Integration of change and evolution in the lifecycle of SMEs information systems

Alexandre Métrailler

Information System Institute, Faculty of Business and Economics, University of Lausanne,
1015 Lausanne, Switzerland,
`Alexandre.Metrailler@unil.ch`,
WWW home page: `http://www.hec.unil.ch/isi/home`

Abstract. Today's software enterprise systems continue to suffer from symptoms like aging and the lack of evolution capabilities. One of the major reasons for this problem is that the future maintenance and evolutions are undervalued during traditional software systems implementation. The best way to overcome the problem of software aging and the misalignment with business is by placing the change in the center of software system lifecycle.
This thesis proposal presents the main activities required to create a method allowing the integration of change in the lifecycle of SMEs information systems (IS). By following a design science approach in IS, this study will explore the main challenges faced by small and medium enterprises (SMEs) regarding the evolutions of their IS, and the benefits of an agile approach to integrate change throughout the IS lifecycle.

1 Introduction

Small and medium enterprises (SMEs) represent a sector with one of the largest growth in the economy. Following a long term and well known trend of large companies, SMEs become increasingly dependent on information systems to support their operations (Premkumar, 2003). While there has been extensive research concerning the lifecycle and the evolution of information systems (IS) in large companies, only few research has explored the topic of lifecycle and evolution of IS in SMEs.

Nowadays, business environments progress and change rapidly to move with evolving markets. Those evolutions lead to a considerable amount of changes in business processes. Some of those are supported by IS and since the business processes change or new market opportunities appear, the IS needs to be adapted in order to continue to support the business. Regarding this context, the need of IS evolution is intrinsic.

Currently, if SMEs want to compete effectively within their market, the adoption of a sustainable enterprise IS is necessary. In terms of information system needs, SMEs tend to adopt a kind of incremental approach to problem resolution. Namely, when they need an application to manage accounting, they buy (or develop) it; when they need an application to manage stocks, they buy (or develop) it, etc. (Equey, 2006). Thus, SMEs incrementally construct their IS infrastructure.

IS undergo several distinguish stages during its life. According to Bennett and Rajlich (2000), the first stage is *initial development*, where the first functioning version of the software is developed. Then it goes through *evolution*, where the capacities are extended, afterwards comes the stage *servicing* during which the software is used and maintained. The two last stages are *phase out* and *close down*, during them, no more servicing is being undertaken, the system is used as long as possible until it is withdrawn from business. Here is the traditional description of an IS lifecycle. Bennett and Rajlich (2000) propose a versioned model of this traditional staged lifecycle model. The versioned model consists in the creation of versions of the system and to apply to them the traditional lifecycle stages. This research proposes to extend Bennett versioned model by integrating a continuous evolution perspective inside the lifecycle of IS.

To integrate change in the software development, a typical approach is to follow an iterative and incremental process. These development methodologies are also known as agile software development methodologies (Beck et al., 2001). They have been developed as a reaction to the waterfall-like bureaucratic project management methods ending with a big rollout (*big bang*). One of the principles of agile methodologies is to welcome change and to consider it as an opportunity (Fowler, 2001).

According to Alleman (2002), such a methodology allows the ERP projects to adapt to the SME context. Moreover, the work of Stender (2002) points out that the increasing complexity of ERP deployment projects has led to the recommendation of encouraging incremental over big-bang approaches. Agile methodologies promote collaborations, interactions, user involvement, working software and high response to change (Beck et al., 2001).

Based on these aspects, during preliminary research work, we addressed the following research question : *"How to apply agile principle to the deployment of an open source ERP in SMEs"*. The output of this previous part was an agile and incremental methodology for ERP deployment. We published a first paper in the proceedings of the AIM 2009 Conference (Métrailler and Estier, 2009), and an extended version of our research is currently under review (Métrailler and Estier, 2011).

By comparison with traditional (waterfall) methodologies, this approach allows, among others, an improvement of flexibility and of project priorities in planning, a reduction of the ERP complexity through several rollouts and a better distribution of the extra workload among stakeholders. This methodology has been developed in collaboration with business partners who are currently using it and obtain good results.

