Reconsidering the Notion of "Requirements Engineering" for Enterprise Systems Selection and Implementation

Michael Rosemann
Queensland University of Technology
Brisbane, Australia
Email: m.rosemann@qut.edu.au

Iris Vessey
Queensland University of Technology
Brisbane, Australia
Email: i.vessey@qut.edu.au

Ron Weber
Monash University
Melbourne, Australia
Email: ron.weber@infotech.monash.edu.au

Boris Wyssusek
Queensland University of Technology
Brisbane, Australia
Email: b.wyssusek@qut.edu.au

Abstract

Notwithstanding the pervasiveness of enterprise systems, their selection and implementation is still being perceived as problematic if not risky. Research in the tradition of requirements engineering argues that its insights will ameliorate this condition. Yet, the context of enterprise systems has outmoded requirements engineering from the viewpoints of both method and practice. Theoretically, analysis of the presuppositions of requirements engineering has identified a methodological difference that proffers a cogent explanation of the empirical findings regarding today's practices. The conclusions are of immediate relevance for organisations, since they present an updated view of current practices of selection and implementation of enterprise systems.

Keywords

Enterprise system selection and implementation, requirements engineering, alignment

INTRODUCTION

Despite enterprise systems having been adopted successfully by an ever increasing number of organisations, the selection and subsequent implementation of such systems remains a high-risk endeavour, as shown by a non-negligible number of disasters with their implementation. In general, it appears that enterprise system implementation projects tend to run late, exceed budget—and sometimes even fail completely (e.g., Appleton, 1997; Scott and Vessey, 2002).

Research aiming at the understanding of the causes that led to the above mentioned problems with enterprise system selection and implementation projects has revealed a great number of contingencies that supposedly impact the outcome of such projects. Most popular are so-called "critical success factor models" that exhibit correlations between a rather small number of variables. Those variables are frequently concepts such as "system quality," "information quality," "information use," and "organisational impact" (e.g., DeLone and McLean, 1992)—largely devoid of empirical measures. Hence it should not come as a surprise if some conclude that "studies of ERP's critical success factors offer few insights" and "their contribution to understanding ERP implementation is limited" (Robey et al., 2002, p. 21). Others acknowledge that reducing success and failure of information systems development to law-like contingencies does not give consideration to the socio-cultural nature of information systems development (e.g., Lyytinen and Hirschheim, 1987; Introna and Whittaker, 2002).

With respect to the selection and implementation of enterprise system packages, contemporary research frequently claims that many implementation failures are likely due to insufficient requirements engineering (e.g., Rosemann et al., 2004) or due to difficulties arising while using specified requirements in the selection and implementation process (e.g., Rolland and Prakash, 2000). These claims usually rest on presuppositions that eventually view information systems development as a "planned, deliberate activity—bounded in time and

carried out in a systematic and orderly way" (Bansler & Havn, 2003, p. 51). In other words, information systems development has been understood as a method-based process, to an extent that "the modern concept of method has been so strongly impressed on our thinking about systems development, that the two concepts, information systems development and information systems development method, are completely merged in systems development literature" (Truex et al., 2000, p. 56).

In contrast, empirical findings question the very idea of method-based information systems development, since "methods are often unsuitable for some individuals (Naur, 1993) and settings (Baskerville et al., 1992). The deployment of the same methods in similar settings may yield distinctly different results (Turner, 1987). Developers may claim adherence to one method while ignoring this method in actual practice (Bansler and Bødker, 1993)" (Truex et al., 2000, p. 54). Consequently, FLOYD (1992) criticises the discipline's "view of methods as rules laying down standardized working procedures to be followed without reference to the situation in hand or the specific groups of people involved" (p. 86).

This suggests that while research has been preoccupied with method, it has, simultaneously, neglected (or ignored) the "*a*methodical" aspects of information systems development, as explained by TRUEX et al. (2000): "When the idea of method frames all of our perceptions about systems development, then it becomes very difficult to grasp its non-methodical aspects" (p. 74). In a similar vein, BANSLER and HAVN (2003) note that these aspects "become marginalized and practically invisible, e.g. how ISD is subject to human whims, talents and the personal goals of the managers, designers and users involved" (p. 51).

Our research is focused on gaining a deeper understanding for the actual practices of evaluating, selecting, and implementing enterprise systems. We utilised an exploratory field study based on semi-structured interviews as our research method. In light of the empirical findings of our ongoing research, we have come to question many claims made by contemporary literature on the subject of requirements engineering for enterprise system selection and implementation. Our findings revealed that the method-based understanding of requirements engineering for enterprise systems selection implementation cannot account for many phenomena we were able to observe. The widespread ignorance of factors that are beyond an ideal methodical approach does not do justice to the reality of such a complex process, heavily influenced by economic, political, and other social contingencies (e.g., Markus, 1983; Truex et al., 2000). Thus, we believe it is time to take a fresh look at prior research. Our focus on requirements engineering for the selection and implementation of enterprise systems is motivated by (1) the significance generally being attributed to this early phase in the entire information systems development lifecycle—an insight that can be traced back to BOEHM's (1981) study of the economics of software engineering, and by (2) research claiming that insufficient requirements engineering is likely to be responsible for the failure of enterprise system selection and implementation projects.

