Journal of Computing and Information Technology - CIT 16, 2008, 3, 157–168
doi:10.2498/cit.1000898

Tailoring Software Development Methodologies in Practice: A Case Study

Sergio de Cesare[1], Chaitali Patel[2], Nicola Iacovelli[3], Antonio Merico[3] and Mark Lycett[1]

[1]Department of Information Systems and Computing, Brunel University, Uxbridge, United Kingdom
[2] Agilisys Ltd., London, United Kingdom
[3] Svimservice S.p.A., Bari, Italy

Software development methodologies (SDM) have been traditionally defined in a prescriptive manner with an underlying assumption of universal applicability. However, as industrial practice suggests, this assumption is fundamentally flawed. Software development projects very rarely adopt a methodology in such a rigid fashion. Conversely, methodologies are normally adapted to meet specific contextual characteristics. This adaptation, known as Method Tailoring (MT), generally occurs implicitly. Implicit adaptation has several drawbacks. Firstly, responsibility and consequences are not attributable to the decisions made during MT. Secondly, MT experience is not captured, thus not being shared and reused within the organization. As a consequence, implicit MT leads to reactive rather than proactive adaptation with negative effects on both productivity and efficient use of resources. This paper presents a case study in which MT was applied. In order to elicit the tailoring process, a high-level conceptual framework was developed. The framework was drawn from the existing literature. As a result the know-how and experience accumulated during the practice of Method Tailoring was made explicit and organized for the benefit of future projects. The framework was applied a posteriori to a project carried out by a medium-sized software development company for the Italian national public health service.

Keywords: method tailoring, development context, reflection, experience capture, method fragments

1. Introduction

Software development methodologies (SDM) have been traditionally defined in a prescriptive manner with an underlying assumption of universal applicability. However, as industrial practice suggests [1, 2, 3, 4, 5], this assumption is fundamentally flawed. Software development projects very rarely adopt a methodology in such a rigid fashion. Initially introduced as a means to alleviate the problems of the software crisis, SDM manifested serious limitations due to the lack of any predisposition to flexibly adapt to the needs of specific projects, domains and organizational settings. As a consequence, significant undesired effects were generated. Such effects include lack of a political and organizational dimension, goal displacement, inhibition of creative thinking, and developers' resistance to change (i.e., adopting practices that are different from the familiar and consolidated ones) [6, 7]. In recent years the limitations of prescriptive methodologies have been recognized by both academics and practitioners [4]. The need to tailor methods to specific situations and context is now seen as imperative. The key issue in methodological support to software development today is flexibility and adaptation of methodologies. In response to these problems and issues, at least four different solution streams can be identified.

Firstly, a new methodology can be defined from scratch each time it is considered necessary. This represents an extreme solution and probably the less timely and most expensive [8, 9]. In addition, there is a high risk of reinventing the wheel with each methodology without harvesting the benefits of reusable knowledge and experience.

Secondly, a methodology can be selected from those available on the market or in the published literature. This is known as the Contingency Approach. This solution preserves the rigidity and inefficiencies of the previous option; moreover detailed knowledge of a wide range of methodologies would be needed in order to adequately make a sound judgment with significant effects on the cost of resources and training. Teams with specialist skills on individual methodologies would be required. Given the specific characteristics of each methodology (e.g., paradigm, notations and terminology), the knowledge transfer would be difficult as well as the allocation of resources. In addition, many authors also argue that existing rigid methods cannot adequately cover all contingencies [2, 10, 9].

Thirdly, adaptation and flexibility can be achieved through mixing and matching parts (i.e., method fragments) of different methodologies [8, 9, 11]. This is known as Method Engineering. This solution has the benefit of not constraining the development team with a predefined solution; however effort needs to be placed on harmonizing method fragments, e.g., different notations, techniques and terminology [8]. When fragments are derived from methodologies with different underlying paradigms, harmonization becomes particularly problematic and may include, for example, paradigm transformations and mappings in order to conserve desired levels of integration and traceability [12, 13, 14]. As a result, Method Engineering has not been popular in practice.