Our agile deployment methodology progress in an incremental manner, which means that between two rollouts, the system is used in production. Since each rollout provides new features, there is an evolution of the system in production. Consequently, this approach helps SMEs to integrate a continuous evolutional perspective in their ERP lifecycle. Therefore, the continuation of this research focuses on identifying challenges relative to IS evolution in SMEs and on the opportunities concerning the use of an agile approach for evolution.

This paper is structured as follows. In next section, we expose the research questions. In section 3, we present an overview of previous research in systems evolution. In Section 4, the research methodology used to conduct this study is described, namely design science in IS. Section 5 presents the current state of our work and the required stages to carry out this research. In addition, Section 5 presents the research activities for each stage of the research methodology. Finally, the expected contributions and the limitations of this research are presented in Sect. 6.

2 Research questions

Regarding the fact that to support business activities the IS must evolve, and the fact that an agile, iterative and incremental process allows the integration of change, the general research question is as follows:

Are agile principles beneficial to SME's information systems evolution?

To answer this question, we aim to design an agile method to conduct IS evolution in SMEs and to evaluate its impact on IS evolution. In order to accomplish this, additional research questions also need answers:

Concerning SMEs IS in general:

- *How do SMEs deal with the changes of their IS? Are there differences between in-house developed software systems and commercial off-the-shelf (COTS) systems?*
- *What are the advantages and disadvantages of both in-house developed systems and COTS systems regarding system evolution?*
- *Do known laws of software evolution apply to SMEs information systems evolution?*

Concerning agile priciples and SMEs IS:

- *In the first stage of IS lifecycle, is an iterative and agile implementation methodology more efficient regarding the incoming evolutions of IS?*
- *How can we enhance such iterative and agile implementation methodology to integrate an evolutional perspective?*

3 Literature review

In this section, we present a brief overview of previous research on which we refer and on which we will rely to design our artifact to conduct evolution in SMEs IS.

3.1 Information systems evolution

The term evolution is elusive to define. Dictionaries and common sense refer to the following definition: “Evolution is a process of gradual development in a particular situation or thing over a period of time[1]”.

When it comes to software systems, the term evolution has many different interpretations depending on the role of stakeholder. To define evolution in a way that is independent of subjective interpretations and that captures characteristics of evolution in software systems, Lehman (cited in Cook et al. (2006)) proposed the following general statement:

> ‘a . . . process of discrete, progressive, change over time in the characteristics, attributes, [or] properties of some material or abstract, natural or artificial, entity or system or of a sequence of these [changes]’.[2]

One of the principal researchers in the field of software evolution is Professor M.M. Lehman. He is the principal developer of the laws of software evolution. These laws have been developed for more than 30 years with the first version published in 1974. They are the basis for the elaboration of a theory of software evolution (Lehman and Ramil, 2000, 2001a; Madhavji et al., 2006). Lehman uses the term *law* to describe general principles of how software systems change over time. The laws describe phenomena such as continual adaptation, complexity increases, self regulating capabilities, stability and familiarity conservation, feedback loop system, as well as how these principles vary by increasing, decreasing, remaining constant or by being organized in multiple (Lehman and Ramil, 2001b; Cook et al., 2006).

Lehman made a fundamental distinction between three types of programs. Those written to satisfy a fixed specification (*S-type*, S for specification), those developed to satisfy a need in the real world (*E-type*, E for Evolving), and a third type of programs the *P-type* (P for Problem). This third type always satisfies the definition of either *S-* or *E-type*, thus, in his subsequent research Lehman ignored the type P. This distinction is known as the SPE taxonomy (Cook et al., 2006).

