

THE SCENARIO FOR CONSTRUCTING FLEXIBLE, PEOPLE-FOCUSED SYSTEMS DEVELOPMENT METHODOLOGIES

Bajec, Marko, University of Ljubljana, Faculty of Computer & Information Science, Trzaska 25, 1000 Ljubljana, Slovenia, marko.bajec@fri.uni-lj.si

Krisper, Marjan, University of Ljubljana, Faculty of Computer & Information Science, Trzaska 25, 1000 Ljubljana, Slovenia, marjan.krisper@fri.uni-lj.si

Rupnik, Rok, University of Ljubljana, Faculty of Computer & Information Science, Trzaska 25, 1000 Ljubljana, Slovenia, rok.rupnik@fri.uni-lj.si

Abstract

Systems development is a very complex process, which requires disciplined methodological approaches. While there are many arguments underpinning the use of systems development methodologies empirical investigations show that methodologies are underused in practice and what is more their use is not on the increase. In this paper we discuss the usage of agile methodologies which tend to be more people-focused and adaptable to project-specific circumstances. We believe that usage of such methodologies may encourage practitioners to rank the methodologies higher in terms of their contributions to successful development. In the paper we propose a scenario for development, introduction and maintenance of an agile methodology in an organisation.

Keywords: systems development, systems development methodology, agile methodology, method engineering

1 INTRODUCTION

According to the literature, the use of methodologies in systems development is axiomatically appropriate as it improves both, the process and its' product (Fitzgerald 1995). In practice, however, the picture seems different. While there are a number of arguments in favour of systematic methodological approaches, reports show that practitioners do not see methodologies as panacea for problems in systems development. Even those using methodologies rank them low in terms of their contribution to successful development (Fitzgerald 1998). One of the explanations for this is the use of highly prescriptive methodologies that encode as much as possible about the way of working defining each development step in its most detail. The use of such methodologies in practice is difficult due to several reasons:

- The burden carried by the participants of the project, if following a complex methodology, is heavy, since they have to take care of numerous side tasks and work products. This hinders the use of methodology and puts it in question, especially when rapid results are required (Cockburn 2002, Willcocks & Sykes 2000, Middleton 1999, Larman 2003, Ambler 2002).
- In many cases highly prescriptive methodologies put too much emphasis on the development process at the expense of people and organisational issues. The ignorance of sociological aspects has been proved in practice as having negative effect on the methodology usage (Middleton 1999, Cockburn 2000).
- The results in Fitzgerald's research (1998) show that methodologies are neither applied rigorously nor uniformly, even when training in their use has been provided. This supports the view that in each development project unique methodology-instance is created. The use of the highly prescriptive methodologies is thus inappropriate due to their inherent complexity, which makes the adaptation of the methodology to project-specific circumstances very difficult (Henderson-Sellers 2003). Even though the notion of this phenomenon is old at least two decades (see DeMarco 1982, p.131) we can still find methodologies – especially in bureaucracies – that are extremely rigid and do not allow any adaptation (Middleton 1999).
- Many current methodologies are derived from practices and concepts relevant to the old organisational environment (Fitzgerald 1998). Such methodologies clearly need to be reconsidered to found out if they still serve to the new climate. One of the characteristics of the today's business environment which seems to be neglected in the past is the need for rapid development which is forced by today's dynamic environments and the continually evolving nature of information technology (Willcocks & Sykes 2000).
- In many cases highly prescriptive methodologies have no empirical base (e.g. SSADM, see Middleton 1999). In such cases, the techniques and methods are ignored if not found to produce benefits.

The objective of this paper is to describe an approach that contrasts with the use of complex, heavily prescriptive and rigid methodologies. Those are typically presented as *out-of-the-box methodologies*, which are ready for immediate use. Our experiences in practice have shown that a methodology is much more than just a set of methods and techniques. An important aspect of a methodology is its sociological component reflecting the organisations culture, personal incentives and above all the knowledge and skills of the organisation members. This underlines the fact that a methodology cannot arise as something independent from people for whom it has been meant and puts in question the use of *pre-packaged* methodologies (section 2). In the paper we will introduce the concept of an agile methodology, which is today well-known but often unfairly used term in systems development. We will explain our understanding of the concept emphasising the fact that an agile methodology is not necessarily *light methodology* (section 3). As the main contribution we will present a scenario for constructing flexible, people-focused systems development methodologies (section 4). The scenario has been developed based on our experiences in implementing agile methodologies.