The fourth solution stream is represented by Method Tailoring (MT). MT refers to the adaptation of one methodological framework to specific software development projects. The importance of tailoring SDMs is recognized by well-known standards and reference models such as IEEE/EIA 12207.0-1996 [15] and CMMI [16]. Past studies show that in practice, organizations develop or adopt one SDM organization-wide and then tailor the method to specific projects. In particular, Guimaraes' [17] study found that 77% of the firms that use SDM employ a single, formal methodology. An essential condition for MT to be effective is the adoption of a non-prescriptive and flexible methodology. The evolution of SDM has proceeded in this direction [18, 19]. Currently, methodologies like the Rational Unified Process (RUP) and the Object-oriented Process, Environment and Notation (OPEN) Process Framework (OPF) do not prescribe the use of a particular process, but define a set of process components which can be selected and chosen to suit certain project/organizational characteristics [18]. The difference with the previous solution is the availability of process components (or method fragments) developed with the same underlying paradigm. Problems of consistent and coherent mixing and matching of method fragments (e.g., process components) are significantly reduced. Consistency in notation and terminology increases the level of knowledge sharing and reuse across teams leading to easier resource allocation. Method tailoring represents a balanced solution. It maintains the benefits of 'standardization' while it allows for controlled flexibility and adaptation to specific contexts.

MT, although in a primitive form, has been conducted implicitly by practitioners even earlier than the research literature recognized the significance of the problem area [4]. Software developers and project managers have instinctively, and sometimes subconsciously, carried out tailoring in one form or another [7]. However, various problems arise when adaptation of methods is conducted implicitly [20]. Firstly, responsibility and consequences are not attributable to the decisions made during MT. Secondly, MT experience and rationale for selecting/adapting method fragments are not captured, thus not being shared and reused within the organization. As a consequence, implicit MT leads to reactive rather than proactive adaptation with negative effects on both productivity and efficient (re)use of resources.

Modern methodologies currently provide the necessary building blocks to assemble a tailored process hence suggesting some form of contingent adaptation, however, besides flexible methodologies, software developers require practical guidance in MT [7, 11]. The limited literature on the topic provides examples of theoretical frameworks that can be applied to this problem; however, the frameworks themselves have not been empirically validated.

The aim of this paper is to present a case study in which MT was applied. The case study was analyzed through the use of a conceptual framework which was derived from the MT related literature. Generally speaking, the framework

can assist software development teams and organizations in the elicitation of their Method Tailoring processes. As a result, the know-how and experience accumulated during the practice of Method Tailoring is made explicit and organized for the benefit of future projects. The proposed framework is applied to a real-world industrial project as a means of eliciting a posteriori the tailoring process applied during the project.

The paper is structured as follows. Section 2 briefly reviews the MT related literature from which common elements are derived and used to construct the framework presented in Section 3. Section 4 presents the case study and applies the framework to it. Section 5 presents final conclusions.

2. Background

Method Tailoring frameworks serve a dual purpose. Firstly, they define the fundamental components required for tailoring SDM. Secondly, frameworks represent a means of eliciting the tailoring process applied by software development teams and organizations in industrial projects. This section carries out a brief comparative analysis of MT frameworks derived from the literature. The analysis is aimed at synthesizing previous research work in order to highlight common issues, components and relationships that will form the basis of the framework presented in Section 3.

Among the few MT frameworks that have been developed the following are considered: Harmsen, Brinkkemper et al. [8], Baskerville and Stage [21], Henninger, Ivaturi et al. [20] and Fitzgerald, Russo et al. [7]. As shown in Table 1, the analysis reveals a high degree of overlap between the frameworks, both in terms of observations and components.

The common MT components, which will be detailed in section 3, are as follows:

- Context: refers to the contextualization of the project environment.
- Method Fragments: refers to well-identifiable and named parts or components of the SDM and how such fragments are maintained and managed.
- Method Tailoring Process: refers to the process of selecting method fragments and adapting the SDM.
- Tailored Method: refers to the documentation and presentation of the tailored process.
- Experience Capture: refers to the means of reflection and capturing experience for reuse.

Although similar components are identified, two main differences characterize the MT frameworks. Firstly, the terminology used to describe the components differs. Table 1 highlights the main components (or activities) and the different terminology used to describe each concept within the component. Secondly, each author elaborates and focuses on different aspects of the MT framework. For example, [8] places emphasis on the maintenance (creation, deletion and modification) of method fragments, while [8, 22, 11] concentrate their attention on modeling tools, and [23, 24] focus on the capturing of experience. Therefore previous research tends to highlight specific areas of MT, rather than treat the process holistically.

Most of these frameworks (such as [21, 8, 9]) have been developed theoretically, based primarily on researchers' deductive observations (as summarized in Table 1), with limited exposure of the frameworks to live software development projects [21, 23, 25]. As a consequence, these frameworks remain high-level with the limitation of not providing examples of how the individual components are implemented in industrial settings.

This paper builds upon previous research work by detailing further the individual MT components in terms of questions which can help practitioners reflect upon how tailoring was carried out in individual projects as well as across projects at a wider organizational level.

