A Lightweight Workshop-Centric Situational Approach for the Early Stages of Requirements Elicitation in Software Development

Chad Coulin [a,b] , Didar Zowghi [b] and Abd-El-Kader Sahraoui [a,c]

[a] LAAS CNRS, 7 avenue du Colonel Roche, 31077 Toulouse France

[b] University of Technology Sydney, PO Box 123, Broadway NSW 2007 Australia

[c] IUT de Blagnac, 1 place Georges Brassens, 31703 Blagnac France

Abstract

The elicitation of requirements for software systems is one of the most critical and complex activities within the development lifecycle. Although the subject has received some degree of attention in the research literature, there remains a need for situational methods and processes that can be easily utilized by the majority of practitioners in typical projects. In this paper we present a flexible, yet systematic approach to the early stages of requirements elicitation based on collaborative workshops, and the construction of a lightweight situational method within a general process framework. The research provides practitioners with an approach to requirements elicitation that can be readily applied to real-world projects in order to improve both the process and results. It also offers researchers an example of how lightweight situational method engineering can be applied to very practical activities and situations.

Keywords: Requirements, Elicitation, Situational, Workshop

1 Introduction

Within the software development lifecycle, requirements elicitation is the activity typically performed after project initiation and preliminary planning, but before system conception and design. Subsequently requirements elicitation can be broadly defined as the operations related to the acquisition and elaboration of goals, constraints, and features for proposed software-based systems by means of investigation, exploration, and analysis. Furthermore it is generally understood and accepted that requirements are 'elicited' rather than just captured or collected. This implies discovery, development, and creativity elements to the process. Similarly, Hickey and Davis (2003) have defined requirements elicitation as "learning, uncovering, extracting, surfacing, and/or discovering needs of customers, users, and other potential stakeholders".

Widely regarded as one of the more challenging activities within the scope of Requirements Engineering (RE), elicitation by its very multi-disciplinary nature is subject to many factors and a variety of both technical and social issues. When performed poorly, the results regularly include costly rework, schedule overruns, and in some cases, project and system failure (Hickey and Davis, 2002). Often expensive in terms of time, effort, and cost, it is therefore even more important that the elicitation of requirements is conducted properly and with the appropriate level of rigor. The production of high quality requirements through effective elicitation is absolutely essential for the engineering of successful software products.

One of the major problems in requirements elicitation today is the significant gap between expert and novice analysts. This can be attributed to a number of factors, not least of which is the extensive skill set and range of experience necessary to perform

this activity successfully. True experts in this complex and critical activity are few and far between. This situation is further aggravated by the current lack of systematic guidelines and flexible methods. Somewhat related and just as important is the current gap between requirements elicitation theory and practice. A lack of awareness by novice analysts of thee state of the art techniques and tools for requirements elicitation, combined with a general unwillingness to adopt them is largely responsible.

As a result the objectives of the research described in this paper are specifically focused on filling the void that currently exists by providing support for novice analysts and those projects without a prescribed software development methodology, with a useful and useable situational approach to requirements elicitation. Within this approach, we wish to take advantage of the collaborative aspects of group workshops, as well as the benefits in combining a number of different techniques, whilst attempting to directly address some of the current issues and challenges faced by analysts.

Our primary goal, however, is to improve the effectiveness and efficiency of the requirements elicitation process in terms of time, costs, effort, and quality of results, thereby increasing the likelihood of project success with respect to on-time and on-budget delivery, while achieving customer and user satisfaction. The focus of this work is principally on the early stages of requirements elicitation, and not other often associated RE activities such as modelling and prioritization. Instead we concentrate primarily on the fact-finding and information gathering operations.

The paper is structured as follows. Section 2 describes some background information on requirements elicitation and related research. Section 3 introduces the topic of situational method engineering and how it can be related to requirements elicitation. Section 4 presents the developed approach to requirements elicitation including the situational method and process framework. Section 5 demonstrates the implementation of the approach by way of a real-world software development project example, and Section 6 provides a discussion of the approach and its potential implications. Finally Section 7 presents some conclusions from the research, and Section 8 suggests future work to be done.