2 RESEARCH METHODOLOGY

The research of which results are presented in this paper is primarily based on the literature review and the experiences captured while implementing agile and non-agile methodologies in Slovenian companies. Comparing it with the existing research, our work mostly lean on the findings of the so called “method engineering”, which support the fact that it is unreasonable to expect the same methodology can work for any project. While a lot of research work has been done on this matter, mostly from technical point of view (e.g. how to construct a methodology so that it will suite to the needs of a particular project), social factors that heavily impact the possibility of the methodology acceptance by the team, have been typically neglected. Agile (light) methodologies that represent a new trend in the field of IS development tend to put social requirements forward, focusing on the methodology users rather than on the methodology itself. For the purpose of our research several light methodologies were studied comparing their approaches and techniques.

The scenario, which is described in section 5, was tested and refined based on the three real projects. Profiles of the companies involved are described below:

- Marand: Marand is a software company which develops software following an object oriented approach. It counts over 30 developers, from which the majority is well experienced and skilled. Before the scenario was used the company didn't use any formal methodology.
- Intereuropa, Intereuropa is the leading logistics provider in South-Eastern Europe. The company strives to renovate its IS. Its IT department counts about 40 employees, with no skills from the contemporary development tools and technologies. Before the scenario was used the company didn't use any methodology at all, since the IT department was only entitled to take care about the existing applications.
- Faculty of Computer & Information Science (FRI): FRI is a high-school institution with about 2500 students. For the purpose of developing a students record IS an add-hoc development team was created. The team shared a well framed theoretical background on software development methodologies but had little experience from practice.

3 METHODOLOGY AS SOCIAL CONSTRUCT

In the New Oxford English Dictionary, methodology is defined as ‘a system of methods used in a particular area of study or activity’, while method is ‘a particular form of procedure for achieving or approaching something in a systematic way’. Our opinion is that system development methodology is not just a set of methods which taken together can be used to develop software, but is in a first place a social construct. A methodology is full of philosophy, principles, ideas and points of view of the organization staff or its users, what explicitly emphasizes its social component. A methodology is everything we do to achieve a certain result, i.e. a product or services which are the goal of our work. When talking about systems development it does not merely mean activities that are directly connected to the development (i.e. analysis, planning, etc.) but also patterns of communication, collaboration and coordination, support procedures, means of communication with the parties involved, rules of decision, etc (Cockburn 2002). In this sense a methodology can be explained as a set of agreements made by a certain project group or organization (Agile Alliance). A methodology clearly cannot arise as something independent from people for whom it has been meant.

Another important reason for emphasising social aspects of a methodology comes from the methodology contents. A methodology consists on one hand of formal elements such as procedures, rules, directions, tools, standards, documented either in electronic or classical manuals, and on the other of certain undocumented elements, and above all the knowledge of the organization members. This is of utter importance for an organization, because it represents its own values and competitiveness. A methodology that is used by a particular organisation is typically richer then its' formal and documented part. A substantial part of a methodology is embodied by its users through the

knowledge and experience they carry. When their knowledge becomes sufficiently routinized and expressible, it can be transformed into an explicit form (e.g. method, procedures, guideline, advice, etc).

The knowledge transformation as described above is very important for a methodology construction and its use. A documented (formal) methodology that a company develops or purchases (out-of-the-box methodology) forms the basis from where the users learn and shape their own perception about the work, communication, decision-making etc. By using the methodology they become more and more experienced and enrich their knowledge, which consequentially influences the methodology itself. The moment the knowledge becomes routinized and expressible it adopts the form of data and thus can be formalized. In other words, the methodology can be enriched on the basis of individuals' tacit knowledge which can be transformed into explicit forms. It has to be taken into account, however, that tacit knowledge, which is the basis of any explicit knowledge, is always richer than the primary knowledge (Morabito & Bhate 2001, Maier & Rehtin 2000, O'Dell 1996). Explicit knowledge is only a manifestation of a richer tacit knowledge (Morabito & Bhate 2001). This explains why a methodology even if it is extensively documented, can never be 'taken apart' from their users. An interesting and in-depth discussion on this issue, examining canonical versus non-canonical work, can be found in (Brown & Dugid, 1991).

Figure 1 shows the cycle beginning with a formal methodology as the basis of learning and understanding of the methodology and proceeding to the formalization of the informal methodological components, which can eventually become routinized enough to be standardized.

A large number of software development companies still develop their software based on informally defined methodologies, i.e. methodologies which have not been documented. Although the procedures used are known and settled, are rarely explicitly put down. And even more rare they are refreshed by newly accomplished experiences and knowledge.