The remainder of the paper is structured as follows: First, we review the classical understanding of requirements engineering as presented in software engineering and outline underlying presuppositions. Second, we show how the classical understanding of requirements engineering has framed the prevailing conceptualisation of requirements engineering for the selection and implementation of enterprise systems. Third, we present empirical evidence challenging this conceptualisation. Fourth, we reframe the conceptualisation of requirements engineering for the selection and implementation of enterprise systems in light of the empirical evidence presented. Fifth, we conclude with a summary and recommendations for the future.

THE TRADITIONAL NOTION OF "REQUIREMENTS ENGINEERING" AS KNOWN FROM SOFTWARE ENGINEERING

The development of requirements engineering as a dedicated field within the discipline of software engineering can be traced back to the now famous NATO Conference on Software Engineering, where the name "software engineering" was coined and the associated discipline established. Most notably, "[t]he phrase 'software engineering' was deliberately chosen as being provocative, in implying the need for software manufacture to be based on the types of theoretical foundations and practical disciplines, that are traditional in the established branches of engineering" (Naur and Randell, 1969, p. 13). Hence, the term "requirements engineering" inherited the visionary nature of the term "software engineering." Reviewing the course of the development of the discipline of software engineering and of its subfield requirements engineering it becomes obvious that the presupposition underlying classical engineering disciplines had a guiding role in the respective developments.

Typical textbook definitions may refer to engineering as "Creating cost-effective solutions […] to practical problems […] by applying scientific knowledge […] to building things […] in the service of mankind" (Shaw, 1990, p. 15), with the scientific knowledge stemming almost exclusively from formal and natural sciences. Consequently, both software engineering and requirements engineering view information systems as technical hence neutral artefacts that can be studied in their own right—a notion that also expressed in contemporary information systems literature (e.g. Orlikowski and Iacono, 2001).

Understanding organisations as teleological systems, organisational information systems such as enterprise systems are a means to an end—a way of helping to achieve the goals of the organisation. The reality of the organisation and its environment is assumed to be independent from an observer. Consequently, current and desired future states of reality provide the information system developer with objective requirements for information systems development. The task of the developer is to model reality in a way that the information system eventually provides the user with a faithful representation of reality—the assumed prerequisite for the successful use of the information system. In this sense, information systems development and implementation is understood as a technical, i.e., rational and method-based process.

In the tradition of Cartesian thought, it is the rationalist understanding of information systems development—as outlined above—that puts methods and their use in the focus of researchers and practitioners alike. Methods are assumed to provide the means to achieve a given end, e.g., the successful development of information systems. In turn, failure of information systems development is assumed to be largely the result of the use of faulty methods or of the faulty use of correct methods (e.g., Tolvanen, 1998). The importance attributed to rational processes guided and supported by methods finds its expression in the emergence of the field of "method engineering" (e.g., Brinkkemper, 1996). Framed by Cartesian thought, researchers do not reflect upon their focus on methods and their understanding of information systems development as a method-based process. Instead they continuously developed yet another modelling approach (Oei et al., 1982).

Methods are used throughout the entire requirements engineering process, which is frequently conceptualised as consisting of the following activities (e.g. Paetsch et al., 2003), with the first four usually being performed in a chronological order—not excluding iterations of activities or even the entire process:

- *Requirements Elicitation* is concerned with the 'discovery' of requirements. This is usually done by means of various methods, such as interviews with stakeholders, surveys, focus groups, document analysis, observations, or prototyping. The goal of this phase is the establishment of an initial set of requirements.
- *Requirements Analysis* is performed on the basis of the initial set of requirements, the output of the preceding phase. The goal of the analysis is to identify and resolve inconsistencies, to discover missing or incorrect requirements and the like. Hence it may be necessary, and it is very likely, that one has to go back to the requirements elicitation phase in order to overcome the deficiencies of the initial set of requirements. The outcome of the analysis phase should be a complete and consistent set of requirements.
- *Requirements Documentation* is concerned with the creation of various representations of the requirements. Those representations can be informal (text in a natural language, e.g., English), semi-formal (textual and/or graphical, e.g., UML), or formal (via a formal requirements specification language, e.g., ALBERT II). Requirements documentation serves multiple purposes, such as providing (1) a source of reference that can be used, for example, during the evaluation of the product, (2) a means for communication among various stakeholders, (3) input for the subsequent phase of requirements validation, and (4) the means for the ongoing management of requirements and of the requirements engineering process.
- *Requirements Validation* is performed to ensure the quality of the requirements documentation, e.g., its completeness and correctness. At the end of the requirements validation process all stakeholders should agree upon the requirements documentation, i.e., they should agree that the requirements documentation is an adequate description of the information system eventually to be developed. The validated requirements documentation ultimately provides the input for all subsequent phases of the information system development process.
- *Requirements Management* is concerned with the management of the entire requirements engineering process. It should coordinate and keep track of all activities constituting the requirements engineering process, manage the flow of information during the project, and keep track of changes of requirements.