2 Requirements Elicitation

Much of the research performed over the past two decades on and around the topic of requirements elicitation for software systems has been focused primarily on the development, implementation, and evaluation of a variety of techniques, methods, and tools. Many of these were adopted from other disciplines such as the social sciences (Ball and Ormerod, 2000; Beyer and Holtzblatt, 1995) and knowledge engineering (Hudlicka, 1996; Maiden and Rugg, 1994). Regardless of their origin, the principal motivations for these approaches were to reduce the complexity of the process, improve the quality of the results, and address some of the major issues often experienced by practitioners.

In reality there are literally hundreds of approaches that can be used for requirements elicitation. A survey by Goguen and Linde (1993) examined at a relatively high level only a small number of the more traditional techniques such as interviewing, observation, and task analysis. In a more recent survey on the theory and practice of requirements elicitation (Zowghi and Coulin, 2005), several additional and more current approaches were examined including those based on goals (Dardeene et al., 1993), scenarios (Potts et al., 1994), viewpoints (Sommerville et al., 1998), and do-

main knowledge (Sutcliffe and Maiden, 1998), combined with a review of the tasks, sources, and types of information involved.

Of the many available techniques, collaborative approaches to requirements elicitation, and especially facilitated workshops, where the different stakeholder groups are represented and work together to collectively conduct the process of requirements elicitation, are a common and often default approach. These have also been found to be very helpful and successful in not only producing quality requirements, but also in achieving stakeholder buy-in and instilling project ownership (Gottesdiener, 2002). One of the obvious advantages of using group workshops is the ability to integrate other elicitation techniques into them, and then to incorporate their combined usage into a defined requirements process (Maiden et al., 2004).

As a result of the relative strengths and weakness of the available approaches, most projects will normally require a combination of several techniques in order to produce quality results (Maiden and Rugg, 1996). Although over the years a number of different process models have been proposed as generic roadmaps to address this (Sommerville and Sawyer, 1997; Kotonya and Sommerville, 1998), and significant progress has been made in developing better techniques to support the process of eliciting requirements for software systems, there still remains a lack of appropriately flexible guidelines and sufficiently detailed methods which can be used by the majority of practitioners in typical projects.

Clearly requirements elicitation does not occur in a vacuum and is highly dependant on specific project, organizational, and environmental characteristics (Christel and Kang, 1992). Software can be developed in a variety of contexts such as bespoke systems from scratch, or based on the customization of a COTS solution. The development may be contracted out to an external entity, or performed by internal personnel. Software may be produced for the general market, or for an individual customer. Requirements elicitation itself may be part of a feasibility study, a COTS selection process, or the development of an entirely new system. Furthermore there are many kinds of systems that may be developed including distributed, web-based, and embedded just to name a few. The possible permutations of these situational characteristics are numerous. Moreover, there are number of other internal and external factors that may affect the project and how it is conducted including government regulations, changing market conditions, political considerations within the organization, and the technical maturity of the organization and the users of the target system.

Requirements elicitation is an absolute prerequisite for all the different types of software development projects. It is precisely because of the many and varied contexts in which software development is performed, and the large number of factors, techniques, and issues that may have an affect and are involved even in the most typical of processes, that a situational method is not only suitable, but essential. Furthermore it is proposed that the integration of such a method with collaborative workshops and a combination of techniques provides an innovative and productive approach to requirements elicitation for the development of software systems.

3 Situational Method Engineering

Method Engineering (ME) represents a structured way in which methods for software development activities such as requirements elicitation can be designed, constructed, and adapted. Situational Method Engineering (SME) is therefore the configuration of

these methods specifically for individual projects (Brinkkemper, 1996). Naturally this is an important topic as no two software development projects are exactly the same, and all projects cannot be adequately supported by a single static method. This is especially the case with requirements elicitation, where the heavy dependence on human involvement adds a significant number of social and communication variables.