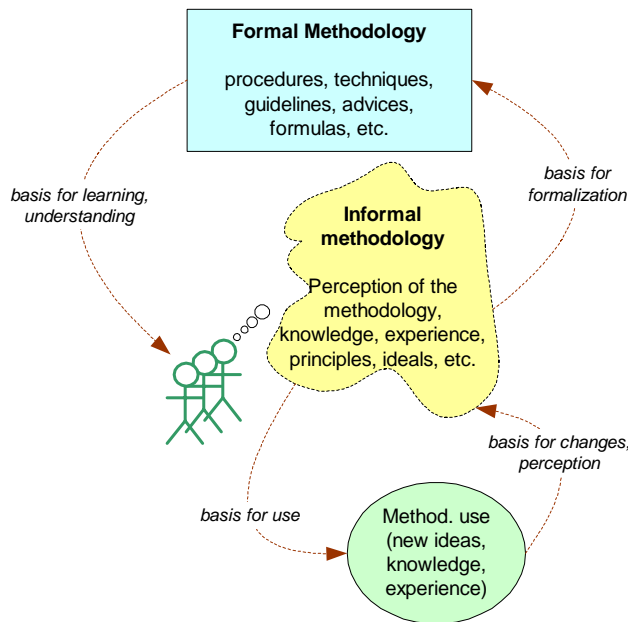


Figure 1: From formal to informal methodology

The borderline between formal and informal parts of a methodology is not easy to set up. If the formal part is extensive, the methodology is hard to maintain and quickly becomes obsolete not reflecting the actual development process. On the other hand, informal methodologies may lead into development

process which is completely dependent on the users. The need for the formalization of methodologies arises from the fact that the process of software development is a systematic process which has to be appropriately designed and documented to direct the groups and individuals easily towards the better results. If the methodology is based purely on an informal level, then the development process is more difficult to standardize, and the whole procedure may become too fuzzy and accordingly uncontrollable. There is also epistemological rationale for formalization of methodology. Formal methodology may provide a structural framework for the acquisition of knowledge. Any learning from the past development experiences can be systematized and stored for the future reference (Stolterman 1994). The question is, of course, to what extent should methodology be documented? Which are the most stable and most logical elements to formalize? How to ensure the methodology will suite to all projects taken by the organisation? How to achieve appropriate adaptability of the methodology and in the same ensure the development process will go through all the steps that are really important?

4 FLEXIBLE, PEOPLE-FOCUSED METHODOLOGIES

4.1 Method engineering

In academic literature the approach that contrasts with the use of pre-packaged methodologies is known as *method engineering* (Brinkkemper et al, 1996). The idea of method engineering is based on the construction of the methodology from the methodology fragments or components that form the methodology (Rolland & Plihon 1996, Harmsen 1994). Especially important is situational method engineering which is defined as the creation of a methodology specifically attuned to the project at hand (cf. Ralyté 2002, Brinkkemper 1996, Brinkkemper 1998, Rolland & Prakash 1996, Harmsen et al 1994, Kumar & Wellke 1992). Although advocated in the academia and implicitly promoted in ISO standards (e.g. ISO 12207), the method engineering has never been widely acknowledged or practiced by software engineers (Henderson-Sellers 2003). One of the reasons is that practitioners often view the method engineering as having a costly overhead in terms of time, people and money and do not take into account the cost and effort in case when a pre-packaged methodology is used and found as inappropriate to company's business processes (Henderson-Sellers 2003). With the emergence of the new approach, frequently called "agile approach", the idea seems to become more feasible, as this time the initiative comes from practitioners.

4.2 Understanding the concept of an agile methodology

The term "agile" was in connection with systems development methodologies first used in February 2001 when the group of 17 gurus from the field of light methodologies (Adaptive Software Development, XP-Extreme Programming, Feature-Driven Development, Crystal, Scrum, Dynamic System Development Method etc.) came together. Their aim was to find the common denominator between their 'own' methodologies, and according to their findings set up joint methodological fundamentals. The result of their meeting was establishment of an interest group (Agile Alliance, see Cockburn 2002, Appendix A) which promotes the "search for a better approach to the software development by involvement of the group members and help to others". The reason the group members adopted the term "agile" was due to their agreement on the importance of being able to respond to changing requirements within the project timeframe (Cockburn 2002). Today however, the term became buzzword and is in many cases used unfairly. To avoid these confusions we define here the concept of an agile methodology as it is used in this paper.

In literature, an agile methodology is often confused with a light methodology. Proponents of the agile approach often say an agile methodology like extreme programming (XP) must be followed in its entirety (e.g. XP without pair programming is not XP) (Henderson-Sellers, 2003). We argue that the

adaptability is independent from the methodology *weight*¹. Both *heavy* and *light* methodologies can be agile, as long as they enable an ad-hoc adaptability of the methodology size and its construction from the methodology components according to the project's actual needs.

The figure below depicts the concept of an agile methodology as promoted in this paper. As illustrated, the fundamental feature of the agile methodology is its ability to adapt its size and its contents according to various parameters, such as characteristics of the project, the organization culture, experience of the team, customers' special demands, Capability Maturity Model level of maturity, available tools, quality desired, number of people on the development team, etc. In this way an agile methodology becomes more organisation/people-focused and flexible. In further text we will use the term "agile methodology" to represent such methodologies that are constructed according to myriad variables pertinent to the development organisation and support ad-hoc adaptations according to project-specific circumstances.