In short, requirements engineering is the collection, consolidation, and documentation of information about the characteristics of a desired future state of reality that will be shaped by the information system eventually to be implemented. Thus the documented vision of a future with the information system in place serves both as a guideline for the development of the information system and as a point of reference for the evaluation of the information system developed. If the documented vision of a desired future state of reality is identical to the then actual state of reality, the information systems development has been a success.

REQUIREMENTS ENGINEERING FOR ENTERPRISE SYSTEM SELECTION AND IMPLEMENTATION

The outcome of software or systems requirements engineering is thus a blueprint for development, and a means against which the artefact can be assessed. For custom-made software it may be hard to distinguish between development and implementation of a system because these phases may appear as a single project. Enterprise systems, however, are developed, sold, and purchased, and then implemented. The software becomes operational, therefore, via different processes and differentorganisational contexts.

The question arises, however, as to what extent the blueprints provided by traditional requirements engineering are applicable in the activities of selection and implementation of enterprise systems. The answer provided by research until now has been that traditional requirements engineering—as in software engineering—is in principle equally applicable for enterprise systems selection and implementation, as it has been in the past for the development of customised information systems.

It has been acknowledged that requirements for development and those for selection differ; while the former should be completely specified, the latter are not. Nevertheless, it is argued that specifications have to be modelled and that true business specifications, i.e., organisational requirements, must be expressed in a generic, vendor independent modelling language (e.g., Soffer et al., 2001). In contrast, it has also been acknowledged that, in the context of commercial off-the-shelf (COTS) software, users may express their requirements in terms of the assumed functionalities of a specific product (e.g., Maiden and Ncube, 1998).

Certain authors have proposed re-conceptualising systems functionality at a high-level of abstraction thereby facilitating the display of "capabilities" and enabling potential users to choose from an array of possible implementation alternatives (i.e., configurations) (e.g., Rolland and Prakash, 2000).. Such an approach would emphasize the role of requirements analyses in communicating system requirements. What could be called the "strong requirements" approach insists that "system and the requirements are represented in the same modeling language" in order to accomplish "alignment" between specifications of organisational needs and of systems functionalities throughout selection and implementation (Soffer et al., 2003, p. 682).

Thus, traditional requirements engineering has been transferred—with some modification—into the context of organisational adoption and adaptation of enterprise systems. Yet, we contend that this transfer is guided by the basic assumptions that have also led requirements engineering for the development of custom information systems: (1) Through requirements engineering, decisions on the future state of the system and consequently of the organisation can be made in the form of rational choices. (2) Requirements engineering identifies the objective information needs of the user organisation. (3) Requirements engineering will eventually arrive at a correct and true description of what the users need and what can be provided by the enterprise system.

These assumptions determine how the requirements engineering process for enterprise systems selection and implementation is now conceptualised. Based on assumption (3), the objective of the selection process is to achieve a complete match between organisational requirements and enterprise system functionality. Accordingly, processes and data should be modelled corresponding to the organisational objectives, and this activity must precede and be completed prior to entering into the selection tasks. Subsequently, during implementation, the requirements blueprint represents the guidelines for what needs to be accomplished. At the same time, because the requirements as determined are a true reflection of the best match between needs and functionality, they are the point of reference for establishing implementation success: meeting the objectives stipulated by the requirements represents success.

Traditional requirements engineering construes technology as neutral artefacts, vis-à-vis organisations (which are distinct from their technologies) and cognitively accessible due to their rational and immediately observable structure. This dualism of technology and organisations means that requirements engineering is the rational determination of organisational "requirements" that eventually have to be met by the "capabilities" of the enterprise system to be implemented. This dualism continues for the assessment of the stipulated requirements and the assumed "capabilities" of software: if requirements are met, then organisation and technology are "aligned," if not, then there is a "misfit" (e.g., Rolland and Prakash, 2000; Soh et al., 2000; Soffer et al., 2003; Wu et al., 2005). Aligning the "capabilities" of the technology and the "requirements" (or "needs") of the organisation, hence avoiding misfits, is the objective of enterprise system selection implementation. Yet "alignment" and "misfit" are rather elusive notions that lack obvious and substantial empirical content (e.g., Strassmann, 2004)—an issue that has spawned a number of research projects and publications dealing with the operationalisation of those notions (e.g., Sia and Soh, 2002; Soffer, 2004; Rosemann et al., 2004).

However, making notions amenable to measurement cannot compensate for a restricted or misleading theoretical outlook. The technology–organisation dualism implies "managerial bias toward rationality and efficiency, thereby imposing limits on the explanatory ability of studies conducted from this vantage point" (Alvares, 2000, p. 1655), postulating a straight forward means–ends relationship that can be worked through rationally and

methodically. Hence, presupposing a technology–organisation dualism necessarily leads to the exclusion of what is commonly referred to as the social world, and can treat it only as (1) a disturbance to a technically perfect project, and (2) as an explanatory device (e.g., lack of management support, lack of user involvement, etc.) when the preconceived goals of a plan have not been achieved according to the expectations of stakeholders.