The idea of creating a situational method for requirements elicitation is not new. As early as 1982, Davis identified the need and importance of developing situational requirements elicitation methods (Davis, 1982), although he referred to them as "requirements determination strategies". However the basic premise of characterizing a specific project based on some criteria, and as a result selecting from a set of methods those that are most appropriate, remains the same. Davis also suggests that the development of such a method should be based on the types of information to be elicited.

The scope of ME research for software development in recent times has included models for situational method engineering (Ralyté et al., 2003), assembly techniques and rules for method construction (Ralyté and Rolland, 2001; Brinkkemper et al., 1998), and more generic process modelling and engineering approaches (Henderson-Sellers, 2002). The combined result of this work over the past ten years in particular provides a suitable foundation for the development of an activity specific situation method as proposed in this paper for requirements elicitation. Of specific interest and relevance to the objectives of our research is the work by Henderson-Sellers and Firesmith relating to the OPEN Process Framework (OPF, 2005), which has been applied to the entire software development lifecycle, as opposed to just the elicitation of requirements as is our case.

Requirements elicitation alone, however, represents an excellent candidate for the implementation of a situation specific method. As it is usually the first phase of a development project, it does not have to conform to the assumptions or constraints of other methods employed in previous phases, or rely on the outputs from other activities. The flip side to this, however, is that at the requirements elicitation stage, very little may actually be known about the project and the target system.

4 The Requirements Elicitation Approach

The following situational approach has been developed based on a critical analysis of the state of the art in requirements elicitation, as well as an ongoing survey of practice. The approach represents not only a detailed process for conducting requirements elicitation, but also an integrated situational method. The overriding intention of this approach is to support the analyst structure and perform requirements elicitation workshops based on project specific characteristics.

Throughout the approach we have purposely endeavoured to keep the process of engineering the method, and the process of requirements elicitation itself, as lightweight as possible. This is in direct response to our objective of providing support for novice analysts and projects without a specifically prescribed software development process.

4.1 Meta-levels of the Approach

The general approach can be explained using the three-tiered structure as seen in Figure 1 (adapted from Henderson-Sellers, 2002) which differentiates between a method meta-model, a specific instantiation of that method meta-model, and an individual

project specific instance of a specific instantiation. This can be described in more basic and practical terms by the following example. L2 represents the components and rules for constructing a requirements elicitation process for an organization that produces different types of software. L1 then represents one possible example of a process, constructed using the components and rules set out in L2, for say the development of customized business information systems. L0 would then represent a project specific enactment of that process detailing the requirements, goals, and constraints of the business information system for that particular customer.

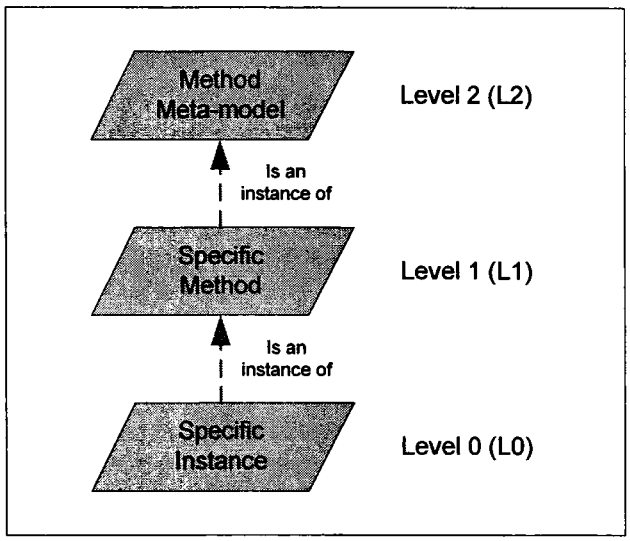

Figure 1: Approach meta-levels

The top level of this structure (L2), from which processes are constructed, consists of four main 'meta-types' or 'classes' as shown in Figure 2 and described in Table 1 below. In fact instances of these meta-types (Info Types, Tasks, Sources, and Techniques), which are sometimes referred to as 'method fragments' or 'method chunks', are just the building blocks used to construct what we have called 'method components', and it is actually a set of these method components that are selected and sequenced to form a specific method (L1) which can then be enacted (L0).