In Figure 2, the mechanism supporting the ad-hoc construction of a methodology instance is depicted as a repository based system, which takes various parameters as input and constructs a methodology instance as output. The realisation of such a system might be very complex (see e.g. Heym & Osterle 1993) and is out of the scope of this paper. It has to be highlighted however that in the easiest way such a mechanism would only support the selection of the methodology fragments (of some predefined methodology skeleton) that are compulsory and those that are optional (Henderson-Sellers 2003).

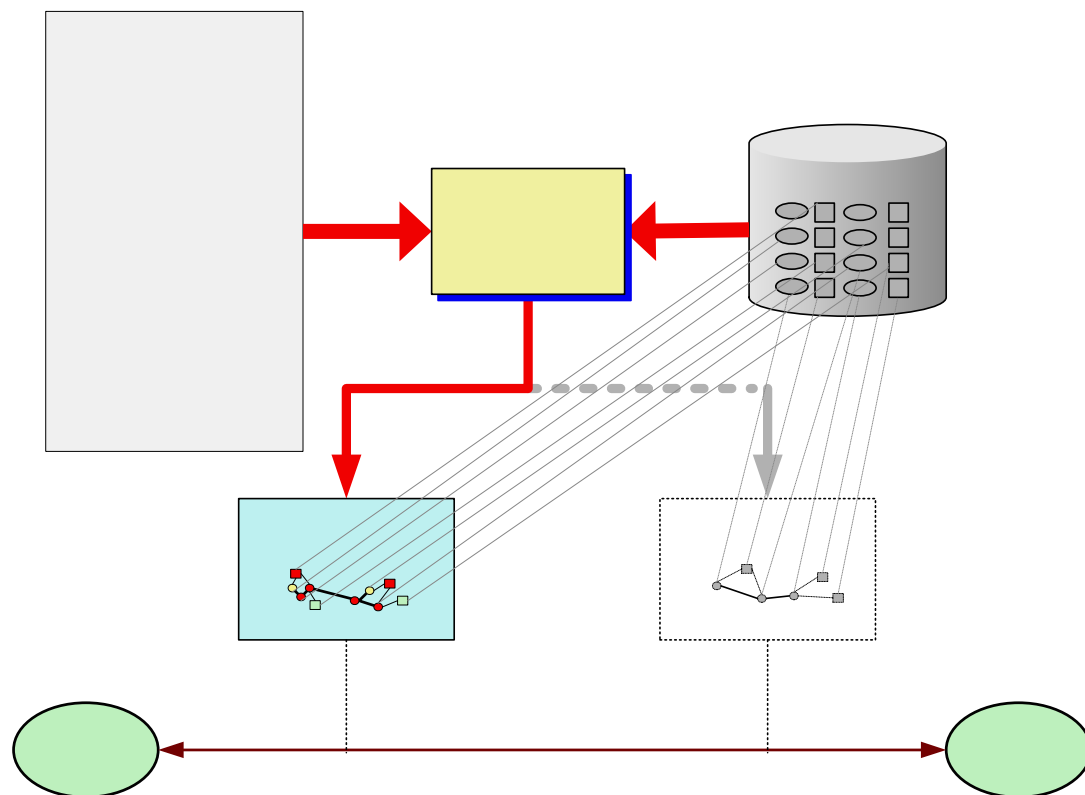


Figure 2: The concept of an agile methodology

¹ Cockburn (2002) defines methodology weight as a product of size and ceremony. The ceremony represents the amount of precision and the tightness of tolerance in the methodology. Greater ceremony corresponds to tighter controls. The size is the number of control elements in the methodology. Each deliverable, standard, activity, quality measure, and technique description is an element of control.

5 THE SCENARIO FOR DEVELOPMENT, INTRODUCTION AND MAINTENANCE OF AN AGILE METHODOLOGY IN AN ORGANISATION

Based on the experience that we gained in the last few years implementing agile methodologies in software companies, we have determined a scenario that covers the most important activities for development and introduction of an agile methodology in an organisation. The scenario helps to establish a system development methodology which is most appropriate for an organisation and tells how to organise the development process in a way that it will support agility and ad-hoc adaptations based on a particular situation or project. The scenario also provides activities and roles that take care about constant accumulation of knowledge perceived through the methodology use.

The scenario is depicted in Figure 3. It consists of three phases:

- Design of an agile methodology,
- Adapting the methodology for project-specific circumstances and
- Continuous knowledge accumulation and improvement of the methodology.

Each phase determines the activities that are required or suggested within the lifecycle of an agile methodology (In Figure 3, these activities are numbered). The purpose and contents of the phases and their activities are discussed in the next three subsections.

