Tacitly developed research methods in ICS[1]

Ivar Tormod Berg Ørstavik
HiST

Abstract

Methodology is the study of research methods. This study tests three different methodological approaches to the same material, 93 doctoral theses from NTNU, to learn more about (a) the research methods actually in use in ICS and (b) each methodological approach's ability to shed light on (a). First, a KWIC analysis of "research method" is done. Despite being both primitive and time consuming, the KWIC analysis depicts well the extent and scope of the methodological debate in the material overall. Second, research methods employed in 772 thesis chapters are classified using both Zelkowitz's (1997) and Ramesh's (2004) taxonomies. This study finds that "research method per text" classification is much more problematic than previous studies have suggested and tells us more about the taxonomies themselves than the research methods employed in the material. Third, examples of "ICS engineering" used as a research method are drawn from the material. Trend exemplification is highly subjective and qualitative, rendering the findings tentative. Despite its weaknesses, trend exemplification manages to positively shed light on a tacit research method employed in ICS and some of its methodological premises.

Introduction — Information and Computer Science

ICS is a broad academic field spanning disciplines such as Computer Science (CS), Information Systems (IS), and Software Engineering research (SE) (Vessey 2005). These disciplines have different but related areas of interest and overlapping focus. CS's prime interest is computer systems and focuses mainly on the "technical levels" of their design and implementation (Ramesh 2004: 174). IS shares CS's interest in system design, in addition to an interest in how these systems function in organizations and business (Hevner 2004). This leads to a similar but more abstract focus on system design, in addition to a more practical focus on system use in social settings. Lastly, SE "seek better ways to develop and evaluate software" (Shaw 2003: 727). SE focuses on the system–man processes behind and surrounding the systems of both CS and IS, opening up many questions regarding social interaction with systems, questions similar to those pursued by IS and those relating to technical aspects of systems in CS.

Furthermore, it is not only the overlapping scopes and interests that unite the disciplines of ICS: CS, IS, and SE all share a common heritage from engineering. The engineering tradition has a strong preference for applied scientific results, and this has in turn established a strong preference for applicable research results in ICS disciplines. Tichy (1995: 9) describes this tradition as follows: "A large part of CS research consists of proposing new designs: systems, algorithms, and models. Such designs must be judged by whether they increase our knowledge about what are useful and cost-effective problem solutions." The system is first and foremost a product of application. The pursuit of results in ICS is, however, not restricted to man-made artifacts but also extends to human actions and practices: "Software engineering is concerned with techniques useful for the development of effective software programs" (Zelkowitz 1997: 735).

Several studies reviewing ICS literature (Orlikowski 1991; Ramesh 2004; Shaw 2003; Tichy 1995; Zelkowitz 1997) show a result-oriented research focus. Potentially applicable outputs from the research process dominated ICS journals up to the late 1990s in the form of descriptions of system designs and implementations, applicable

[1] ICS refers to the study of Information and Computer Systems and their use and development.

This paper was presented at the NIK 2008 conference. For more information see http://www.nik.no/

methods for system development, and experience applying information systems. But these studies also concur on another point: too little emphasis has been placed on describing the implied, tacit premises on which these results rest. Examples of such premises can be the research methods by which the results were obtained, the pros and cons of these methods, the empirical data, and the theoretical background underpinning conclusions. This lack of emphasis on research premises calls for attention, and this study examines tacitly developed research methods in ICS.

The status of research methods in ICS

Research focus and interest is not an unlimited resource in any discipline. Too strong an applied, result-oriented focus can overshadow other less applicable topics. In their argumentation for applicability, Hevner (2004: 81) illustrates this negative side effect: "A justified theory that is not useful for the environment contributes as little to the IS literature as an artifact that solves a nonexistent problem." Particularly vulnerable in such traditions is the research process. From a strong applied research position, the internal workings of research processes are only indirectly relevant: the research method is a means to an end. Devaluation of research topics that are not externally useful can thus inadvertently influence "highly practical research in the field of information systems [to] simply ignore the research method issue" (Baskerville 1998: 90).

It is possible to interpret such a lack of interest in research methods as an absence of research methods altogether: if no descriptions of the researcher's methods, processes, or empirical data are given, he or she might have had none to begin with. But a lack of discussion can also only mean a lack of discussion. Common knowledge about computer scientists' extensive and highly standardized training suggests that they share an extensive reservoir of methods. ICS literature also "exhibit[s] a single set of philosophical assumptions regarding the underlying nature of the phenomena being investigated, the appropriate research methods to be used, and the nature of valid evidence" (Orlikowski 1991). This discipline coherence contradicts a view of ICS as "methodless" and supports a view of ICS as a discipline relying on the tacit use of research methods.