Several researchers described the evolution of E-type softwares. To compare and categorize evolutions of such systems, three decades ago, Lientz and Swanson (1980) proposed a software maintenance typology that distinguishes among perfective, adaptive and corrective maintenance activities. More recently, Chapin et al. (2001) refined this typology into 12 different types of software changes: enhancive, corrective, reductive, adaptive, performance, preventive, groomative, updative, reformative, evaluative, consultive, and training. Moreover they distinguished whether these changes are categorized as software maintenance or

[1] Collins Cobuild English Dictionary for Advanced Learners 4th edition published in 2003 © HarperCollins Publishers 1987, 1995, 2001, 2003 and Collins A-Z Thesaurus 1st edition first published in 1995 © HarperCollins Publishers 1995

[2] The ‘Software Evolution and Evolutionary Computation Symposium’ (EPSRC Network on Evolvability in Biology and Software Systems), Hatfield, U.K., 7–8 February 2002.

evolution. This work categorizes software changes on the basis of their purpose (i.e. the *why* of software change). On an other side, Buckley et al. (2005) take a complementary view of the domain; indeed, they focus on the technical aspects of software change by creating a taxonomy of software change (i.e. the *when, where, what* and *how* of software change). These results provide a strong basis to classify software evolution according to each dimension cited above. Consequently, we will use this typology and this taxonomy to investigate the evolutions of SMEs enterprise IS.

3.2 SMEs and information systems

A significant amount of research has been completed on what distinguish SMEs from larger firms, it points out that the principal distinctions are the limited resources and the limited knowledge base of SMEs (Fuller and McLaren, 2010; Kugel, 2007; Loh and Koh, 2004). These two distinctions can impact the choice of enterprise IS. Therefore, the system should be affordable and easy to implement (Fuller and Mclaren, 2010).

SMEs information systems can be grouped in two main categories: enterprise systems developed by the company (in-house developed software systems) and enterprise systems purchased by the company (COTS) like ERP or SaaS (Software as a Service) and BoB (Best of Breed) approaches. Based on the fact that all these solutions will eventually evolve according to the business context of the company, our interest is to determine the advantages and inconvenients of each category and whether a solution is more able to evolve in SMEs business context.

A recent study of Fuller and McLaren (2010) points out that ERP systems are better aligned with SMEs than SaaS and BoB. Moreover, ERP systems have a higher potential for long-term growth. However, SMEs are not yet familiar with ERP. Indeed, many of them are still in the process of adopting these tools. For example, the percentage of Swiss SMEs using an ERP is low, between 17% and 19% (Equey, 2006). Consequently, in the years to come, a considerable number of SMEs may need to seek an enterprise information system. Thus, considering the dynamics of business environments, the integration of an evolutionary perspective in the early stages of IS seeking should not be underestimate.

3.3 Conclusion

We believe that the existing research has missed to explore the evolution of SMEs information system.

The research concerning the laws of software evolution is based on the evolution of large E-type software. Lehman's Evolution laws are originally based on observations regarding the evolution of IBM's OS/360 and OS/370.

Empirical research on the evolution of open source software point out that the evolution of such software breaks some of Lehman's laws. One way among others to explain these results is the decentralized manner by which such software are developed and the loosely-coupled community of developers (Scacchi, 2003).

Another study concerning the evolution of Eclipse plug-ins (Businge et al., 2010) identifies two Lehman's laws that cannot be validated. Consequently, Lehman's laws do not seem to be applicable to all software systems. Moreover, it is not clear whether they are applicable to medium sized systems like SMEs IS or to systems incorporating COTS components.

When it comes to the taxonomy of software evolution or the typology of software change, there is no application of them to SMEs context. The fact of carrying out such a research could point out interesting elements concerning namely the temporal properties, the object, the system properties or the challenges relative to software changes in SME.

On another side, there is no research concerning the evolution challenges relative to in house developed or COTS systems in SME, is there an approach which is more able to evolve than the other?

4 Methodology

To carry out this research, we use a design science approach in IS. According to (Hevner et al., 2004), design science *"creates and evaluates IT artifacts intended to solve identified organizational problems"*.

Peffers et al. (2007) propose a Design Science Research Methodology (DSRM) process model which is composed by six steps (1. Identify problem and motivate, 2. Define objectives of a solution, 3. Design and development, 4. Demonstration, 5. Evaluation, 6. Communication). This process is structured in a nominally sequential order. However, it is not expected that researcher should always proceed in order from activity one to six. Peffers et al. (2007) describe four "entry points" through which the researcher can start (from 1 to 4).