EMPIRICAL STUDY OF ENTERPRISE SYSTEM SELECTION AND IMPLEMENTATION

The aim of our empirical study was to learn about the actual practices of requirements engineering for the selection and subsequent implementation of enterprise systems, and to identify critical steps within this process. With "critical steps" we mean activities or non-activities (i.e., activities that were not performed) that eventually lead to a 'misfit' between organisational requirements and functionalities of the enterprise system to be selected and implemented.

In contrast to studies that are based on actual selections and implementations, we chose to interview enterprise systems consultants with strong experience either in the pre-sales or the after-sales (implementation) phases of the enterprise systems lifecycle. The motivation behind this choice rests with the fact that enterprise systems consultants have gained deep insights into the intricacies of the selection and implementation of enterprise systems over the course of a rather large number of projects with a wide variety of client organisations. Thus we believe that interviewing consultants provides us with insights and evidence that would take us many years to collect while observing actual selection and implementation processes. Nevertheless, we are fully aware of the limitations of such a 'single-sided' approach.

Method

Via already established contacts to enterprise systems vendors, consultants, and client organisations we have selected 9 enterprise systems consultants with at least 10 years of pre-sales or after-sales (implementation) experience. The consultants were either freelance consultants, or consultants working for large, independent consulting companies who specialized in enterprise system package selection and implementation, or consultants who worked for enterprise system vendors. For the most part, the consultants had gained their experience with large organizations, i.e., those having several thousand employees. The consultants were of various nationalities, coming from Australia, Europe, the Middle East, and North America. Their experience spanned multiple continents and multiple countries.

The interviews were semi-structured. On average they took about 90 minutes with the final 30 minutes being spent in an open discussion or exploring some issues that arose earlier in more detail. Depending on availability, some consultants were interviewed twice, allowing more focused questions. The interviews were either recorded and then transcribed, or notes were made by at least two of the researchers and consolidated subsequently.

Findings

Enterprise systems selection and implementation processes come in two kinds: "Vanilla implementations" with hardly any customisation, and implementations with a huge effort going into the customisation of the enterprise system—"customised implementations." The overall ratio of "vanilla" to "customised" implementations appears to be around 80 to 20. Different implementation methodologies are being applied to the different kinds of implementations

Vanilla implementations have become the de facto standard for enterprise systems implementation in business organisations. Fully customised implementations are in most cases for government organisations.

When describing the enterprise systems selection and implementation process, consultants refer to some more or less 'official methodology' but admit that such a reference is only by analogy and only for explanatory purposes. The actual implementation may resemble superficially such an 'official methodology,' but details are ultimately determined by the knowledge of the consultant and the situation at hand.

Vanilla Implementations:

- For vanilla implementations, the process of enterprise system selection can hardly be considered to be method-based in the traditional sense. In general, the client organisation engages a consulting company to support the selection process. Due to the complexity of enterprise systems, a detailed analysis of requirements does not take place. Rather the consulting company presents the client organisation with a list of about 20 to 30 selection criteria, which denote the major differences between alternative enterprise systems.
- The actual selection is heavily influenced by consultants' knowledge of the respective systems. Thus, the selection criteria are by no means objective. Choosing a consulting company for the selection of an

enterprise system largely determines the eventual outcome of the selection process. Consultants' experiences enable them to propose the 'best' alternative (according to their own knowledge) almost in an ad hoc fashion. If a more formal selection process is performed, requests for offer tend to be skewed towards some preferred enterprise system. The preference is usually not based on a rational evaluation a priori, but rather on political reasoning such as relationships with vendors, peer pressure (e.g., competitors are using a certain enterprise system), market share of the vendors, and the like.

- Pre-sales activities of vendors and consultants are characterized by the competitiveness of this stage. It is well-known and accepted in the industry that pre-sales consultants paint a rather ideal picture of the respective enterprise system. Pre-sales is about getting over the first hurdle, i.e., to become short-listed. During sales, optimism gives way to reality in that offers are generally requested subject to a certain budget. Sales activities are mostly concerned with determining the eventual configuration—at a rather coarse level, but one that allows the allocation of funds to the respective activities. High-level and special requirements become part of the contract. During sales the important activities continuously shift from sales people to implementation consultants, since sales people have a broad, but only superficial, knowledge of the features of specific enterprise systems.
- Implementation consultants are the driving force during implementation. The eventual configuration, that is, the actual features of the implemented enterprise system, are determined and are subject to local contingencies that surface only during the implementation process. Implementation basically means the successive exclusion of alternatives, beginning with the definition of the scope of the project, e.g., human resources, and concluding with the selection of, e.g., specific functions or data formats.
- Participation of the client organisation during the implementation is sometimes limited. To control the implementation process, consultants provide representatives of the client organisation with choices, rather than eliciting requirements. Thus, detailed requirements are determined implicitly by the consultants' knowledge of the enterprise system. In general, the representatives of the client organisation provide input by answering questions posed by consultants, which helps them to eliminate alternative configurations.
- The successive elimination of alternatives takes place in a number of iterations, with the first iteration dealing with the exclusion of whole parts of the system and the last iteration dealing with the minute details of the data structure. Thus, a detailed picture of the final configuration of the enterprise system implementation does not exist a priori. Consultants have an overview and experience with various implementations, which enables them to handle the details as soon as they become an issue.