The historical lack of open discussion on research method has resulted in "a lack of methodology for study and research in scientific software development" (Diaconescu 2002: 72). Missing is systematic and appreciated discussions about different research methods, that is, a methodology: "CS has not developed a concise taxonomy of methods applicable for demonstrating the validity of a new technique" (Zelkowitz 1997: 1); "SE researchers rarely write explicitly about their paradigms of research and their standards for judging quality of research" (Shaw 2002: 1).

The status of research methodology in ICS

Several reviews of ICS literature (Glass 1995; Orlikowski 1991; Tichy 1995; Zelkowitz 1997) address the lack of explicit discussions about research methods. These studies are part of a blossoming critique of ICS methodology that is scattered across ICS. As a whole, this discussion is too wide to be included in this study, but a couple of descriptive trends will be presented.

In SE, several studies (Basili 1986; Tichy 1998; Zelkowitz 1997) propose "experimentation" as the common denominator for SE research methods, and today the International Software Engineering Research Network "is a community that believes software engineering research needs to be performed in an experimental context" (ISERN 2007b). This methodological tradition places a high emphasis on *observation* as a research approach, and "experiment" is defined as "a trial that is conducted in order

to verify a hypothesis defined beforehand in a controlled setting in which the most critical factors can be controlled or monitored" (ISERN 2007a). More on recent developments in this trend can be found in (Sjøberg 2007; Sjøberg 2005; Zannier 2006).

Baskerville (1998: 90) suggests "action research" as a suitable and fruitful method for IS: "The discipline of IS seems to be a very appropriate field for the use of action research methods. IS is a highly applied field [and] action research places IS researchers in a 'helping role' within the organizations being studied." The fit suggested between IS and action research thus builds on and strengthens the result-oriented, applied engineering tradition in ICS and couples it with a participatory methodological tradition.

Common to both trends sited above is the extent to which they have drawn on external methodological traditions from natural and social sciences in shaping their methodological taxonomies and arguments. The "experimental" trend states that there are problems with currently used ICS research methods, and both trends promote importing methods from other sciences to enhance ICS research methodology.

However, when studying methods in a field, it is crucial that the methodology used to evaluate existing and new methods suits the field and its problems and objectives. All methodological frameworks are adapted to suit a particular set of research problems, empirical data, and objectives. If a methodological framework is transferred from one academic discipline to another, it will likely be confronted with a different set of research problems, theories, empirical data and objectives. Such a shift of scenery is likely to result in a mismatch between a methodology tailored to one field and methods tailored to another. The resulting discrepancy between the imported methodology and existing research methods might at first glance appear as a deficiency with existing methods but actually result from a skewed methodological approach.

Several indications suggest that the discrepancy between the methodology and methods in ICS has more to do with the makeup of these frameworks than with the methods used. One example is Ramesh (2004). Their taxonomy presents 22 different research methods, many inspired by other disciplines. In their study of 628 articles, 11 categories were not matched with a single article; 4 categories matched only 0.8% (5 of 628 articles); and 2 almost identical categories accounted for 88.54%. A similar problem was identified with Zelkowitz's (1997) taxonomy. In their study, 9 of 13 categories for research methods accounted for only 16.9%. Two categories on the fringes or outside of their methodological framework accounted for almost two-thirds of the articles. One conclusion that can be drawn from these studies is that studying research methods in ICS should not only focus on what, how, and why research methods are used but also on what, how, and why methodological frameworks are used to describe this situation.

A KWIC case study — the methodological debate at NTNU

Our first question concerns the methodological debate in ICS: Are research methods discussed, and if so, how? To address this question, a "KeyWord In Context" (KWIC) analysis of the keyword "method" was performed on 93 doctoral theses in ICS at NTNU from 1980 to 2006 (Conradi 2008).

First some comments about the material. As a genre, the doctoral thesis should cover a lengthy research project in a full and lengthy report. The same concerns for brevity that might exclude discussions of research premises, such as research method, from shorter articles should therefore not affect doctoral theses to the same extent. Thus the doctoral thesis should reflect existing methodological discussions in the field. The material in this study is limited to the main type of doctoral thesis (dr. ing.) at one engineering department at one university (ibid. nos. 26–123, excluding 72, 81, 89, 96,

and 102 for various practical reasons). These doctoral theses are dominated by CS and SE. Therefore, this analysis yields only a local view of ICS practices and should be read as such.