3. Framework

Capturing or eliciting the process by which a software development methodology has been tailored requires systematic work. The developers or stakeholders should document the fundamental aspects of the various elements that

Name of Framework	Framework based on following observations	Different terminology used for describing each component of the framework				
		Context	Method Fragments	Experience Capture	MT Process	Tailored process
Harmsen, F., S. Brinkkemper, and H. Oei, [8] – The process of configuration of situational methods.	Methods are never followed literally; they are tuned to the situation at hand. Knowledge and experience of the project team determine the structure of the development process and the resulting products in order to deliver the desired IS. All kinds of project factors related to the technology, the development expertise, the external factors and application domain characteristics influence an approach suitable for the project. Large parts of the new methods are taken from old methods.	Project environment, Characterization of project	Method fragments, Method administration, Method Base	Project performance, Experience accumulation	Selection of fragments, Assembly of fragments, Validation, Request for adaptation	Situational methods
Baskerville, R. and J. Stage [21] – Components of a social process for Method fragment adaptation.	Methodologies are not a true representation of how systems are developed in practice. Systems development processes are emergent; they are only transient regularities in work place that are constantly shifting form. We must improve our understanding of, and means to support, the way in which development is conducted in practice.	Work setting, set of determinants of fragment selection	Method Fragments, Innovated method fragments	Capture practice	On going accommodation (Select, invent and combine)	Work practices
Henninger, S., et al. [22]	There is a need to create flexible software processes so organizational methodologies can be tailored to individual needs of projects and capture experiences that are used to refine and modify the standards. There is also a need to transform the methodologies into resources for managers and developers, to truly support the development process as it is actually practiced. The framework is based on organizational learning principles to capture and provide relevant development knowledge throughout the development lifecycle.	Elicit project characteristics	Software development resources (process framework, Tailoring criteria, Experienced bases repository, Guidelines and standards)	Experienced packages, Lesson learned	Tailored project activities	Software creation (Tailored process used, Team level review)
Fitzgerald, B., N. L. Russo, and E. Stolterman [7]	Methodologies have several pressures that support, or are against the use of methodologies. Formal methodologies are not used in practice as prescribed deliberately as they do not fit the specific project situation. Methodologies play various roles in the organization which influences the actual way methodologies are used in practice.	Development context: Problem situation and Business Opportunity	Original formalized methodology	Developers' experiences and expertise, repertoire of strategies.	Profile of development environment, developer embodied factors and roles of methodology in practice all influences method in action	Method in Action

Table 1. Comparison of Method Tailoring related literature.

have contributed to the tailoring process. Fundamental are also the decisions (and underlying reasons) as to why parts of the methodology were adopted and/or changed. Documentation of the reasons underlying such decisions is important for future development projects in order to identify possible reusable tailoring patterns. Figure 1 depicts the Method Tailoring elicitation framework applied to the case study in Section 4. The subsections that follow will describe each component along with relevant questions that can be applied to guide the elicitation of the tailoring process used in software development projects.

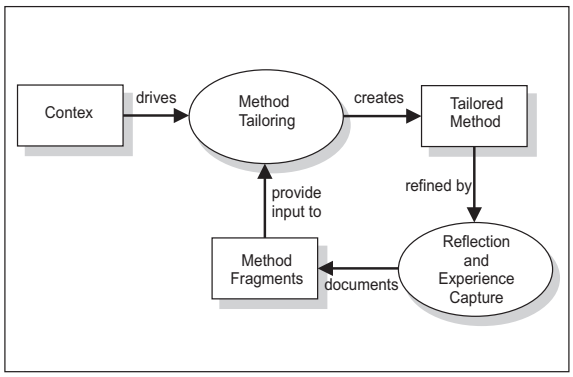

Figure 1. Method Tailoring elicitation framework.

3.1. Context

In the process of method tailoring, the starting point is a given, dynamic and evolving environment which is part of a larger organizational setting. This component is known as context. Fitzgerald et al. [7] consider context as a given that cannot be changed and is an input to the software development process. Fitzgerald et al. and Harmsen et al. [7, 8] consider the project environment to include both the supplier organization as well as the customer organization. A distinction is also made between the project environment and characterization of the project. They state that the project environment includes existing information infrastructure, the users, and the organizational culture, while project contingency factors, such as application characteristics, external factors, technical factors, and the available development expertise are in some way determined forming project characterization.

MT Component	Topics/Scope	Related Questions
Context (contextualization of the project environment)	Scope of context, capturing context.	Organizational: size, types of applications developed, team dynamics/structure, etc. Project: size and complexity of project, type of application, etc. What contextual factors have significantly affected the tailoring process?

Table 2. Context.