Customised Implementations:

- For fully customised implementations of enterprise systems, the selection and implementation process is quite different from the one for vanilla implementations.
- Pre-sales still paints an optimistic picture, but the selection process is formal and accompanied by a detailed requirements analysis, usually performed by the organisation and supported by consultants. During the selection phase, vendors and consultants respond to the detailed requests for offer. Consulting companies involved in the selection process are usually excluded from the implementation process in order to avoid conflict of interest during the selection of an enterprise system.
- Before the actual selection, vendors and consultants are generally asked to provide some proof of concept and proof of their capabilities to complete the implementation project successfully, including being able to meet the requirements.
- The implementation process follows fairly closely some 'official methodology,' yet the driving force during implementation is the client organisation. Thus, the client organisation is heavily involved in the entire implementation process.
- Fully customised implementation is not driven by the exclusion of alternatives but by the goal of meeting every single requirement of the client organisation. Consequently customising does not only mean to configure the enterprise system in a certain way but frequently also to implement extensions and enhancements 'on the fly.'
- The need for fully customised implementation is frequently explained as being due to legal and regulation-based design of the activities of the given organisations, such as government organisations.; that is, requirements of those organisations are already pre-specified by laws and regulations.

Besides explaining the need for vanilla implementations and customised implementations largely on the basis of a distinction between two types of organisations, further issues surfaced during the interviews that need to be taken into consideration when theorising about enterprise system selection and implementation.

Independent of organisation type, fully customised implementations are also influence by the *maturity of the product* with respect to the functionality to be implemented. For example, if an enterprise system is to be implemented in an industry that so far has not been covered by the enterprise system vendor, experiences with the configuration of the product in that industry are not available. It Also largely unknown is the extent to which the functionality of the enterprise system will serve the needs of organisations in this industry. Hence, fully customised implementations are necessary if an immature product is to be implemented. In such a case, the implementation might even include ad-hoc software development by the vendor in order to provide important functionality. Comprehensive mapping of business processes is performed and fed back to the developers of the enterprise system. Frequently, the vendor and/or the sales-organisation (e.g., a consulting company) cover the additional costs—in the expectation of further implementations in that industry.

The *maturity of the market* impacts the way in which the selection and implementation of enterprise systems is performed. Whereas in a mature market the competencies of vendors and consultants as well as the functionalities provided by an enterprise system are generally acknowledged, in immature markets client organisations are more sceptical. This finds its expression in more elaborate and formal selection processes and a more rigorous monitoring of the implementation process. In immature markets, vendors and consultants are more frequently asked to provide proof of concept and proof of their capabilities to deliver the promised results.

RECONSIDERING REQUIREMENTS ENGINEERING FOR ENTERPRISE SYSTEM SELECTION AND IMPLEMENTATION

Information systems are widely regarded as an "enabling factor." With this understanding it is no longer possible to separate the "capabilities" of information systems from the "requirements" of an organisation. Organisational requirements and 'capabilities' of enterprise systems are constructed during the processes of system selection and implementation: requirements are "emergent, in the sense that they do not already exist, but rather emerge from interactions between the analyst and the client organisation" (Goguen, 1992). This characterization holds for the selection process as well because it is usually also supported by consultants.

With the advent of commercial off-the-shelf enterprise system packages it no longer makes sense to state detailed requirements a priori—without considering the enterprise system that will eventually provide the solution to the problem at hand. "Well, in the mid 80's, early 90's I went thru an exhaustive requirements gathering business process mapping process to diagram all that out and forced my client to look at diagrams that they didn't understand. […] Stop trying to model your business and find a package that matches that. Don't do that. You can't do that in the time frame that you want to do it. Use my counsel to say, o.k., you are in health care. Here are the 25 differentiating factors" (consultant).

After the selection of an enterprise system, the vendor of the system basically holds a monopoly. Thus, organisations have to perform *reverse alignment*. Reverse alignment "involves the adoption of enterprise systems (ERP) on the basis that they provide a common upgrade path because the supplier continually enhances the technology and a common platform that allows extensive interconnection across a large, complex organisation. Adopting companies have chosen to redesign their business around the IT systems" (Sauer and Willcocks, 2004).