The results from the KWIC analysis (figure 1) are clearly consistent with the situation described by previous studies of ICS literature (Orlikowski 1991; Ramesh 2004; Shaw 2003; Tichy 1995; Zannier 2006; Zelkowitz 1997). In the 35 theses from 1980 up to 1995, the term "method" is used only six times in the context of "research method." The KWIC analysis then shows a shift in trends occurring in the mid 1990s. From the mid-1990s onward, several theses mainly within SE devote lengthy discussions and entire chapters for discussions of research method. But most theses continue to refer less than five times to the term "method" in the context of research method. This analysis therefore confirms previous findings that ICS relies on tacit use of research methods. The analysis also, however, suggests that within ICS (and mainly SE) a segment has evolved in which research methods are studied and discussed.

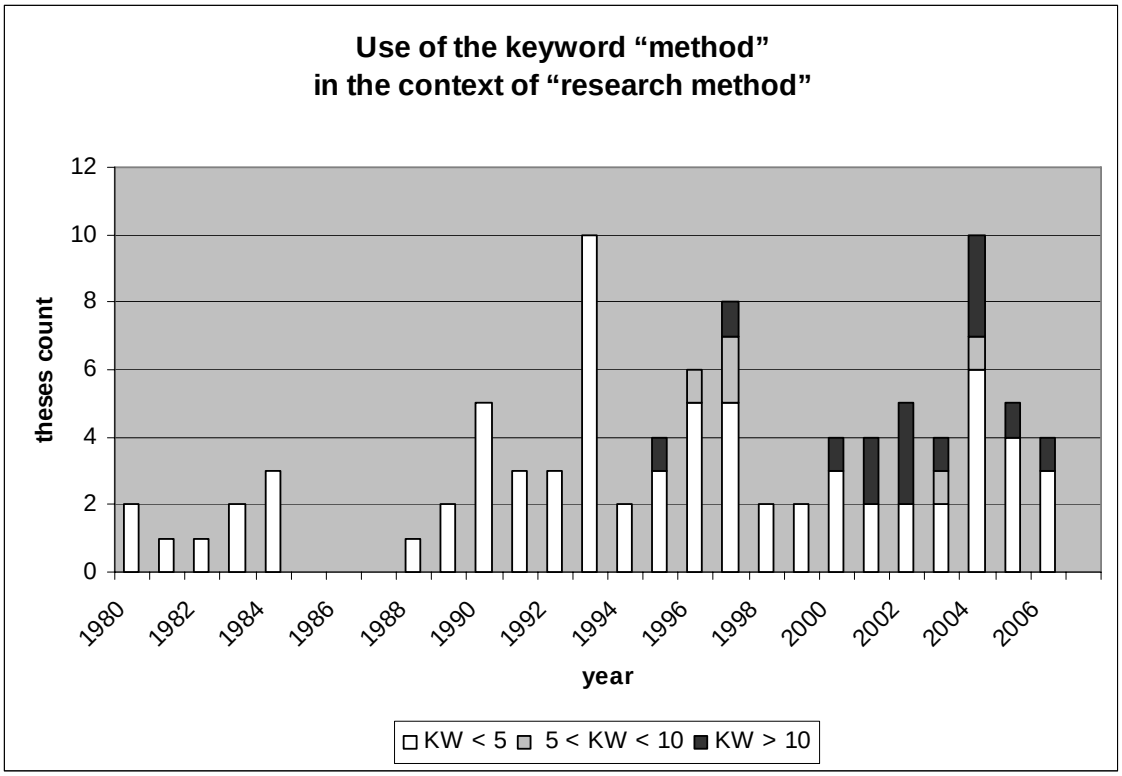

Figure 1: Number of doctoral theses using referring to "research method" less than five times, less than ten times, and more than ten times.

Current methodological framework and taxonomies

As the lack of references to research method illustrates, a representative view of ICS research methods cannot be read explicitly from the theses as a whole. In order to understand ICS research methods, we therefore need to read between the lines and analyze premises implied in each thesis.

The main method for such literary analysis is "text classification" as exemplified by Zelkowitz (1997) and Ramesh (2004). To follow up the KWIC analysis, this study has therefore classified the theses chapter by chapter (772 chapters in 93 theses) according to the categories that Zelkowitz (1997) and Ramesh (2004) suggest (figure 2). The aim of this study is to positively identify and describe research methods used but not

discussed. Short chapters and chapters introducing or summarizing other chapters were excluded.