Context in this framework includes characteristics that affect the selection and adaptation of method fragments. Glass [4] considers size, application domain, criticality and innovativeness factors that differentiate projects and captures context. Similarly, Cockburn [12] also considers size, criticality and project priority as differentiating project factors. Based on the comparative analysis of the literature, the context component in this framework has four categories of factors that influence software development. These broad categories are: organizational characteristics, team dynamics/structure, project characteristics and product characteristics. Methods are tailored to a particular context; hence it is important to investigate how this context is captured and defined in practice.

3.2. Method Fragments

This component is often described by using the building blocks analogy. It is widely accepted that in order to tailor methods, methods need to be composed of method fragments. A method fragment can be considered as a well-identifiable part of a SDM such as an artefact (diagram, model or document), role, technique, tool process workflows, activities or tasks or a coherent modularized set of any of the above.

Method fragments provide the advantages of standardization as they are described with the same notation and terminology. The degree of flexibility of a SDM has great impact on how easy or difficult tailoring is. One way to gain or achieve tailorability within SDM is to modularize SDM fragments for flexibility [2]. Keller [26] also discusses the modularization

of fragments as a prerequisite to tailor methods. Method Fragments need to also have other characteristics that allow them to be assembled to form tailored methods. Modularized method fragments need to be loosely coupled to each other and highly cohesive [26, 25].

MT Component	Topics/Scope	Related Questions
Method fragments and maintenance	Addition, deletion and modification of method fragments. Modularity of fragments. Case tool support: Repository or method base for storing method fragments, retrieving and assembling method fragments.	What methodology does your organization use? Is it prescriptive? Does it have modularized method fragments that you can select? Are they loosely coupled and highly cohesive? Do they use the same notation and terminology?

Table 3. Method fragments.

3.3. Method Tailoring Process

This component refers to the actual process of selection and assembly of method fragments to fit the particular context. In the elicitation of the tailored method, it is important to identify how tailoring occurs in relation to the context (both project and organization). This implies understanding how method fragments are selected including what factors affect the addition, removal or modification of fragments. If in the organization method tailoring is formalized in any way, then key questions concern the roles and responsibilities of those involved in the tailoring process and the way decisions are documented and maintained for future memory.

MT Component	Topics/Scope	Related Questions
Method Tailoring Process	Incremental ME, selection and adaptation of method fragments to project characteristics, case tool support.	Is MT done? Is MT seen as an important activity or a negative activity as in not following the process rigorously? How do you tailor your process based on the context you have and the Method fragments? How are these method fragments selected in practice and how are they assembled? Who does the tailoring and when is it done? Have you formalized the method tailoring process? Does your organization have tailoring guidelines that you can follow?

Table 4. Method tailoring process.

3.4. Tailored Method

The tailored method is the result of MT. This includes all elements that would be typically found in a SDM. Given its application in a specific project, as well as future projects that share similar characteristics, it is essential to present, document and store the tailored method so that all interested parties can access and retrieve it. The format with which the tailored method is presented is essential to its future reusability and therefore acts as a template in other projects.

MT Component	Topics/Scope	Related Questions
Tailored Method	Ability to reuse the tailored process, case tool support.	How is it applied and updated? How are templates produced, used and adapted?

Table 5. Tailored method.

3.5. Reflection and Capturing Experience

All the frameworks analyzed from the literature were based on one same observation; people in practice tailor methods based on past experiences. Hence it is important to capture this experience for others to reuse. This enables an organization to learn from past successes and mistakes. This component deals with the issue

MT Component	Topics/Scope	Related Questions
Capturing experience	Reflection, capturing method rationale, review process, reuse, case tool support.	What were the lessons learned in terms of benefits and limitations derived from the tailoring experience of the project? How is experience captured and preserved?

Table 6. Capturing experience.

of capturing method tailoring experience and presenting it in a reusable way so others can use the information. For example, capturing rationale for selecting certain fragments is also important, as this helps gather knowledge so team members can use and learn from others experiences.

4. Case Study and Application of Framework

The framework described in the previous section was applied to a project carried out at S-Service, a medium-sized software development company located in southern Italy. S-Service is specialized in the development of information systems for the healthcare sector and local public administrations (PA).

This case study documents a large development project carried out for the Italian National Health Service (INHS). The project is aimed at the realization of a series of 'software services' strategically intended to improve the quality of service of the INHS allowing medics, healthcare staff and citizens to directly interact with local and regional health structures through the Internet. The services were called Network Application Services (NAS). Ten NAS were defined and allocated to distinct subprojects. The case study therefore refers to a coordinated project which was divided into ten subprojects. From a technological perspective, the NAS were based on web services.

