words it is not intended as a normative model for how software companies and software project managers should manage their software projects. It should be noted that only competences 3–6 relate directly to what the project managers spent most of their time discussing with us. We call these reported competences, because we simply report in a structured way the content of the conversation. Technical management and process management competences were reported, but not discussed in nearly so much detail by the PM’s. However, secondary analysis and contextual factors lead us to believe that these are assumed or implicit competences. WM-data project managers assume that you cannot be a project manager without these competences. All the project managers we spoke with possessed these skills; however they were not necessarily the crucial skills which engaged their attention most. The final competence, uncertainty management, is neither reported nor assumed, but deduced. It was little discussed at the interviews, suggesting it was not uppermost in the PM’s thinking, but deduced by the researchers from the analysis, and subsequently introduced into the discussion at focus group and plenary meetings to confirm that the participants could recognize it as an important competence.

8. Discussion: tools and techniques perspective versus competence perspective

We presented in Section 3 an analysis of the conventional software engineering perspective on software project management. This perspective is normative, tool and technique based and usually the product of desk-based research. In management theory [71] it could be associated with the scientific management tradition in which rational means are used to analyze work tasks and provide optimized solutions to them. We provided a different perspective achieved though grounded qualitative analysis. Here we provided a descriptive theory of project management based on the idea of competence. This approach could better be associated with the human relations school of management theory [71], in which organizational success is primarily dependent on the capabilities of workers and the organization’s ability to harness those capabilities.

The two approaches do not necessarily compete with each other, either as descriptive tools, or as normative mechanisms. It is possible both to strive after better processes and techniques, and to maximize human resources. However, in this section we return to a discussion of the implications of our approach and findings for software engineering research.

Software project management tools and techniques did figure to some degree in our study. Amongst the tools and techniques in use are WM-data are conventional project management tools such as Microsoft Project, risk management tools (in the form of a spreadsheet with a graphical presentation), TroubleTicket (a fault reporting system and SameTime (a real time communication system). All the tools mentioned were widely used, off-the-shelf, traditional technologies. Whereas one project manager expressed a desire for better tools, another referred negatively to

“some huge Oracle thing that’s hopeless, because we’ve finished the project before it’s up and running – and we’ve used all our development hours on it” Torben

PM’s discussion of their tools and techniques represent less than one side of some 200 sides of transcriptions and analysis. Project managers do routinely use conventional project management tools, and ability to use them is expected (a kind of process management competence). However, they do not consistently report tools and techniques (or their lack) as important in solving the problems they face. Neither, to be fair, did they often directly refer to competences, but they did discuss many things that were easy for us later to represent as competences. Whereas both tools and techniques or competences can be valid ways of researching software project management, a more serious mismatch is between the way software engineering researchers conceive their research and the way real project managers manage. If software engineering researchers understand that the way to improve software project management is to provide better process management tools and techniques, but software project managers have totally different concerns, then this represents a serious mismatch that needs addressing.

There are some reasons to be concerned where a literature displays a strong normative (prescriptive) bias without a corresponding descriptive theoretical foundation. Tools and techniques are invariably based on assumptions about practice. Thus a simulation displays the results of the mathematical manipulation of a set of parameters. Both the parameters and their mathematical relationship are assumed to mirror a generalized account of practice in a meaningful way. Risk management techniques specify common sources of risk in practice; estimation techniques specify data collection parameters such as size, breakage and effort assumed to be meaningful, and so on. In addition, tools and techniques are intended to be used in practice; they must therefore display a reasonable correspondence with practitioner understandings of project
management practice in order to be usable. Thus good
descriptive theory is a pre-condition both for the develop-
ment and the adoption of usable tools and techniques.
Normative tools and techniques research can be seen as a
kind of design theory [72], where one of the conditions
for good theory is a reliable theoretical grounding (on
which the design theory is based); good descriptive theo-
ries can form this grounding. Conversely, a design theory that
is poorly related to existing practice should be expected to
be difficult to implement.

Related literatures, such as the project management lit-
erature (which also contains work on IT projects) and the
information systems literature (which focuses more on the
management of IT developments in conventional organis-
tions rather than software firms), also display a strongly
normative and process oriented focus on tools and tech-
niques (though with less concentration on the development of
software support). However, these literatures also dis-
cuss alternative perspectives, and include better descriptive
accounts of practice (including some limited discussion of
competences).

For these reasons we think that the competence perspec-
tive which we have begun to outline represents a promising
line of research which can provide helpful theoretical
descriptions of practice for the software engineering field.

9. Conclusions

In this study, we presented the results of an analysis of
the software engineering literature concerning software
project management, which we found to be dominated by
desk-based research of a normative character, primarily
focused on the development of tools and techniques to
assist process management. However, when we used two
qualitative research techniques to develop grounded
descriptive theory of software project management in a
successful Danish company, we concluded that the project
managers had wider and broader concerns, which we char-
acterized as seven competences. Though the use of tools
and techniques formed a part of the process management
competence, it was not the principal focus of the project
managers, who typically had many other concerns in lead-
ing successful projects.