During the preliminary research work, we followed a problem-centered approach by starting with step 1. We identified some issues that emanate from SMEs, namely the problem of flexibility in the traditional (waterfall) ERP deployment approach, difficulties related to the workload, the limited resources of SMEs, and the complexity of appropriating the tool. Using an iterative approach we performed each step to create our agile deployment methodology.

The sequel of our research will follow a Design- and Development-Centered approach as described by Peffers et al. (2007) and illustrated in Fig. 1. Indeed, based on the agile deployment methodology we developed previously, we want to build a model of evolution management integrated throughout the lifecycle of SMEs IS. Obviously, new questions arise concerning the evolution of SMEs IS: their nature, the causes which trigger them, the applicability of our methodology to other type of IS, etc.

Our research activities for each stage of the DSRM process model (see Fig.1) are described in the next section.

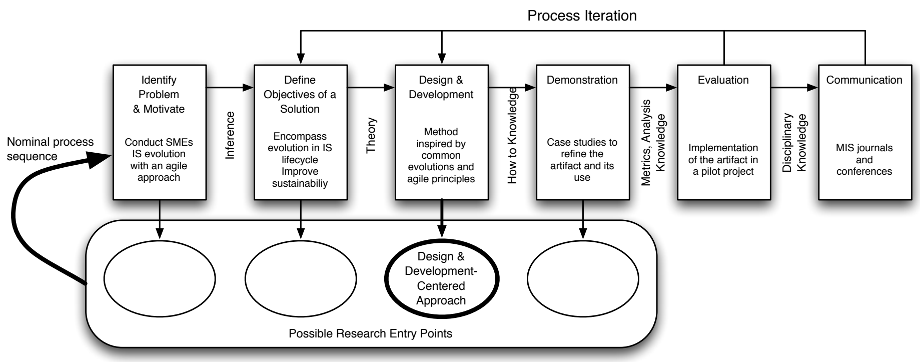
Process Iteration
Identify Problem & Motivate
Conduct SMEs IS evolution with an agile approach
Inference
Define Objectives of a Solution
Encompass evolution in IS lifecycle Improve sustainabiliy
Theory
Design & Development
Method inspired by common evolutions and agile principles
How to Knowledge
Demonstration
Case studies to refine the artifact and its use
Metrics, Analysis Knowledge
Evaluation
Implementation of the artifact in a pilot project
Disciplinary Knowledge
Communication
MIS journals and conferences
Nominal process sequence
Design & Development-Centered Approach
Possible Research Entry Points

Fig. 1. The DSRM process model applied to this research which starts in the third "entry point" (Design- and Development-Centered approach)(Peffers et al., 2007)

5 Description of the current state of the work

This section gives a brief description of the current state of our research and introduces the ideas we are currently working on. The stages of the research follow the DSRM process model described in Sect. 4.

As mentionned before, the sequel of this research will follow a Design- and Development- Centered approach (Peffers et al., 2007). In our case, the entry point of the research is the reuse of our agile methodology concepts to build the artifact of this research, namely a method to conduct IS evolution in SMEs.

5.1 Problem identification and motivations

After having identified the entry point of the research, the next activity in Peffers DSRM (Fig. 1) is to define the research problem and to justify the value of the solution. The global research problem has already been defined in Sect. 1. Today's software enterprise systems continue to suffer from symptoms like aging and the lack of evolution capabilities. One of major reasons of this problem is that the future maintenance and evolutions are undervalued during traditional software systems implementation. The best way to overcome the problem of software aging and misalignment is to center the software system lifecycle on change management (Mens et al., 2005).

5.2 Objectives of the artifact

The objective of our artifact is to conduct SMEs IS evolution by applying principles of agile software development methodologies to the management of IS evolutions. Our previous work points out that an agile approach for ERP deployment is more aligned with SMEs context than traditional waterfall approaches. Moreover, it is well known that agile development methodologies enable the integration of change throughout software development. We believe that such approach enables the simplification of IS evolutions, an improvement of IS reactivity towards business adjustments and an expansion of the IS lifecycle by maintaining an evolution stage instead of going through servicing and phase-out stages. The IS should be designed for evolution from the earliest stage of its lifecycle, by having the evolution integrated in its conception and deployment.