5.1 Designing an agile methodology (Phase 1)

The main objective of this phase is to design an agile methodology that will suite to organisation requirements and needs.

The first activity is examination of the organisation's existing methodology (Figure 3, Activity 1). Assuming that in any organised work elements of a methodology can be identified (at least informal elements) the scenario suggests first to study the organisations' existing process and discuss with the process participants the elements they are satisfied with and the elements they see as problematic. Particularly when the organisation members are experienced, the analysis of the existing process is essential as it helps to capture and share individuals' knowledge. Discussion with the process participants may also reveal important social elements such as organisation's culture, individuals' attitude to development process etc., which may impose significant limitations for the future methodology.

After the analysis of the existing development process the skeleton of the new methodology is determined (Figure 3, Activity 2). Beside the elements of the old methodology that have been selected as useful the skeleton has to take into consideration the organisation's characteristics and requirements. This is especially important when organisation's original development process is poor and users are not satisfied with. In such cases, the methodology skeleton can be established with taking into consideration software development methodologies/processes that are available today in the market (pre-packaged methodologies). We can choose among vast number of structural or object-oriented software development processes. However, the selection of development process, which is most suitable for an organisation, is due to a large number of candidates difficult and requires good knowledge of software development processes. To this end, the scenario uses a special tool - decision support system - that is able to tackle the problem. The system is based on a set of presumptions about correlations among the organisation characteristics and methodology characteristics. Considering the characteristic of the typical projects the company is taking, the preferences of the company (e.g. object oriented development, iterative cycle, light methodology, etc.), and the knowledge and experience of individuals from the software development team, the model suggests the kind of methodology (its characteristics) that would be for the company most appropriate. Detailed information on the decision model can be found in (Reference will be added after the review process).