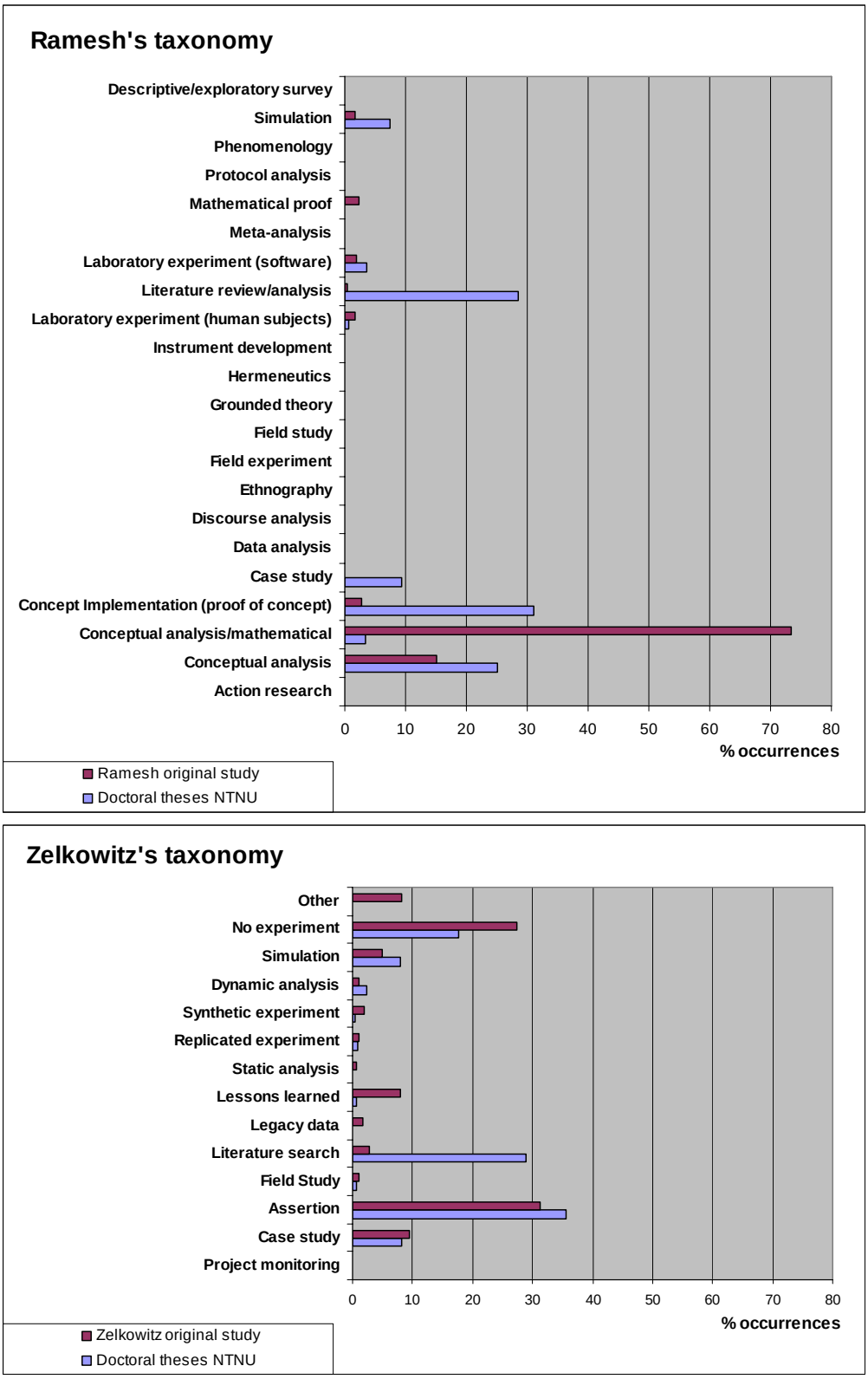

Figure 2: Comparing research methods in doctoral thesis chapters against the findings of Ramesh's and Zelkowitz's original studies.

In broad strokes this analysis found that research literature was used to build up roughly one-quarter to one-third of thesis chapters. Another one-quarter to one-third

used *Concept Implementation* or presentation of system design or implementation as the basis for argumentation. In the Zelkowitz taxonomy these chapters were classified as *Assertions* that designs or implementations are feasible. Around one in ten relied on some form of *Case Study* yielding experience from real or hypothetical problems, and another one in ten relied on some form of *Simulation* or system experiment yielding numerical data. The remaining chapters employed such a variety of methods, ranging from rigid observational experiments to open subjective discussion, that they elude coherent description.

This study's findings also echo core trends in the two original studies. Several categories such as *Ethnography*, *Protocol Analysis*, or *Hermeneutics* were incompatible with the literature in both this and Ramesh's (2004) study. Reusing Zelkowitz's taxonomy reflects the same problem. Put up against the thesis chapters, several categories such as *Static Analysis*, *Project Monitoring*, and *Legacy Data* seemed irrelevant.