The SDM adopted for this project was the Rational Unified Process (RUP). RUP was tailored to suit specific contextual organizational and project characteristics. The tailoring process was elicited a posteriori by applying the framework proposed in this paper. The elicitation of the tailored process was conducted manually and documented in specific artifacts/reports such as the quality plan. The following subsections are structured in line with the framework components and describe S-Service's experience in tailoring this specific project. Due to limitations of space, a representative extract of the case study documentation is presented. As the case study will demonstrate, the framework contributed in the following areas: (1) provided the project team and the organization with a structured means of eliciting the tailoring process; (2) allowed the development team to document the experience of the project for the benefit of future development and maintenance efforts and (3) facilitated reflection and constructive self-criticism.

4.1. Context

RUP was originally adopted by S-Service in 2002. The decision to adopt RUP was strongly influenced by the emergence of object-oriented methodologies on the market. Given RUP's characteristic of being a methodological framework rather than a rigid and binding methodology, decisions were taken at the beginning of the project as to the processes, roles and artifacts that would be applied and managed. With the experience previously acquired, the project team adopted a tailoring process that had evolved from previous projects.

Contextual factors that most affected method tailoring for this project were as follows:

- Team structure: presence of staff/team members highly competent in the domain area with detailed knowledge of the business processes and regulations of the public healthcare sector.
- Project characteristics: domain and type of client. In the Italian public healthcare sector, as with all public bodies, contracts are acquired through a request for proposals and a public bidding process; hence very formal procedures, which require precise and detailed documentation of the system to be delivered.
- Product characteristics: there were requirements of Enterprise Application Integration (EAI) with existing legacy systems. This factor influenced the whole lifecycle which needed to take legacy and integration requirements into consideration.

From past tailoring experiences at S-Service, tailoring at the organizational level involved adjusting the method to (1) the types of applications normally developed (e.g., business applications), (2) the organizational models and/or (3) the technological architectures used. Considering that the objective was to obtain a method that was also applicable to most of the company's projects, tailoring here mainly regarded

the elimination of those components that were not likely to be applied (e.g., components specialized for the development of real-time software).

4.2. Method Tailoring Process

The tailoring process was driven by the Quality Assurance (QA) team in collaboration with the development team. A series of meetings took place at which tailoring decisions were made. These decisions were then formally documented in the Quality Plan. The Quality Plan defines (for each individual project) the overall method adopted. The tailoring process was initiated at the beginning of the project. Previous experiences suggested that the tailored method could be either entirely decided at the start of the project or the method could be refined across various iterations as development progressed. A decision was made to define all workflow details and activities before the corresponding workflow took place. This decision was also based on cost/benefit analyses carried out in order to assess the effect of an individual workflow and its activities on the project as a whole.

Based on S-Service's past experience, it can be noted that tailoring most frequently occurred at the project level, instead of, for example, at the organizational or departmental levels. Generally speaking, project level tailoring is mainly aimed at adapting the method to specific project characteristics. Often a method that is tailored for a project can be used as the basis for developing future projects with similar characteristics, such as case tools, technological architecture and development environment. In this situation, a method tailored at a project level has the potential of being used at a departmental level and therefore covering a class of common projects.

In the case study, method tailoring was a part of the overall development planning. As far as tailoring was concerned, planning involved:

- Possible elimination of some of the original method activities/tasks
- Addition of new activities/tasks;
- Adjustment (tweaking/redefinition) of activities/tasks to suit the project characteristics;
- Definition of checks (e.g., reviews/inspections) on the deliverables aimed at assuring correctness, consistency and traceability of requirements all the way down to implementation.

The above tailoring activities were motivated and documented in the Quality Plan. Within the Quality Plan reference was also made to other appropriate documents (e.g., naming standards document, planning document, etc.). The main contextual adjustments carried out during the project were as follows:

- The Business Modeling workflow was not applied in this project due to the previous in-depth knowledge that the project team had of the healthcare domain. The team was also very familiar with the procedures in place in order to obtain contracts from the National Health Service.
- The level of technical detail of the documentation was tailored to meet the specific requirements of the customer (e.g., intermediate revisions, additional support activities for testing, etc.)
- The software lifecycle model that was defined took into account integration requirements with existing legacy systems.

4.3. Tailored Method

As previously mentioned, Method Tailoring was mainly documented in the Quality Plan. Other documents were also used. For example, the Configuration Management Plan documented, among other things, how software and documentation should be configured, versioned and so on.