Understanding information systems as an "enabling factor" also means that the implementing organisation must be, and usually is, willing to change. Due to the overall complexity of the implementation and organisational change process it is impossible a priori to determine the outcome of the processes. For example, new information system functionality will require new organisational arrangements, which in turn may require new information system functionality—eventually provided by a different configuration or, occasionally, by an add-on. The interplay between organisational arrangements and technological configurations is highly situational (e.g., Suchman, 1987; Johnston and Milton, 2002) and not guided (and not guidable) by formal methods, but is ultimately determined by local knowledge and practices (e.g., Ciborra and Lanzara, 1994). Enterprise system implementation is learning by doing (Fleck, 1994), and can be conceptualised as a process of organisational learning (e.g., Scott and Vessey, 2000). Besides the changes an organisation undergoes due to enterprise system implementation, the organisation also changes due to various other contingencies, such as market conditions, other ongoing projects, personnel turnover, organisational learning, etc. Hence, with enterprise system implementations taking generally many months or even years from initiation to completion, they are faced with an unstable organisational environment. This instability necessarily reflects back on the enterprise system implementation process.

Further, current implementation conditions in businesses operating in mature enterprise system markets do not support the idea of an a priori determination of requirements. Current implementations are largely guided by the imperative of avoiding customisation, thus of avoiding uncommon requirements: "And [the members of the implementation team of the client organisation] heard it from their executives loud and clear that 'we are not going to customise because we are not going to continue to put investment into our systems […].' So they have heard the no customisation message loud and clear. So from a chain of management perspective, clients are

almost scared to put requirements in front" (consultant). Thus, the client organisation is not in control of the requirements, but vendors and consultants are: "Where the gap is typically created is the customer's expression of requirements without regard to package capability. If I can be the one describing detail and having the client tell me yes or no, then I don't have a gap. I simply have a package feature that the client may or may not want" (consultant).

Yet for government organisations the process is different: "Don't tell the […] government what the package does and that what they want to do isn't relevant. So we are implementing [an enterprise system] right now for the […] government and we have a very different approach to it. Because what they say wins." "[I]f the package happens to support [what they want], then that's wonderful. If not, we have some work to do" (consultant). But even if traditional requirements engineering for the selection of enterprise systems packages is attempted, e.g., in the case of government organisations that are forced to go through a formal selection process, it is impossible to follow the traditional model. Enterprise system packages are far too complex to perform a full-fledged analysis of their features. From a practical point of view, a mapping of the description of the functionality of an information system with the description of the functional requirements of an organisation is not feasible due to the sheer number of functions and functional requirements (Rolland and Prakash, 2000). Hence, significant deviations from traditional requirements engineering are always necessary (Lauesen, 2004).

Enterprise systems do not really exist before they are implemented. This is due to their nature being an instance of configurational technology: "Configurations comprise assemblies of technological and non-technological components, including human factors, built up to meet local contingencies" (Fleck, 1993). Most likely, many of the local contingencies have never been reflected upon before the implementation of an enterprise system. During implementation, implicit knowledge of employees is 'turned' into espoused theories (Argyris and Schön, 1974), which more or less reflect the actual contingencies. Hence, trial and error are necessary characteristics of the implementation process (Fleck, 1994). Only on an abstract level—detached from the intricacies of the actual implementation—is it possible to describe the implementation process a priori, for example via so called "business blueprints."

The gap between research and practice of enterprise system selection and implementation is also exhibited by differences in terminology. In practice, the terms "requirements engineering," "alignment," and "misfit" are hardly used, and if they are used, then their meanings are quite different from the ones found in academic literature. As outlined above, there is good reason not to apply the term "requirements engineering" in the context of the selection and implementation of enterprise systems because the differences between "requirements engineering" in the traditional sense and 'requirements engineering' for the selection and implementation of enterprise systems are rather significant.

With respect to the research–practice gap it might be concluded that the statement below not only holds for that particular context but also for contemporary research on the selection and implementation of enterprise systems as well: "Many of our academic researchers in [software engineering] and [requirements engineering] continue to tackle 'interesting and researchable' [requirements engineering] problems, often without knowledge of relevant issues and problems in practical life. More empirical research is needed to determine what the problems are and what types of solutions may become applicable in practical life" (Bubenko, 1995, p. 162; see also Glass, 1994; Fenton et al., 1994).

CONCLUSION

The case of enterprise systems indicates that some concepts from the early days of business computing and systems development have ceased to have any import on the actual processes of deployment of large-scale information systems. First, the activities of development and implementation are nowadays separated by the task of system selection, which is essentially economic in nature. Second, the standardisation, as well as the scope and potential diversity of commercially-available software seems to have removed many if not all concerns that used to be pertinent in times when custom development was the rule. Hence, we have reached again the point of departure of systems development and software engineering, i.e., requirements engineering in its traditional sense. The engineering approach to systems development might already have been based on presuppositions that could have some legitimisation within a limited context, but were most likely unsuitable for conceptualising systems development and implementation in its entirety.

These fundamental notions of requirements engineering have, however, been applied to analysing and improving processes of selection and implementation of enterprise systems. Although, it had been acknowledged that there is a difference between custom development and a configurable commercial application, the perspective on the deployment did not change in principle. Researchers in this area in particular insisted on the necessity and utmost importance of requirements engineering as the way for achieving rational selection decisions and effective implementations. Concepts such as "alignment" (imported from management theory) and "misfit" were introduced for stating the objective of requirements engineering in an entirely changed environment.