However, the most relevant finding is this study's large deviation from both of the two original studies. First, as would be expected of doctoral theses versus journal articles, *Literature Search/Review* is much more widely employed. Almost a third of all chapters are primarily based on literature. Second, compared with the Ramesh study, *Concept Implementation*, *Case Study*, and *Simulation* were found to be more widely used in the set of doctoral theses. Representing a more varied field, such differences might also be due to differences in the material. But some deviations and findings are more disturbing. This study found *Concept Analysis (Mathematical)* only to be relevant in 3.37% of the texts analyzed compared to 73.4% in Ramesh's study. Another example is categories such as *Lessons Learned*, *Static Analysis*, and *Legacy Data*, which in Zelkowitz's (1997) original study registered a low count and in this study were found to be almost completely irrelevant. These deviations cannot simply be attributed to different textual material and will therefore be reviewed in more detail.

A critical review of text classification

Having repeated the experiments of Zelkowitz (1997) and Ramesh (2004) on a different set of ICS texts, some comments are needed about this method of literary analysis. The aim of this study was to positively describe ICS research methods used but not discussed in the theses. However, during and after the analysis, the act of classifying each text unit was found to be much more problematic than anticipated.

First, research methods are not employed in a 1:1 ratio to the text unit chosen. Most thesis chapters, which included several journal articles, could well be linked with several different research methods at the same time: a single chapter could, for example, present the design of a new prototype, literature supporting it combined with some simulation results. In order to classify such a chapter into a single category, the use of different methods must be weighed against each other. This is very problematic for two main reasons. First, how should we compare the quantity of use of different research methods? How do we measure the amount of literature reviewed against the amount of prototype developed against the amount of simulation performed? Second, how should we compare the quality of different research methods? Should *Simulation*, for example, trump *Lessons Learned*, which should then trump *Literature Search*? The answer to these questions would greatly influence the outcome of a 1:1 text classification, but neither Zelkowitz's nor Ramesh's text classification studies describe such weighing or ordering. This study therefore finds that text classification should not assume a 1:1 relationship between thesis chapter/journal articles and research methods.

Second, the taxonomies were very problematic. By themselves and one by one, most of the categories appeared readily understandable, with notable exceptions such as *Concept Analysis*. But when juxtaposed to several thesis chapters, both their individual interpretation and the border between them became blurred. Zelkowitz's categories, for example, contained several tightly related categories such as *Case Study/Field Study* and *Dynamic Analysis/Simulation*. Zelkowitz (1998) shows how such closely related categories can be distinguished against one specific example. However, what this single example does not illustrate is the problem of correspondence between many such individual distinctions. This problem arises *both* when two or more researchers interpret a set of categories against one text *and* when one researcher interprets the same categories across two or more different examples. Thus the act of interpreting the meaning of individual categories and their distinction (and hence the categories themselves) quickly become dependent on both the analyzers' subjective judgment and the texts that have been analyzed.

Third and most problematic, a large portion of the thesis chapters poorly matched any individual category and neither of the two taxonomies' overall design. The research presented was predominantly engineering projects where the researcher himself participated in the constructive processes analyzed. Zelkowitz's taxonomy is designed to capture observational experiments and has no categories suited to discussions about system design or construction. Similarly, Ramesh's taxonomy spans methods from a wide academic range. However, only *Concept Implementation* and the general *Concept Analysis* seemed to fit system engineering. Thus neither taxonomy captures important methodological nuances in ICS engineering.

To conclude, even the best methodological taxonomy juxtaposed to a large body of works will need to stretch and spread its interpretations of its categories in many different directions. If this process of qualitative interpretation is not disclosed, this will inevitably skew any numerical findings.

Exemplifying methodological trends

To describe tacitly developed methods, we would ideally need a method of literary analysis that: lets the theses speak for themselves about a topic that they do not address directly; compresses these findings in a distributable format; and is not skewed by the researchers' own views during the analysis process. Unfortunately, no such method exists. The analytical tools we use to explicate the implied, compress it, and then present it will inevitably affect our findings. The best way to address these shortcomings in terms of objectivity is to reveal their subjective premises honestly.