For this project the Quality Plan documented the following elements:

- Selected RUP workflows: Support and Management workflows (e.g., configuration management, project management, testing, etc.) were not selected since the corresponding processes were already catered for in the company's standard procedures.
- A table was prepared in which the following information was reported for each workflow:
  - Workflow: Name of RUP workflow

  - Activity: Name of workflow detail activity
  - Objective: Goal of the activity or workflow as it is defined by the method
  - Inclusion: Specifies whether an activity has been included in the tailored process. If an activity was not included, the reasons were given in the Notes column.
  - Output: Outputs of activities and workflows were defined (i.e., artifacts)
  - Notes: Annotations, suggestions, reasons were annotated
- Checks to carry out design and implementation artifacts aimed at verification (consistency between inputs and outputs of each phase, traceability checks, respect of standards, etc.) and validation (software testing with appropriate procedures). The checks were tailored depending on the deliverables produced and the tools used (e.g., Rational Rose).

The RUP deliverable templates were examined and adapted to the needs of the project; therefore producing tailored templates that were adopted by each subproject team.

Scheduling of the tailored project activities was documented in the Project Plan. The Project Plan defined:

- Roles and skills required to carry out the activities
- Resources required by the project (human, tools, infrastructure)
- Scheduling (e.g., Gantt chart)
- Monitoring times and metrics
- Predicted and actual project costs
- Evaluation (both technical and managerial) of project activities aimed at highlighting critical aspects.

The plans were treated as 'living' documents, i.e. they were updated during the project in order to correct inadequate situations or to prevent critical situations caused by erroneous tailoring decisions.

4.4. Reflection and Capturing Experience

According to the organization's standard procedures, the Project Manager is required to write a 'Project Closure Report' documenting the overall project experience. The report is submitted to Senior Management and it contains an overall analysis of the project including decisions made, reasons for deviating from predefined standards and objectives. In this case study, an evaluation of the tailoring decisions made was included in the report. The report was accompanied by a set of metrics that serve as indicators of project success.

At present, these metrics are recorded on paper documents. It is intended by the organization to develop a repository and populate it over time with information on all projects. This repository would support decisional processes (which methods to adopt, tailoring, etc.) and evaluations (benchmarking) for the development and maintenance of future projects.

Various benefits of MT have been identified and documented for this project. These benefits can be summarized as follows:

- Consistency of the development process within the same project;
- Consistency of the techniques used by the team members (same training, reduction in communication overheads, standard communication protocols);
- Consistency of artifacts, models and standards;
- Potential reuse of the same process in similar projects (a sort of process polymorphism) by simply tweaking where necessary;
- Benchmarking and measurements applied in one project can be used as the basis for predictions in other projects.

4.5. Method Fragments and Maintenance

Reuse at a project level is applied when a new project is started and the decision is taken to adopt a method previously used by the organization. The project benefits in this case from past experience and previously tailored and documented methods. The latter (along with the Project Manager's final evaluation) represent a point of reference for the project's processes

and deliverables as well as support for the initial definition of the project's activities.

Generally speaking, for every tailored development project the following elements are "instantiated": (1) a configuration library containing software artifacts and documentation and (2) a "Development Standards" document defining for the specific project standards and techniques that analysts, designers and programmers must adopt. The "Development Standards" document generally includes a description of the repository's logical structure. The repository contains workflow artifacts and a description of a "generalized method" that that can be specialized if required. In this case study, for example, the repository would describe: RUP related packages and artifacts, naming conventions of packages and artifacts, package contents and relationships between packages.

The Development Standards document also defines development patterns that must be adopted for the realization of different architectural components. For example, the Model-View-Controller pattern used for the Web tier of applications and a formal description of how such a pattern should be applied. There normally is one Development Standard document for each production line and referenced in each project. A production line is characterized by the same methodology, the same development technology and the same software architecture.

5. Conclusion

This paper presented a case study of a large-scale software development project in which the Rational Unified Process was tailored to suit the specific contextual characteristics of the organization, project team, client and application type. A Method Tailoring framework derived from the literature was adopted to conduct the research.

The main findings of the case study were as follows. The organization had previous tailoring experiences; as a consequence, the MT process was formalized in some aspects (e.g., defined tailoring roles and responsibilities along with formal documents in which decisions were annotated) and not fully mature in other aspects (e.g., repository management and benchmarking required more growth). From a procedural perspective, tailoring was carried out iteratively (for most part) and conducted hierarchically. Three levels were identified: organizational, departmental and project. In the project described in Section 4, the project itself was divided into ten subprojects. Even in this case successive refinements were applied. The initial subprojects served as pilots for the following ones.