Though there are some limits to the generalizability of
our competence theory of software project management,
it does have certain strengths. It is well-grounded in prac-
tice, and focuses both on hard and soft management skills.
It is broader in scope than traditional process- and tech-
nique-based project management theories and assumes that
successful project managers are the experts in project man-
agement. The researchers’ role is to document that expert-
ise and generalize it to theory. Qualitative research
method is both methodologically secure and well adapted
to this task. A traditional prejudice against it in the harder
engineering research disciplines does not disqualify it, though
it may make acceptance of its conclusions harder. Conventional software engineering research into project
management can also benefit from the development of
other management perspectives. Design theories and the
development of tool support for project managers can be
improved by better grounding in qualitative descriptive
theories of practice. Descriptive theories help establish bet-
ter understandings amongst practitioners which lead to
better practice results. Better focus on practice helps avoid
undue concentration on a narrow range of process factors
that are only a part of the total picture and could lead to
the development of a broader range of tools. Furthermore,
software tools, particularly simulations, can play a sub-
stantial role in the development of appropriate competenc-
es in software project managers.

We developed a competence pyramid that can represent
the starting point for a competence theory of project man-
agement. We further demonstrated a methodological
approach (based on the qualitative research tradition) suit-
able for further development of the competence perspec-
tive. In future research, we hope to add further levels of
detail to the seven competences, and a better account of
their relationship with each other. We also expect to
develop an account of the different project conditions
under which the competences become useful, and relate
the competences to specific desirable project outcomes.

Acknowledgements

Thanks to Otto Veie and colleagues at WM-data
Denmark for inviting us to help investigate their project
management. Thanks also to the project managers co-
operating in the study: Niels Jacobsen, Soren Rieck, Torben
Brennegard-Nielsen, Bent Handberg, Morten Skifter
Jensen, Kristian Dahl, Michael Hermansen, Ole Thesbjerg
and Leif CharmothNielsen.

The research was in part made possible by the Danish
Government through their sponsorship of the Software
Processes and Knowledge project.

Appendix A. Alphabetical list of project management
competence codes developed through Hypertext analysis