One such approach is to subjectively select and analyze important methodological trends across a few examples: trend exemplification. Trend exemplification has several benefits. First, trend exemplification can base itself on the research methods actually used in the examples, not a predefined taxonomy. This enables the researcher to tailor the focus and categories more freely to suit the methods implicit or explicit in the material. Second, trend exemplification should be transparent in regard to both subjective approach and evaluation. While quantitative studies might hide subjective, qualitative assessments behind numerical values, trend exemplification does not portray subjective interpretations as objective findings. Third, a sample of a few studies is much more manageable than several hundred chapters or articles. This conserves research time and enables recipients to review the sources and conclusions. However, using a smaller amount of material and a custom-built methodological taxonomy in no way guarantees the study's representability. The recipients must judge for themselves the scope of the study's methodological findings.

This study therefore concludes with a trend exemplification analysis based on four theses: Lindseth (2002), Thorbjørnsen (1995), Digernes (1982), and Opdahl (1992). These theses were selected as examples of a trend here referred to as "ICS engineering," which in prior phases of analysis was perceived as a principal methodological trend in the 93 theses. The first two theses were selected as examples that conform to a classic and commonplace perception of ICS engineering. The last two were selected as less prototypical examples of this tradition. The examples were analyzed to show how they are related methodologically and can be understood as employing a related research method: ICS engineering.

Two typical examples of ICS engineering

An ultrasound system. The first example presented is Lindseth (2002), who study ultrasound systems. "The thesis is part of the ongoing research activity in ultrasound-guided surgery" (ibid.: 7) and focuses on visualization, measurement, and navigation based on ultrasound data. The problem area that the system is to be applied within is medicine and surgery, and the system is comprised of both specialized hardware for obtaining ultrasound input and specialized software to process the data.

A database system. The second example is Thorbjørnsen (1995), who study database systems. The thesis proposes "a multi-site [database] architecture for achieving very high system availability" (ibid.: 153) using a new declustering strategy called Q-rot (ibid.: abstract). The first chapter, "Motivation," presents the need for such database systems in telecom (ibid.: 1–3). The thesis then discusses existing declustering strategies, technical solutions, and present design and partial implementations of a new and better multi-site database.

Although both examples discuss their methodological choices extensively, both discussions are concerned with either the end result's applicability or more detailed technical aspects of their methods. Neither thesis makes explicit their overall methodological approach. Still, these two theses are not hard to recognize as ICS engineering projects, as they conform to a commonplace conception of what computer engineers do. The two examples thus imply some tacit methodology that needs to be described in order to be better understood.

First, both examples work with solutions and designs *implementable* in hardware and software. The medium of traditional hardware and software systems conforms to commonplace conceptions of Information and Computer Systems. Second, both projects develop new versions of systems. The two projects engineer solutions and only devote time to problems that can be described as solutions. Any analytical process (process that takes things apart) is subsumed in this *constructive* outlook. Third, the constructive work is directed out toward societal needs in medicine and telecoms. The projects goals are *applied* knowledge and are justified by their usefulness (cf. Hevner 2004). Lastly, the main source of insight is gained not from observing systems but from participation in their construction. The researcher himself is the brains behind many of the ideas and solutions discussed, and the conclusions drawn are therefore likely to favor the researcher's own work and approach. Combined with a preference for applicable, useful results, the researcher has a personal stake in the discussion outcome and might wish to present new solutions as success stories. Hence, the researcher's role should be understood as *subjective participation*, not objective observation.

Two alternative examples of ICS engineering

Typical ICS projects that clearly conform to a commonplace conception of ICS systems and development confirm overall suspicions but tell little about tacit practices not

already known. Examples of ICS projects that do not readily fit common stereotypes can, however, confront preconceived, tacit assumptions more abruptly and thus bring them to the surface. Therefore, two examples that were more difficult to assess methodologically were analyzed.

The fishing boats with fishermen system. Digernes (1982: viii) "deals with economical evaluation of fishing vessel operation and design." The project has an applied focus to increase the efficiency of the Norwegian fishing fleet. The project also takes a constructive and participatory approach to the problem, aiming to contribute new solutions. Problems that can be solved are pursued, while problems that cannot are left behind. Still, this project is not readily interpretable as an ICS project today, even though it was officially presented as such 26 years ago. Why? Because a contemporary reading of this thesis tests an important boundary of ICS engineering: the concept of system.

While the thesis' solution domain is within the frame of engineering, it is not necessarily within the frame of ICS. The fishing boat's wooden hull and fuel-consuming engine is not what is meant by "hardware" and "software." Furthermore, the fishermen controlling the boats are guided by economical and practical needs. Industrial fishing might not be viewed as systemic at all but as socioeconomic. Hence, the fishing boats with fishermen clearly divert from common notions about what an ICS system is.