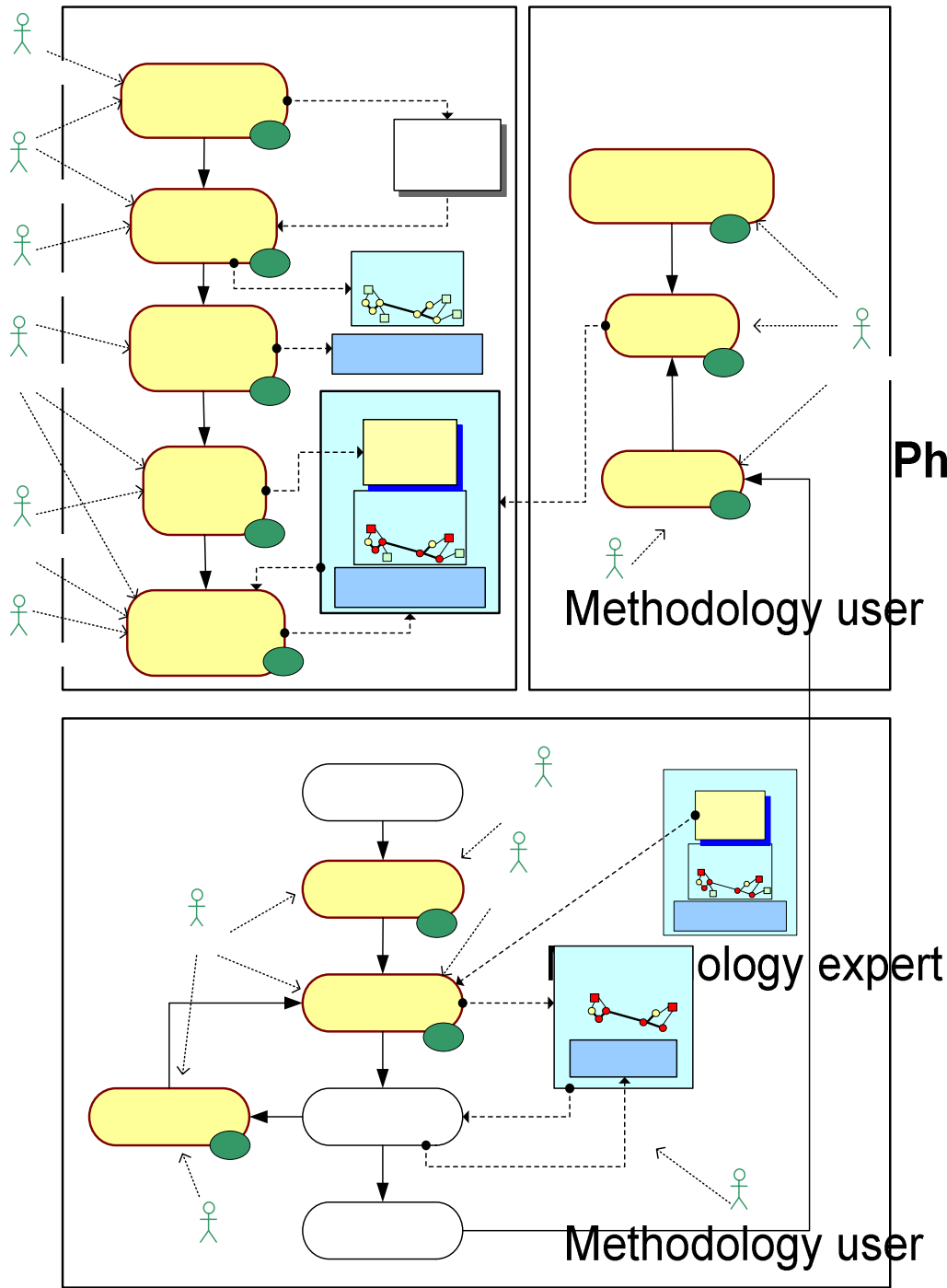


Figure 3: Scenario for development, introduction and maintenance of an agile methodology

Once the methodology skeleton is determined, its content has to be described in more detail (Figure 3, Activity 3). As discussed in section 2, it is not advisable to encode as much details as possible, but rather to document only those elements that are important and are not expected to change soon. Otherwise, the methodology may become difficult to maintain. Another important fact we have to bear in mind when describing the methodology is that thorough descriptions are not always appropriate. If an agile methodology is to be used it is expected from the users that they will be able to

Methodology expert

learn from the methodology every time they have to do something. In most cases we talk about short insight only, and not studying the methodology as such. For novices who want to learn more, the methodology should include references to sources and literature, where detailed descriptions on the methodology elements can be found.

Our experiences have shown that the presentation of a methodology in an electronic format increases its usability. We provided users with a simple web tool, which was actually the only way to access the methodology. The tool included efficient navigation and searching facilities, and most important an editor for capturing and sharing the knowledge (in form of recommendations and guidelines) individuals accumulated through the methodology use. We believe this is of utter importance for a formalised methodology to be constantly improved and aligned with the actual perception of the methodology used in an organisation (see also Figure 1).

The next activity in the first phase is the selection of the methodology construction mechanism. As described earlier, an agile methodology must be able to adapt its contents so that is best suiting to the project-specific circumstances. The way such a mechanism is implemented may vary from very simple, where optional and compulsory activities are determined to complex one, where the construction is based on semi-intelligent mechanism that uses meta-methodology as a base for an ad-hoc construction of the methodology-instance. In our real-cases the flexibility was typically introduced by the selection of the methodology fragments which were seen by the users as project-dependent and were thus determined as optional. This was the most we could get from the users.

Important for a successful implementation and use of an agile methodology are users who fully comprehend the methodology and are satisfied with all the elements the methodology is defining. It is therefore essential to educate the potential users about the methodology and assumptions and facts the methodology is based upon (Figure 3, Activity 5). Since the basic process is established considering the elements of the organisation's current methodology for which users have expressed satisfaction, there should be not many disagreements. However, if there are some dissensions within the participants, the methodology should be studied again and changed accordingly. It has to be emphasised in this stage that the scenario does not recommend anarchy but it takes as important to listen to the users who will actually use the methodology.

Main actors in the first phase are methodology expert, users of the existing methodology and a person who is selected as a methodology manager. Responsible for the analysis of the existing methodology is a methodology expert who is typically an external specialist experienced in the methodology development and use. During the analysis of the existing development process, the methodology expert interviews users of the old process, trying to acquire as much information as possible. Important role is also assigned to a methodology manager, who is responsible for the new methodology, taking care about its use, adaptations and improvements. It is important that this role is entrusted to a person who is familiar with software development methodologies and is experienced in their use.

5.2 Adapting the methodology for project-specific circumstances (Phase 2)

Once the methodology has been successfully established and discussed within its users it is prepared for use. But before it is actually applied to a specific development project it has to be adapted according to the characteristics of the project. The scenario in Figure 3 illustrates several methodological activities that have to be performed during a project lifecycle in order to make the methodology as useful as possible. After the project initiation, both project and methodology manager have to examine the project characteristics and based on that decide which elements of the basic methodology should be taken as obligatory (Figure 3, Activity 6). More the basic methodology is extensive, more is important to narrow it only to the parts that are for the project really important. Significant role here plays the mechanism that enables methodology adaptation and takes care about the elements relationships (we have to be aware that methodology elements are heavily coupled, which means that removing one could affect several others (Vlasblum et al 1995)).

Important role in this activity is assigned to sponsor of the project, who may require specific deliverables to be produced during the project.