5.3 Design and development of the artifact

Despite these facts, the design and the development of such an artifact require investigations regarding the evolutions of SMEs system. As mentioned in the literature review (see Sect. 3), it is not clear whether general E-type software evolution laws are applicable to medium sized systems or to systems that incorporate software applications from different vendors. In order to address IS evolution issues, it is essential to identify these issues. In the literature, there is no research identifying software evolutions in SMEs and their context. Nevertheless, the literature provides tools to undertake such research, that are typology and taxonomy of software evolution (Chapin et al., 2001; Buckley et al., 2005). In section 3, it is mentioned that SMEs information systems can be grouped into two main categories that are in-house developed or purchased. There is no research concerning the evolution challenges relative to each categories of systems, what are the pros and cons of each categories regarding evolution capabilities? Interviews with software developers and software project managers will be used to answer these interrogations and to identify the challenges they face with evolution. This research and the literature review will provide useful foundation to start the development of the artifact.

After having shed light on these interrogations, we would have the basis to start the development of our artifact. The design and development activities will be an iterative process around small case studies to refine and demonstrate the use of the artifact. When our artifact seems to be mature enough, we would apply it to conduct real SME's IS evolutions.

5.4 Evaluation of the artifact

After the development of the artifact, it is necessary to evaluate the artifact to determinate *how well it works* (Hevner et al., 2004). The evaluation of this research artifact will be qualitative, principally based on stakeholders' feedback and satisfaction surveys. The appreciation of how well our artifact achieved his objectives must be collected from experts dealing with maintenance and evolutions of SMEs IS. We will examine whether IS managers, or integrators are willing to adopt the artifact, whether they perceive a higher evolution potential of the IS and if they predict a better sustainability of the system.

6 Expected contributions and conclusion

In this section we discuss about limitations and contributions of our study.

Our study has limitations. The first limitation we distinguish is relating to the iterative and agile approach uses by our artifact to manage IS evolution. Our previous work is based on an open source system, consequently, we have a complete access to its architecture and to its data model. This transparency allows us to master the technical dependencies among components which permits us to uncouple them if necessary. We do not apply yet our previous work to a proprietary system, but we guess the IS's flexibility will be reduced to the editor's choices.

The second limitation to the adoption of our artifact results from the iterative approach and the integration of the evolution perspective in the IS lifecycle. In fact, the IS will be in continuous evolution and those who have to manage it may need to provide an extra effort.

Moreover, we assume that the evolution perspective should be integrated since the beginning of the system implementation. If elements of the systems are strongly interconnected since its implementation, it is possible that our artifact could not be integrally used.

Although this study has limitations, it offers several contributions to practice as well as research. In terms of research, this work will extend the knowledge on evolution of SMEs software systems. The exploratory, qualitative research has a primary contribution of identifying the strong and weak points of both in-house developed and COTS systems regarding their evolution capabilities. Its secondary contribution lies in the identification of the main reasons why software evolves in SMEs and of what types are these evolutions. Moreover, these results will allow us to evaluate how the evolution of small-medium software systems conforms to the laws of software evolution.

By describing and understanding the evolutions of SMEs software systems and the main activities generated by these evolutions, one then has the potential to suggest improvements. The contribution of the design and development phase of a design science research is to determine how to more sustainably conduct the lifecycle of SMEs software systems. To these ends, this study delivers a method to integrate an evolutional perspective into the lifecycle of such software systems.

References

Alleman, G. B. (2002). Agile project management methods for ERP: how to apply agile processes to complex cots projects and live to tell about it. *Lecture Notes in Computer Science*, pages 70–88.

Beck, K., Beedle, M., van Bennekum, A., Cockburn, A., Cunningham, W., Fowler, M., Grenning, J., Highsmith, J., Hunt, A., Jeffries, R., and Others (2001). Manifesto for agile software development. *The Agile Alliance*, pages 2002–2004.

Bennett, K. H. and Rajlich, V. T. (2000). The staged model of the software lifecycle : A new perspective on software evolution. *Evolution*, pages 1–14.