Table 1: Method meta-model class descriptions

Class	Description
Info Type	A specific kind of required data or knowledge, such as 'project assumptions' and 'design constraints'
Task	A specific individual unit of work. Examples include 'identify key constraints' and 'define work processes'
Source	A place or object that provides information. Sources can be individuals, groups, documents, and systems
Technique	A specific way of performing a requirements elicitation task, such as by questionnaire or brainstorming

The term 'method component' has been introduced and used to describe a single instance of the entire method meta-model based on an individual task. Therefore a method component is a method fragment of the Task class with its corresponding and related Info Type, Source and Technique method fragments. Specific real-world examples of method components are given in Section 5. In summary, a method component is a task that elicits an info type from a source using a technique. These method components are used as the base unit of work to organize and execute requirements elicitation within a workshop environment, by placing them into a structured sequence according to the process framework described later.

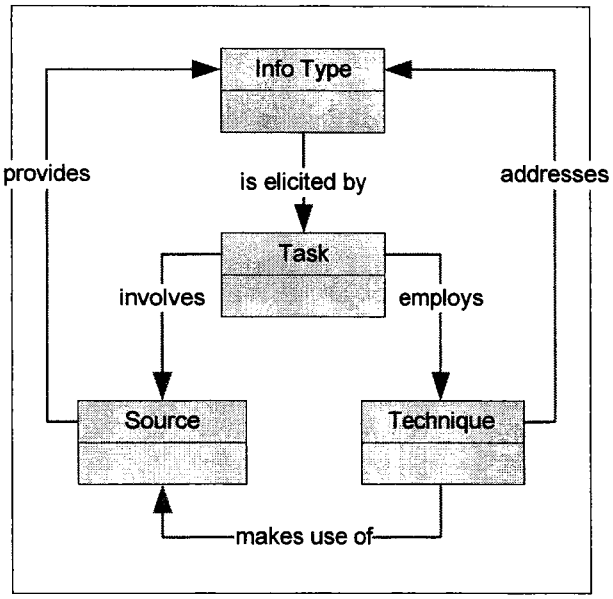

Figure 2: Method component meta-model

4.2 Process Model for the Approach

As previously stated, the approach presented in this paper represents both a process for engineering a situational method, as well as a process for performing requirements elicitation. The process model of the entire general approach as shown in Figure 3 (adapted from Brinkkemper, 1996), provides an overview and illustrates that the engineering of the method and the process of elicitation itself begin with characterization of the project at hand, construction of the method for this project, and then execution of this method within the project at hand. These individual 'steps' of the approach are subsequently detailed in the following subsections, as are the approach guidelines, which in addition to supporting all the steps of the approach, provide supplementary assistance for conducting requirements elicitation workshops, and in particular, provide possible solutions to many of the issues and challenges faced by practitioners during this difficult yet vital development activity.