Apart from lacking traditional hardware and software, one of the biggest problems in recasting the fishing boats with fishermen as an ICS system is that it is difficult to clearly delineate it as a whole, as one system. Wooden hulls and engines seem too separate from that of fishing economics and practices. Thus it is not the technical solutions that unite the thesis but the area of application, the problems. This research premise breaks with both Lindseth (2002) and Thorbjørnsen (1995). The first two examples create coherence with one coherent, interconnected set of technical solutions, a unified technical system. Coherence in the development process is ensured by this unified system. Digernes (1982) presents instead several solutions indirectly connected by their area of application but not directly connected by their medium of implementation. This example's diverse and foreign solution media thus seem to divert from the tacit premise of ICS engineering projects: a traditional ICS media and internal coherence through holistic solutions.

The development process system. Opdahl (1992) also exemplifies important methodological choices associated with "the system." This thesis focuses on the development process and "how the performance of information systems [can] be predicted and improved during development" (ibid.: 5). The composition of the thesis illustrates how the development process is framed as a technical system. In chapter 2 the thesis begins with a broad problem area description of "the practice of developing and maintaining computerised information systems" (ibid.: 27). Chapter 3 narrows this problem area to the solution area of computer-aided software engineering (CASE) systems, i.e. tools for system developers. The remainder, chapters 4–13, is a technical presentation and discussion of CASE systems.

First, the transition from chapters 2 to 4 illustrates how this project pursues problems that can be implemented in traditional hardware and software and leaves other problems behind. Solution orientation tacitly functions as a filter. At first, only implementable problems are selected, and then these problems are recast as solutions in the implementable medium. In all four examples, though varying in strength, *solvable* and *implementable* are methodological criteria for which problems are viewed as relevant.

Solvable and implementable as a relevance criterion can be interpreted as positivism: there is an underlying assumption that world phenomena/problems can be described objectively in formal and logical systems/solutions. However, positivism is usually used derogatory to criticize researchers for being naïve. Also, positivism is not primarily associated with applied science. The driving force behind positivism is a search for truth, not applicability and usefulness. Optimism might therefore be a better term to describe this form of solution orientation. Research questions are not formulated skeptically ("Can X be solved?"), but optimistically ("What is the best solution for X?"). The assumption that a solution exists is tacit and optimistic.

Second, system development is a process where problems, practices, and actions are performed by humans interacting with and through computer systems. Looking at system development, one could expect the human developers to be cast as the main actors in this process. However, the focus is not on the human developers, and the lack of sentences having "developer" as their subject illustrates this. Instead the focus is on the development process itself: the development process is "the system" and thesis subject, and the developer is just another object of this system, a brick in the wall.

Casting predominantly human practices as a technical, asocial system is methodologically problematic. Ethically, studies actively designing systems partly consisting of human actors should actively consider the human subjects' interests. The fruitfulness of such top-down system design is also questionable. Humans are known to act "irrationally" and thus may render a CASE tool conforming to a rational development process useless.

Summary of ICS engineering

None of the four examples can be said to lack methods. They all use a structured, recognizable approach. And even though the problem domains differ greatly, the studies clearly conform to the same methodological tradition: ICS engineering.

As a research method, ICS engineering is strongly solution oriented. The aim of ICS engineering is to develop applicable, useful solutions. This applied goal is propped up by an optimistic outlook that filters out problems that cannot be solved. While working with solvable problems, the active process of designing a solution is central. The researchers participate in these constructive processes and therefore act subjectively when they later analyze their own and others' rival or allied technical concepts. Internal coherence in the research project is primarily achieved through a holistic solution, with the exception of Digernes (1982).

Conclusion

From the review of methodological discussions in ICS, the KWIC case analysis, the research method categorization, and the trend exemplification, the following conclusions can be drawn.

First, methodological discussion in ICS is still most often implied. The KWIC analysis shows that although discussions about research methodology are more prevalent today than 15 years ago, most of the theses studied still use methods tacitly and do not discuss them explicitly.

Second, the methodological discussion in ICS is not representative of the entire field. Proposed taxonomies (Ramesh 2004; Zelkowitz 1997) do not adequately capture the characteristics and nuances of ICS engineering methods. There is a discrepancy between some methods discussed explicitly and methods used implicitly. Future methodological work should therefore: address this discrepancy; focus more on the methods actually in use in ICS rather than methods in other academic disciplines; and

expand the focus beyond observation and experimentation to include participation and construction. The tacitly developed methods in ICS have produced many ground-breaking and useful results and deserve our attention and recognition.