The new, adapted methodology is now available to all users that work on the project and specifically to role managers, who are responsible for activities carried out within a specific role (e.g. Analysts, Designers, Architects, etc.). The scenario suggests establishing the role managers who are responsible to encourage the use of methodology and its continual improvement based on experience and knowledge gained during the methodology use. Note that during the execution of an iteration the role managers can change the methodology descriptions adding new guidelines and recommendations previously discussed with their subordinates.

Further use of the methodology depends on the selected lifecycle. Taking into account the fact that modern approaches in software development are almost always based upon iterative approach where the same activities are done more than once (e.g. in each iteration a part of analysis is done) the process suggests that after each iteration or step a short meeting is called to discuss the methodology use within the iteration (Figure 3, Activity 7). Project manager, methodology manager and role managers discuss the methodology usability. They do not talk about the thorough changes to the methodology but only the details which can only contribute to the methodology efficiency. If such alterations seem legitimate, the methodology is adapted (Figure 3, Activity 8). It is important to stress out however, that the changes needed for a concrete project do not affect the basic methodology.

5.3 Continuous improvement and adaptation of the methodology (Phase 3)

After the project has been finished, the methodology manager organizes a meeting with the role managers where the possible changes to the basic methodology are discussed (Figure 3, Activity 9) and implemented (Figure 3, Activity 10). Important source for improvements and changes represents the methodology manager itself whose continuous job is to investigate new trends and approaches in the field of software development methodologies (Figure 3, Activity 10).

5.4 Limitations of the scenario

The use of the scenario in practice has brought positive results. However, there are some examples in which the use of the scenario may not be successful or appropriate. These are:

- **Rigorous environments:** in rigorous environments, such as for example bank institutions, the methodology is typically extensively formalized and deterministic. This leaves fewer options for adaptability. In such environments, the principles of the agile approach do not really count.
- **Organisations that primarily outsource the software development:** The agile approach can be seen as useful directions to organisations or organisation units that develops software. In organisations where software development is typically outsourced, the point of view is different, as in fact the organisation plays the role of a customer. Even though such an organisation still requires a methodological framework to be able to carry out software purchase or outsource software development, the focus is different and the agility less important.
- **Unmotivated organisations/teams:** The use of the scenario, presented in this paper, heavily depends on the organisation/team and its attitude to the use of software development methodologies. Organisations/teams that do not value system development methodologies as contributors to successful development, may not be motivated enough to accept the scenario. As noted earlier, the method engineering has not been really accepted among practitioners in systems development, as it is often viewed as having a costly overhead in terms of time, money and people (Henderson-Sellers 2003). Indeed, organisation's and personal incentives are often the key determinant of whether the scenario will succeed.

6 SUMMARY AND CONCLUSIONS

In the paper we emphasised that a methodology is much more than just a set of methods and techniques used to achieve a certain goal but is foremost a social construct. The social component of a methodology reflects the organisation's culture and its attitude towards software development. A methodology should thus never be viewed as something independent from people for whom it is created. Another important fact in favour of a methodology being a social construct is that a methodology is always richer than its formal part. A substantial part of a methodology is embodied by its users through the knowledge and experience they carry. It is therefore important to continually evaluate the methodology and enrich its formal part with experiences, recommendations and knowledge accumulated during its use. While in literature agile methodologies are often confused with light methodologies, we have shown that the key feature of an agile methodology is not its size but a mechanism that enables ad-hoc creation of a methodology from methodology fragments based on organisation/project characteristics. The scenario presented in this paper gives a polygon for implementation of an agile methodology in an organisation, taking into account also the social aspects such as organisational culture, personal incentives and individuals' knowledge.

While the use of the scenario has shown positive results in practice it has to be stressed that it only presents guidelines which may not be applicable in any situation. Important factors that are crucial for the successful use of the scenario are skilled methodologist and motivated individuals.