At a project level, methods at S-Service are selected from a repository of previously tailored RUP methods. The selection is based primarily on application type, system architecture and case tools adopted. The repository contains generalized methods that can be specialized (through a form of process 'polymorphism') to similar projects.

In terms of the MT framework presented in this paper, its validity is twofold. Firstly, it is grounded in previous research work documented in the literature. Secondly, it has been applied to document a large-scale industrial case study. Although the work presented in this paper provides a contribution to the area of Method Tailoring, it does nonetheless have its limitations.

At this stage of the research, investigations have been conducted only on one case study. It is desirable to explore further development organizations with different contextual characteristics (e.g., size) in order to identify commonalities and differences with the organization and project described in this paper. These further case studies would highlight, among other things, how the dimensions (medium, large and very large) affect MT decisions and processes. Future research would allow for a further refinement of the framework. These relate to expanding each framework component (e.g., detailed questions and identification of further relationships between the components). This expansion would positively affect the work presented in this paper.

References

[1] F. P. BROOKS, No Silver Bullet – Essence and Accidents of Software Engineering. *Computer*, **20** (1987), pp. 10–19.

[2] J. CAMERON, Configurable development processes. *Communications of the ACM*, **45** (2002), pp. 72–77.

[3] B. FITZGERALD, Formalized systems development methodologies: A critical perspective. *Information Systems Journal*, **6** (1996), pp. 3–23.

[4] R. GLASS, Through a Glass, Darkly. Methodologies: Bend to Fit? *The Software Practitioner, Data Base Advances*, **27** (1996), pp. 14–16.

[5] C. J. HARDY, J. B. THOMPSON, AND H. M. EDWARDS, The use, limitations and customization of structured systems developement methods in the United Kingdom, *Information and Software Technology*, **37** (1995), pp. 467–477.

[6] D. E. AVISON AND G. FITZGERALD, *Information Systems Development: Methodologies, Techniques and Tools*. Third ed.: McGraw Hill, 2003.

[7] B. FITZGERALD, N. L. RUSSO, AND E. STOLTERMAN, *Information Systems Development: Methods in Action*. McGraw Hill, 2003.

[8] F. HARMSEN, S. BRINKKEMPER, AND H. OEI, Situational Method Engineering for Information-System Project Approaches. In *Methods and Associated Tools for the Information Systems Life Cycle*, **55** (1994), Ifip Transactions a-Computer Science and Technology, pp. 169–194.

[9] K. KUMAR AND R. J. WELKE, Methodology engineering: a proposal for situation-specific methodology construction. In *Challenges and Strategies for Reseach in Systems Development*, W. W. Cotterman and S. J. A., Eds.: John Wiley, 1992.

[10] B. FITZGERALD, N. L. RUSSO, AND T. O'KANE, Software development method tailoring at motorola. *Communications of the ACM*, **46** (2003), pp. 64–70.

[11] A. COCKBURN, *Agile Software Development*. Addison Wesley, 2002.

[12] J.-P. TOLVANEN, Incremental Method Engineering with Modeling Tools. In *Department of Computer Science and Information Systems*, Finland: University of Jyvaskyla (1998), pp. 301.

[13] D. G. FIRESMITH AND B. HENDERSON-SELLERS, *The OPEN Process Framework*. Addison Wesley, London, 2002.

[14] A. H. M. TER HOFSTEDE AND T. F. VERHOEF, On the feasibility of situational method engineering. *Information Systems*, **22** (1997), pp. 401–422.

[15] IEEE/EIA, Industry Implementation of International Standard ISO/IEC 12207: 1995. Software life cycle processes. March 1998.

[16] M. B. CHRISSIS, M. KONRAD, S. SHRUM, *CMMI. Guidelines for Process Integration and Product Improvement*, Addison-Wesley, Boston, MA, 2003.

[17] T. GUIMARAES, A study of application program development techniques. *Communications of the ACM*, **28** (1985), pp. 494–499.

[18] B. HENDERSON-SELLERS, Method engineering for OO systems development. *Communications of the Acm*, **46** (2003), pp. 73–78.

[19] G. J. HIDDING, Reinventing Methodology: Who Reads It and Why? *Communications of the Acm*, **40** (1997), pp. 102–109.

[20] M. LYCETT, R. MACREDIE, C. PATEL, AND R. PAUL, Migrating Agile Methods to Standardized Development Practice. *Computer*, **36** (2003), pp. 79–85.

[21] R. BASKERVILLE AND J. STAGE, Accommodating Emergent Work Pratices: Ethnographic Choice of Method Fragments. Presented at *IFIP TC8/WG8.2 Working Conference on Realigning Research and Practice in Information Systems Development*, Boise Idaho, 2001.