So far, it has been impossible to validate empirically the assumptions on which this strand of research has been built. On the contrary, implementation practitioners have consistently acknowledged that many of the traditional notion of requirements engineering are unknown in selection and implementation of enterprise systems. This fact nullifies the desired legitimisation for traditional requirements engineering in the context of the selection and implementation of enterprise systems. Requirements engineering in the traditional sense certainly takes place in the development phase of enterprise systems (e.g., for adaptation to a particular industry), but is largely irrelevant for the user organisation and the implementation partner. In brief, in the case of enterprise system selection and implementation, requirements are neither analysed nor engineered. Our empirical findings clearly support an understanding of 'requirements engineering' for enterprise system selection and implementation as being *a*methodical—in contrast to the traditional understanding of requirements engineering, which are regarded as *methodical*.

The results of our study indicate that classical requirements engineering approaches and existing information system development models cannot explain the current practice related to the selection and implementation of large enterprise systems. Commercialisation of software development and implementation, or in other words the reshaping of the computer and services industry, has redefined organisational information systems, and this should motivate a new direction for research.

REFERENCES

Alvares, R. (2000) "Examining an ERP implementation through myths: a case study of a large public organization" in *Proceedings of the Americas Conference on Information Systems*, Long Beach, California, 1655–1661.

Appleton, E.L. (1997) How to survive ERP, *Datamation*, 43, 3, 50.

Argyris, C. and Schön, D.A. (1974) *Theory in practice: increasing professional effectiveness*, Jossey Bass, San Francisco.

Bansler, J.P. and Bødker, K. (1993) A reappraisal of structured analysis: design in an organizational context, *ACM Transactions on Information Systems*, 11, 2,165–193.

Bansler, J.P. and Havn, E.C. (2003) "Improvisation in action: making sense of IS development in organizations" in G. Goldkuhl, M. Lind and P.J. Ågerfalk (eds.) *Proceedings of Action in Language, Organisations and Information Systems (ALOIS)*, Linköping, Sweden, 51–64.

Baskerville, R., Travis, J. and Truex, D.P. (1992) "Systems without method: the impact of new technologies on information systems development projects" in K.E. Kendall, K. Lyytinen and J.I. DeGross (eds.) *Transactions on the impact of computer supported technologies in information systems development*, Elsevier, Amsterdam, 241–260.

Boehm, B.W. (1981) *Software engineering economics*, Prentice-Hall, Upper Saddle River.

Brinkkemper, S. (1996) Method engineering: engineering of information systems development methods and tools, *Information and Software Technology*, 38, 4, 275–280.

Bubenko, J.A. jr. (1995) "Challenges in requirements engineering" in *Proceedings of the IEEE International Symposium on Requirements Engineering*, York, England, 160–162.

Burrell, G., and Morgan, G. (1979) *Sociological paradigms and organizational analysis. Elements of the sociology of corporate life*, Heinemann, London.

Ciborra, C.U. and Lanzara, G.F. (1994) Formative contexts and information technology: understanding the dynamics of innovation in organizations, *Accounting, Management & Information Technology*, 4, 2, 61–86.

DeLone, W.H. and McLean, E.R. (1992) Information systems success: the quest for the dependent variable, *Information Systems Research*, 3, 1, 60–95.

Fenton, N., Pfleeger, S.L. and Glass, R.L. (1995) Science and substance: a challenge to software engineers, *IEEE Software*, 11, 4, 86–95.

Fleck, J. (1993) Configurations: crystallizing contingency, *The International Journal of Human Factors in Manufacturing*, 3, 4, 45–36.

Fleck, J. (1994) Learning by trying: the implementation of configurational technology, *Research Policy*, 23, 6, 637–652.

Floyd, C. (1992) "Software development as reality construction" in C. Floyd, H. Züllinghoven, R. Budde and R. Keil-Slawik (eds) *Software development and reality construction*, Springer, Berlin, 86–100.

Glass, R.L. (1994) The software-research crisis, *IEEE Software*, 11, 6, 42–47.

Goguen, J.A. (1992) *Requirements engineering: reconciliation of technical and social issues*, Technical Report, Centre for Requirements and Foundations, Oxford University Computing Lab, Cambridge.

Introna, L.D. and Whittaker, L. (2002) "The phenomenology of information systems evaluation: overcoming the subject/object dualism" in E.H. Wynn, E.A. Whitley, M.D. Myers and J.I. DeGross (eds.) *Global and organizational discourse about information technology*, Kluwer, Boston, 155–175.

Johnston, R.B. and Milton, S.K. (2002) The foundational role for theories of agency in understanding of information systems design, *The Australian Journal of Information Systems*, 9, Special Issue, 40–49.

Lauesen, S. (2004) "COTS tenders and integration requirements" in *Proceedings of the IEEE Joint International Conference on Requirements Engineering*, Kyoto, Japan, 166–175.