ability to develop domain understanding
ability to develop domain understanding
application area knowledge
business management sales group_improved honest dialog
is needed
business management sales group_not considering the
customer situation
business management sales group_not realistically
estimating new projects
business management sales group_responsible for extra
introduction time
business management_business management sales
group_had at judging the expected project approach
business management_concept development
business management_contract management
business management_creating new business
business management_economic
business management_economic_billing
business management_economic_cost reduction
business management_economic_hours estimation
business management_economic_profitability
business management_independence from management
business management_project promotion
business management_salesmanship
customer understanding_something to strive for
customer_aligning expectations
customer_communication
customer_consequences of small budget and much uncertainty
customer_courage and willing to take risks
customer_credibility fosters future business
customer_customers work together
customer_early conflict resolution
customer_education
customer_education_development models
customer_education_not aware of his own demands and requirements
customer_education_pay for changes
customer_establishing trust, mutual understanding and social relationship
customer_extra effort is needed
customer_feeling that you have let the customer down
customer_giving in to continuous customer demands
customer_influence their project
customer_informing on progress
customer_involvement
customer_involvement_fostering respect
customer_justifying economic relationship
customer_keeping an open dialog with the customer
customer_knowing the customer
customer_knowing when it is a personal or project matter
customer_maintaining shared understanding of agreements
customer_meeting decisions record
customer_open culture with
customer_participation
customer_partnership
customer_personal relations may spin off new projects
customer_principle contact
customer_problems when comparing old and new system
customer_professionalism
customer_project leader
customer_relationship
customer_relationship_involvement in customers work
customer_responsibility to
customer_setbacks because of unclear requirements
customer_shared expectations and objectives
customer_social relations helps understanding the customer
customer_sparring partner
customer_steering committee
customer_take seriously
customer_train to pay for contract changes
customer_trust
customer_user focus
customer_user participation
customer_work process understanding
facilitate knowledge sharing
independent decision maker
knowledge creation
match competences to project phase
organisation balance common needs with individual project needs
organisation resources available for use
personal_acknowledge feedback and change things
personal_age is a delimiting social factor
personal_approachable
personal_avoid being bottle-neck
personal_bad at solving unspecified tasks
personal_balance short and long sight
personal_being more up-front and demanding
personal_change management skills
personal_communication skills
personal_communication skills_bridging the gap
personal_compensate for gaps
personal_complexity management
personal_complexity management
personal_conflict management
personal_constructive attitude to problems
personal_control - steering and flexible
personal_control the process from inside
personal_cooperating with external project mangers
personal_cope with appearing incompetent
personal_creating respect from participants
personal_crisis management
personal_daily routines
personal_decision making
personal_educational background
personal_emotional_not good at dealing with bad relations
personal Entered project at day one of production
personal_establish bridges with stakeholders
personal_expectation management
personal_experience
personal_experience sharing with other pms
personal_external pm_consequence in management
personal_external pm_cultural understanding
personal_external pm_different culture
personal_external pm_different methods
personal_facilitator
personal_facing customer with confidence and self-belief
personal_flexible
personal_good at handling concrete tasks
personal_improvise
personal_involved
personal_keeping an open dialog with executives
personal_knowing your limitations
personal_leap off building
personal_limit to how many things can be initiated
personal_loneness and raw project management
personal_manage stress
personal_managing customer project managers
personal_managing expectations
personal_must not be too old
personal_negotiation skills
personal_not good at conflict management
personal_not liking too many after work social gatherings
personal_not pms used to formal procedures
personal_open management style
personal_openness
personal_overcome team inertia, remove safety net
personal_overview
personal_planning skills
personal_predict difficulties
personal_prioritising
personal_problems in creating trust to executives and
customer
personal_problems when not knowing the technical domain
personal_process improvement role
personal_process skills
personal_quick decision
personal_responsible for quality
personal_robust
personal_salami technique when negotiating
personal_signalling confidence and calmness
personal_sitting in a different room
personal_small project imply practical tasks
personal_social skills towards customer and participants
personal_stress management
personal_successful experiences
personal_time management
personal_trustworthiness
personal_uncertainty management
personal_visionary
personal_work without formal procedures
personal_cope with wasted work
process focus
process_management_adjustment
process_management_analyse progress
process_management_analysis and design
process_management_checkpoints
process_management_common direction
process_management_continueous improvement
process_management_continuous re-planning
process_management_crisis management_documentation
process_management_establish framework
process_management_estimation
process_management_evaluation/review
process_management_faster delivery
process_management_flexibility of working method
process_management_focus
process_management_follow up on project status
process_management_formal procedures
process_management_improvement_initiatives
process_management_initial phase
process_management_match development model to
customers culture
process_management_methodology
process_management_methodology
process_management_overview
process_management_overview
process_management_phase for considering technology
process_management_planning (also early)
process_management_planning__slack
process_management_planning_follow up on estimates
process_management_preanalysis
process_management_pressure and expanding deadlines
process_management Prototype
process_management_risk management
process_management_risk management
process_management_standards and cookbooks
process_management_standards and cookbooks
process_management_structure versus uncertainty
process_management_task monitoring
process_management_tasks with people
process_management_test
process_management_timely delivery
process_management_tools
process_management_uncertainty management_make
visible uncertainties
process_based on prototype tests
process_formal procedures at a minimum
process_initial phase_consensus between sales and
production
process_management_define responsibilities
process_management_managing demands and conditions
process_management_move the deadline or skip functionality
process_management_new iterative model
process_management_only making what has been ordered and not more
process_management_holding to agreements and deadlines
professional_competence
software_quality
stability and change
steering committee: loyal
team_management between customer and project members
team_management_avoid self satisfaction
team_management_buffer
team_management_coaching
team_management_communication
team_management_consistent employment
team_management_continuous employment
team_management_crisis management
team_management_culture
team_management_culture
team_management_culture_clear framework
team_management_culture_closeness
team_management_culture_competent ignorance
team_management_culture_example setting
team_management_culture_fostering commitment
team_management_culture_good atmosphere
team_management_culture_harder on shorter projects
team_management_culture_mentoring
team management_culture Mutual decision making
team management_culture Mutual support
team management_culture Mutual understanding
team management_culture Not dealing with bad social relations
team management_culture Not wanting responsibility
team management_culture Open access to pm
team management_culture Problem acknowledgement
team management_culture Process focus
team management_culture Responsibility seeking
team management_culture Risk taking
team management_culture Socialise through personal interests
team management_culture Trust
team management_example Setting
team management_expectations
team management_feedback
team management_firing
team management_focus on productive work
team management_focus on productive work for customer
team management_geographically dispersed team
team management_good atmosphere/team spirit
team management_hiring
team management_hiring
team management_individual understanding/support (e.g. well-being at home)
team management_influence over tasks
team management_limit setting
team management_monitor according to need
team management_monitor team’s well-being
team management_motivating staff
team management_motivation
team management_motivation through contact with their users
team management_motivation through success
team management_open culture
team management_openness to change
team management_proactively handle dissatisfaction
team management_professional pride
team management_protecting team members
team management_relief of pressure
team management_resource allocation
team management_respectful treatment
team management_responsibility delegating
team management_role allocation
team management_role delegation
team management_same people for analysis and develop
team management_shared understanding facilitator
team management_signalling professionalism fosters trust
team management_skills development
team management_social relationships
team management_staffing
team management_staffing
team management_staffing a risk for small projects

J. Rose et al. / Information and Software Technology 49 (2007) 605–624
Appendix B. Example causal map

References