References

- AGILE ALLIANCE. Manifesto for Agile Software Development.: www.agilealliance.org
- Ambler, W. S. Agile Modeling: Effective Practices for eXtreme Programming and the Unified Process. John Wiley & Sons, Inc., New York, NY, 2002.
- Ambler, W. S. The Object Primer, Second Edition: The Application Developer's Guide to Object Orientation. Cambridge University Press, New York, NY, 2001.
- Brinkkemper, S., Saeki, M., and Harmsen, F. "Assembly techniques for method engineering". In Proceedings of the Conference on Advanced Information Systems Engineering. Pisa, Italy, June 8–12. Springer Verlag, Berlin, 1998, pp. 381–400.
- Brinkkemper, S. "Method engineering: Engineering of information systems development methods and tools". Information and Software Technology. (38:4), 1996, pp. 275–280.
- Brinkkemper, S., Lyytinen, K. and Welke R.J. "Method Engineering: Principles of Method Construction and Tool Support". Selected papers from the International Conference on "Principles of Method Construction and Tool Support". Atlanta, USA, August 1996. Brinkkemper, S., K. Lyytinen, R.J. Welke (Eds). Kluwer Academic Publishers, Boston, MA, 1996.
- Brown, J. S. and Duguid. "Organizational Learning and Communities-of-Practice: Toward a Unified View of Working, Learning, and Innovation". Organization Science, 2(1), 1991, pp. 40-57.
- Cockburn, A. "Selecting a Project's Methodology". IEEE Software, (17:4), July-August 2000, pp. 64-71.
- Cockburn, A. Agile Software Development. Pearson Education, Inc., Boston, MA, 2002.
- Constantine, L. L. and Lockwood, L. A. D. Software for Use: A Practical Guide to the Models and Methods of Usage-Centred Design. ACM Press, New York, NY, 1999.
- Fitzgerald, B. "Formalized Systems Development: A Critical Perspective". Information Systems Journal. (6:1), 1996, pp. 3-23.
- Fitzgerald, B. "An empirical investigation into the adoption of systems development methodologies". Information & Management. (34), 1998, pp. 317-328.
- Harmsen, F., Brinkkemper, S. and Oei, H. "Situational Method Engineering for Information System Project Approaches". In Methods and Associated Tools for the Information Systems Life Cycle. Elsevier Science B.V., 1994, pp. 169 - 194.

- Henderson-Sellers, B. "Method Engineering for OO Systems Development". *Communications of the ACM*. (46:10), 2003, pp. 73-78.
- Heym, M., Osterle, H. "Computer aided methodology engineering". *Information and Software Technology*, (35:6/7), 1993, pp. 345-354.
- Highsmith, J. A. *Adaptive Software Development: A Collaborative Approach to Managing Complex Systems*. Dorset House Publishing, New York, NY, 2000.
- Kruchten, P. *The Rational Unified Process - An Introduction*, 2nd ed., Addison-Wesley-Longman, Reading, MA, 2000.
- Kumar, K. and Welke, R. J. "Methodology Engineering: A proposal for situation-specific methodology construction". In Cotterman, W. W. and Senn, J. A. (Eds) *Challenges and strategies for research in systems development*. Willey, 1992.
- Larman, C. *Agile & Iterative Development, A Manager's Guide (Agile Software Development Series)*. Addison-Wesley, New York, NY, 2003.
- Maier, M. and Rechtin, E. *The Art of Systems Architecting*, 2nd Edition, CRC Press, Boca Raton, FL, 2000.
- McCarty, J. and Monk, A. "Channels, Conversation, Cooperation and Relevance: All You Wanted to Know about Communication but Were Afraid to Ask". *Collaborative Computing*, (1:1), 1994, pp.35-61.
- Middleton, P. "Managing information system development in bureaucracies". *Information and Software Technology*. (41), 1999, pp. 473-482.
- Morabito, J., Sack, I. and Bhate, A. *Organisation Modelling, Innovative Architectures for the 21st Century*. Prentice Hall, Upper Saddle River, NJ, 2001.
- Nonaka, I. "The Knowledge-Creating Company". *Harvard Business Review*, (November-December), 1991.
- O'Dell, C., Grayson, C. Jr. *If Only We Knew What We Know: The Transfer of Internal Knowledge and Best Practice*, The Free Press, New York, NY, 1996.
- Ralyté, J. "Requirements definition for the situational method engineering". In *Engineering Information Systems in the Internet Context*, C. Rolland, S. Brinkkemper, and M. Saeki (Eds). Kluwer Academic Publishers, Boston, MA, 2002, pp. 127-152.
- Rolland, C. and Plihon, V. "Using Generic Method Chunks to Generate Process Model Fragments", *Proceedings of the 3rd IIE Int. Conf. ICRE'96, USA*, 1996.
- Rolland, C. and Prakash, N. "A proposal for context-specific method engineering". In *Proc. IFIP WG8.1 International Conference on "Method Engineering"*, Chapman&Hall (Pub.), Atlanta, USA, August 1996.
- Sillince, J. A. "A Model of Social, Emotional and Symbolic Aspects of Computer-Mediated Communication within Organizations". *Computer Supported Cooperative Work*. (4), 1996, pp. 1-31.
- Stolterman, E. "The 'transfer of rationality', acceptability, adaptability and transparency of methods". In W. Beates (Ed.). *Proceeding of Second European Conference on Information Systems*. Nijenrode University Press, Breukelen, 1994, pp. 533-540.
- Vlasblom, G., Rijsenbrij, D. and Glastra, M. "Flexibilization of the methodology of system development". *Information and Software Technology*. (37:11), 1995, pp. 595-607.
- Willcocks, L. P. and Sykes, R. "The Role of the IT Function". *Communications of the ACM*. (43:4), 2000. pp. 32-38.