Buckley, J., Mens, T., Zenger, M., Rashid, A., and Kniesel, G. (2005). Towards a taxonomy of software change. *Journal of Software Maintenance and Evolution: Research and Practice*, 17(5):309–332.

Businge, J., Serebrenik, A., and Brand, M. V. D. (2010). An Empirical Study of the Evolution of Eclipse Third-party. In *Science And Technology*, volume 2009, pages 63–72.

Chapin, N., Hale, J. E., Khan, K. M., Ramil, J. F., and Tan, W.-G. (2001). Types of software evolution and software maintenance. *Journal of Software Maintenance and Evolution: Research and Practice*, 13(1):3–30.

Cook, S., Harrison, R., Lehman, M. M., and Wernick, P. (2006). Evolution in software systems: foundations of the SPE classification scheme. *Journal of Software Maintenance and Evolution: Research and Practice*, 18(1):1–35.

Equey, C. (2006). Etude du comportement des PME/PMI suisses en matière d'adoption de système de gestion intégré. Entre méconnaissance et satisfaction.

Fowler, M. (2001). The new methodology. *Wuhan University Journal of Natural Sciences*, 6(1):12–24.

Fuller, S. and McLaren, T. (2010). Analyzing Enterprise Systems Delivery Modes for Small and Medium Enterprises. *AMCIS 2010 Proceedings*.

Fuller, S. and Mclaren, T. (2010). Analyzing Enterprise Systems Delivery Modes for Small and Medium Enterprises for Small and Medium Enterprises. *Information Systems*.

Hevner, A. R., March, S. T., Park, J., and Ram, S. (2004). Design science in information systems research. *Management information systems quarterly*, 28(1):75–106.

Kugel, R. (2007). The Big Midsize Challenge. *Business Finance*.

Lehman, M. M. and Ramil, J. F. (2000). Towards a theory of software evolution-and its practical impact. *invited talk, Proceedings ISPSE*, pages 2–11.

Lehman, M. M. and Ramil, J. F. (2001a). An approach to a theory of software evolution. In *Proceedings of the 4th international workshop on Principles of software evolution*, pages 70–74. ACM.

Lehman, M. M. and Ramil, J. F. (2001b). Rules and tools for software evolution planning and management. *Annals of Software Engineering*, 11(1):15–44.

Lientz, B. P. and Swanson, E. B. (1980). *Software maintenance management: a study of the maintenance of computer application software in 487 data processing organizations*, volume 4. Addison-Wesley Reading MA.

Loh, T. C. and Koh, S. C. L. (2004). Critical elements for a successful enterprise resource planning implementation in small-and medium-sized enterprises. *International Journal of Production Research*, 42(17):3433–3455.

Madhavji, N. H., Lehman, M., Perry, D., and Ramil, J. F. (2006). *Software evolution and feedback*. Wiley Online Library.

Mens, T., Wermelinger, M., Ducasse, S., Demeyer, S., Hirschfeld, R., and Jazayeri, M. (2005). Challenges in Software Evolution. *Eighth International Workshop on Principles of Software Evolution (IWPSE'05)*, pages 13–22.

Métrailler, A. and Estier, T. (2009). Déploiement agile d'ERP open source en PME. In *Actes du Colloque AIM 2009*.

Métrailler, A. and Estier, T. (2011). Agile Deployment Methodology for Open Source ERP in SME. *Submited for publication to "Revue Systèmes d'Information et Management", under review.*

Peffers, K., Tuunanen, T., Rothenberger, M. a., and Chatterjee, S. (2007). A Design Science Research Methodology for Information Systems Research. *Journal of Management Information Systems*, 24(3):45–77.

Premkumar, G. (2003). A meta-analysis of research on information technology implementation in small business. *Journal of Organizational Computing and Electronic Commerce*, 13(2):91–121.

Scacchi, W. (2003). Understanding Open Source Software Evolution : Applying , Breaking , and Rethinking the Laws of Software Evolution. *Artificial Life*.

Stender, M. (2002). Outline of an Agile Incremental Implementation Methodology for Enterprise Systems. In *Proceedings of the Eighth Americas Conference on Information Systems*, pages 907–917.