[22] S. HENNINGER, A. IVATURI, K. NULI, AND A. THIRUNAVUKKARAS, Supporting Adaptable Methodologies to Meet Evolving Project Needs. Presented at *XP/Agile Universe*, 2002.

[23] S. HENNINGER, Turning Development Standards Into Repositories of Experiences. *Software Process Improvement and Practice*, **6** (2001), pp. 141–155.

[24] M. ROSSI, J.-P. TOLVANEN, B. RAMESH, K. LYYTINEN, AND J. KAIPALA, Method Rationale in Method Engineering. Presented in the *Proceedings of the 33rd Hawaii International Conference on Systems Science*, Hawaii, 2000.

[25] M. I. KELLNER, L. BRIAND, AND J. W. OVER, A method for designing, defining, and evolving software processes. Presented in the *Proceedings of the Fourth International Conference on the Software Process*, 1996.

[26] M. I. KELLNER, Connecting reusable software process elements and components. Presented in the *Proceedings of the 10th International Process Support of Software Product Lines*, 1996.

[27] I.-C. YOON, S.-Y. MIN, AND D.-H. BAC, Tailoring and verifying software process. Presented at *Eighth Asia-Pacific Software Engineering Conference (APSEC'01)*, Macao, China, 2001.

Received: July, 2006
Revised: May, 2008
Accepted: May, 2008

Contact address:

Sergio de Cesare
Mark Lycett
Department of Information Systems and Computing
Brunel University
Uxbridge, Middlesex, UB8 3PH, United Kingdom
e-mail: sergio.decesare@brunel.ac.uk
mark.lycett@brunel.ac.uk

Chaitali Patel
Agilisys Ltd.
London, United Kingdom
e-mail: chaitali.patel@agilisys.co.uk

Nicola Iacovelli
Antonio Merico
Svimservice S.p.A.
Bari, Italy
e-mail: nicola_iacovelli@svimservice.it
antonio_merico@svimservice.it

SERGIO DE CESARE holds a PhD in Information Systems from LUISS Guido Carli in Rome (Italy). He is currently a lecturer at Brunel University where he teaches software modeling and Semantic Web technologies. Sergio's broad research interests are in the areas of business and software modeling, model-driven information systems development and the Semantic Web. His current research focuses on the development of ontological models for systems development/re-engineering and the subsequent transformation of such models into platform-independent and platform specific application models. Sergio has authored several papers published in international journals and conference proceedings related to modeling in software development.

CHAITALI PATEL holds an MSc in distributed information systems from Brunel University (West London). She is currently a Project Manager at Agilisys Ltd. in London where she manages software development and integration projects. Prior to this, Chaitali was a Software Process Improvement Consultant at CapGemini (UK.) Chaitali has a keen interest in software development methodologies and frameworks (particularly agile software development) and method tailoring. She has published a number of papers and presented in conferences and workshops in these areas.

NICOLA IACOVELLI holds a degree (with honors) in computer science from the University of Bari (Italy). From 1995 he works at Svimservice, a company of 250 employees located in Bari, which produces software solutions in the areas of Healthcare and the Public Administration. Nicola is Quality Manager at Svimservice where he manages the study and adaptation of software development methodologies, with particular reference to development/maintenance, project management, quality control, testing, estimation and measurement of software. Nicola is a member of the IEEE Computer Society, ACM and Gufpi-Isma (Gruppo Utenti Function Point Italia - Italian Software Metrics Association) with which he collaborates as a member of the SBC (Software Benchmarking Committee). He is affiliated with the UNINFO (Standards for the Information Technology and Related Applications, associated body of UNI) acting as an expert reviewer of technical standards.

ANTONIO MERICO holds a degree in computer science from the University of Bari (Italy). He has previously conducted research in the area of statistical models for query optimization in database systems. His current research interests include methodologies for the development of web services-based applications, function point analysis and architectures. Antonio is currently Software Factory Manager at the Health Information Systems Department of Svimservice, a software solutions company located in Bari. He plans and coordinates activities of analysis, design and implementation of health information systems with different methodologies, architectures and technologies.

MARK LYCETT holds a BSc degree in computing and business management (Oxford Brookes), a MSc degree in information systems (Brunel University) and a PhD degree in information systems (Brunel University). Prior to returning to education, Mark spent a number of years in industry and he has both worked on and managed a number of national and international feasibility/development projects. His research concentrates on all aspects of component-based software engineering and he is currently engaged in ongoing research with a number of organizations. Mark has published work in the area of Component-based Software Engineering (CBSE) in a number of leading journals and international conferences.