Third, although trend exemplification is an imperfect methodological approach, this study finds that more empirically based methods should be used to enlighten the ICS methodological debate. The literature about ICS methodology is dominated by normative studies simply asserting a framework for ICS methods or studies counting categories in a literary sample. Trend exemplification might not be the best way forward, but methodological debates in ICS could benefit from the use of empirically based, qualitative methods.

Lastly and as a lead for future work, studies of ICS engineering methodology need to address how solutions are constructed. Particularly, the language of design and implementation should be investigated. As researchers familiarize themselves with their languages of design and implementation, these languages become ever more "natural" and translucent. Understanding how and why ICS researchers use different design and programming languages will tell us more about the tacit, constructive practices that ICS research has come to take for granted.

References

Basili, V., Selby, R., and Hutchens, D. 1986. Experimentation in Software Engineering. *IEEE Transactions on Software Engineering* **12**(7), 733-743.

Baskerville, R., and Wood-Harper, A. 1998. Diversity in information systems action research methods. *European Journal of Information Systems* **7**, 90-107.

Conradi, R. 2008. Doctoral degrees at IDI, NTNU, 1970-2007. Retrieved 15.9.2008 from http://www.idi.ntnu.no/grupper/su/phd-total-list.html.

Diaconescu, R. 2002. Object Based Concurrency for Data Parallel Applications: Programmability. *Department of Computer and Information Science*. Trondheim: NTNU.

Digernes, T. 1982. An analytical approach to evaluating fishing vessel design and operation. *Department of Computer and Information Science*. Trondheim: NTNU.

Glass, R. 1995. A structure-based critique of contemporary computing research. *The Journal of Systems & Software* **28**(1), 3-7.

Hevner, A., March, S., Park, J., and Ram, S. 2004. Design Science Research in Information Systems. *MIS Quarterly* **28**(1), 75-105.

ISERN 2007a. ISERN basic terminology. Retrieved 24.4.2007 from http://www.cs.umd.edu/projects/SoftEng/tame/isern/isern.definitions.html.

ISERN 2007b. ISERN Manifesto. Retrieved 24.4.2007 from http://isern.iese.de/network/ISERN/pub/isern-beschr.html.

Lindseth, F. 2002. Ultrasound Guided Surgery: Multimodal Visualization and Navigation Accuracy. *Department of Computer and Information Science*. Trondheim: NTNU.

Opdahl, A. 1992. Performance Engineering During Information System Development. *Department of Computer and Information Science*. Trondheim: NTNU.

Orlikowski, W., and Baroudi, J. 1991. Studying Information Technology in Organizations: Research Approaches and Assumptions. *Information Systems Research* **2**(1), 1-28.

Ramesh, V., Glass, R., and Vessey, I. 2004. Research in computer science: an empirical study. *Journal of Systems and Software* **70**(1-2), 165-176.

Shaw, M. 2002. What makes good research in software engineering? *International Journal on Software Tools for Technology Transfer* **4**(1), 1-7.

Shaw, M. 2003. Writing Good Software Engineering Research Papers. In *International Conference on Software Engineering*, (pp. 726-736)

Sjøberg, D., Dybå, T., and Jørgensen, M. 2007. The future of empirical methods in software engineering research. In L. Briand and A. Wolf (eds.) *Future of Software Engineering (FOSE '07)* (pp. 358-378): IEEE-CS Press.

Sjøberg, D., Hannay, J., Hansen, O., Kampenes, V., Karahasanovic, A., Liborg, N., and Rekdal, A. 2005. A Survey of Controlled Experiments in Software Engineering. *IEEE Transactions on Software Engineering* **31**(9), 733-753.

Tichy, W. 1998. Should Computer Scientists Experiment More? *Computer* **31**(5), 32-40.

Tichy, W., Lukowicz, P., Prechelt, L., and Heinz, E. 1995. Experimental evaluation in computer science: A quantitative study. *The Journal of Systems & Software* **28**(1), 9-18.

Torbjørnsen, Ø. 1995. Multi-Site Declustering Strategies for Very High Database Service Availability. *Department of Computer and Information Science*. Trondheim: NTNU.

Vessey, I., Ramesh, V., and Glass, R. 2005. A unified classification system for research in the computing disciplines. *Information and Software Technology* **47**, 245-255.

Zannier, C., Melnik, G., and Maurer, F. 2006. On the success of empirical studies in the international conference on software engineering. In *International conference on Software engineering*, (pp. 341-350)

Zelkowitz, M., and Wallace, D. 1997. Experimental validation in software engineering. *Information and Software Technology* **39**(11), 735-743.

Zelkowitz, M., and Wallace, D. 1998. Experimental Models for Validating Technology. *Computer* **31**(5), 23-31.