Lyytinen, K. and Hirschheim, R.A. (1987) Information systems failures: a survey and classification of the empirical literature, *Oxford Surveys in Information Technology*, 4, 257–309.

Maiden, N.A.M. and Ncube, C. (1998) Acquiring COTS software selection requirements, *IEEE Softwar*e, 15, 2, 46–56.

Markus, M.L. (1983) Power, politics and MIS implementation, *Communications of the ACM*, 26, 6, 430–444.

Naur, P. (1993) Understanding Turing's universal machine: personal style in program description, *The Computer Journal*, 36, 4, 351–372.

Naur, P. and Randell, B. (1969) *Software engineering: a report on a conference sponsored by the NATO Science Committee*, 7.–11. October 1968, Garmisch, NATO Scientific Affairs Division, Brussels.

Oei, J.L.H., Van Hemmen, L.J.G.T., Falkenberg, E. and Brinkkemper, S. (1982) *The meta model hierarchy: a framework for information systems concepts and techniques*, Technical Report No. 92–17, Department of Information Systems, University of Nijmegen.

Orlikowski, W. and Iacono, S. (2001) Desperately seeking the "IT" in IT research—a call to theorizing the IT artifact, *Information Systems Research*, 12, 2, 121–134.

Paetsch, F., Eberlein, A. and Maurer, F. (2003) "Requirements Engineering and Agile Software Development" in *Proceedings of the International Workshop on Enabling Technologies: Infrastructure for Collaborative Enterprises*, Linz, Austria, 308–313.

Robey, D., Ross, J.W. and Boudreau, M.C. (2002) Learning to implement enterprise systems: an exploratory study of the dialectics of change, *Journal of Management Information Systems*, 19, 1, 17–46.

Rolland, C. and Prakash, N. (2000) Bridging the gap between organizational needs and ERP functionality, *Requirements Engineering*, 5, 3, 180–193.

Rosemann, M., Vessey, I. and Weber, R. (2004) "Alignment in enterprise systems implementations: the role of ontological distance" in *Proceedings of the International Conference on Information Systems*, Washington, D.C., 439–447.

Sauer, C. and Willcocks, L. (2004) "Strategic alignment revisited: connecting organizational architecture and IT infrastructure" in *Proceedings of the Hawaii International Conference on Systems Sciences*, Hawaii, USA.

Scott, J.E. and Vessey, I. (2000) Implementing enterprise resource planning systems: the role of learning from failure, *Information Systems Frontiers*, 2, 2, 213–232.

Scott, J. E. and Vessey, I. (2002) Managing risks in enterprise systems implementations, *Communications of the ACM*, 45, 4, 74–81.

Shaw, M. (1990) Prospects for an engineering discipline of software, *IEEE Software*, 7, 11, 15–24.

Sia, S.K. and Soh, C. (2002) "Severity assessment of ERP–organization misalignment: honing in on ontological structure and context specificity" in *Proceedings of the International Conference on Information Systems*, Barcelona, 723–729.

Soffer, P. (2004) "Fit measurement: how to distinguish between fit and misfit" in *Proceedings of the Conference on Advances in Software Engineering Workshops*, 253–254.

Soffer, P., Golany, B. and Dori, D. (2003) ERP modeling: a comprehensive approach, *Information Systems*, 28, 6, 673–690.

Soffer, P., Golany, B., Dori, D. and Wand, Y. (2001) Modelling off-the-shelf information systems requirements: an ontological approach, *Requirements Engineering*, 6, 3, 183–199.

Soh, C., Sia, S. K. and Tay-Yap, J. (2000) Cultural fits and misfits: is ERP a universal solution?, *Communications of the ACM*, 43, 4, 47–51.

Strassmann, P. A. (2002) Out of alignment, *Computerworld*, 36, 10, 27.

Suchman, L. A. (1987) *Plans and situated actions: the problem of human-machine communication*, Cambridge University Press, Cambridge.

Tolvanen, J.-P. (1998) *Incremental method engineering with modeling tools: theoretical principles and empirical evidence*, PhD thesis, University of Jyväskylä , Jyväskylä.

Truex, D.P., Baskerville, R. and Travis, J. (2000) Amethodical systems development – The deferred meaning of systems development methods, *Accounting, Management & Information Technology*, 10, 1, 53–79.

Turner, J. (1987) "Understanding the elements of system design" in R. J. Boland and R. A. Hirschheim (eds.) *Critical issues in information systems research*, Wiley, Chichester , 97–111.

Wu, J.-H., Shin, S.-S., and Wu, C.-C. (2005) "COTS-based systems: a methodology for evaluating data and output misfits" in *Proceedings of the Hawaii International Conference on Systems Sciences*, Hawaii, USA.

ACKNOWLEDGEMENTS

This study is part of a larger project titled "Using Measures of Ontological Distance to Evaluate the Alignment between Organisational Needs and Enterprise Systems Capabilities", which is funded by SAP Research as part of an Australian Research Council (ARC) Linkage Grant – project number: LP0454094. We are indebted to the consultants who generously shared their insights with us.

COPYRIGHT