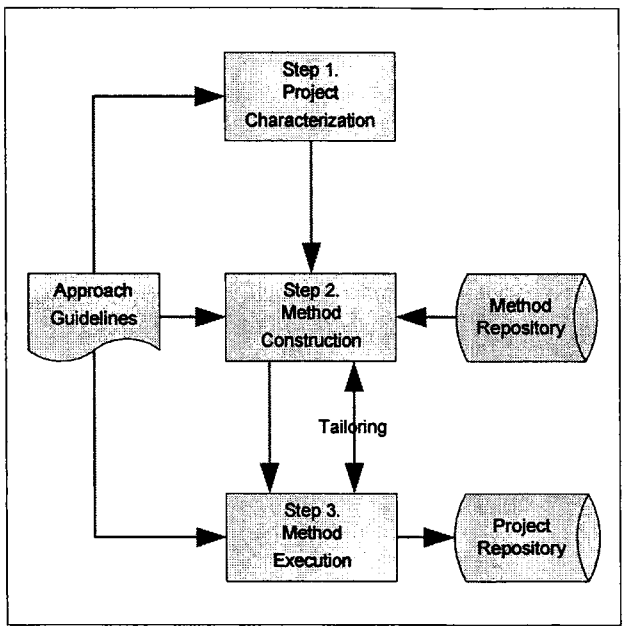

Figure 3: Approach process model

4.3 Step 1: Project Characterization

The first step of the approach is Project Characterization. It is at this point that the situational characteristics of the project at hand are identified in order to direct the construction of the method to be used. There are a number of different ways to characterize a software development project based on the specific situation. One suggestion has been to use taxonomy of specific problem, solution, and other project situational factors (Hickey and Davis, 2003). Another has been to base the characterization on the goals, risks, opportunities, and challenges of the project (Kurtz, 2001). Although these are useful ways of characterizing projects, this information is rarely available before the requirements elicitation phase, and in fact it is the actual purpose of requirements elicitation to discover and develop this information. Therefore, we use the following three basic attributes to categorize the specific requirements elicitation project:

1. **Definition** – The definition of the type of elicitation project being conducted. Examples of project definitions include Custom Development, COTS Selection, and Feasibility Study.
2. **Domain** – The general application domain of the envisaged system. Examples of application domains include Business Information, Group Support, and Embedded Control.
3. **Deliverable** – The required deliverable document from the elicitation project. Examples of project deliverables include Requirements Specification, Concept of Operations, and Vision & Scope documents.

Although admittedly quite basic, characterization of the project by using only these three variables (hereafter referred to as the '3Ds') does however take into account three of the most important and influential situational elements that are known at this very early stage of the software development lifecycle. The combination of values for these 3Ds of a given project is used to guide the construction of the method as described below.

4.4 Step 2: Method Construction

The second step of the approach is Method Construction. It is at this point that the method fragments are selected from the Method Repository and assembled into method components. These project components are then structured according to the process framework, and sequenced according to the approach guidelines. These operations are referred to collectively as the method construction, the result of which is an executable method than can then be enacted for the project at hand.

The Method Repository can be represented as a series of lookup tables, which detail instances of the different meta-types (called method fragments) and their relationships to other method fragments. There are ten tables in total, being one flat list for each of the meta-types (or classes), and one for each of the different possible inter-class relationships. In the Info Type and Task flat lists, flags are used to identify which ones are recommended for each of the possible values for the 3Ds identified in Step 1 of the approach. An analyst can therefore ensure that all the appropriate info types are addressed and all the prudent tasks are included in the requirements elicitation method, by using these tables to assemble a set of method components.

Once the method fragments have been selected and assembled to form a set of method components, this set of method components must then be structured and sequenced to complete the construction of the requirements elicitation method. In order to support this operation, the approach guidelines include a process framework with instructions that can be used by the analyst as a template for arranging and ordering the method components. This process framework, as can be seen in Figure 4, divides the complete set of method components into three key workshop phases of Scoping, High-level and Detailed. These phases are then divided into three stages being Preparation, Performance, and Presentation. Method components of the Performance stage of each phase are subjected to a third division into five different areas of interest.

It is anticipated that expert analysts would be able to select instances of each class from the method repository, assemble method components, and construct the resultant method with little or no reference to the approach guidelines and process framework. Furthermore, expert analysts often know familiar and proven ways of doing things during requirements elicitation, and therefore would much prefer the freedom to choose how the method fragments and method components are to be used. Novice analysts, on the other hand, would undoubtedly prefer a more prescribed approach to method construction given that by definition they lack the knowledge of experience and range of expertise that would be required to do this independently.

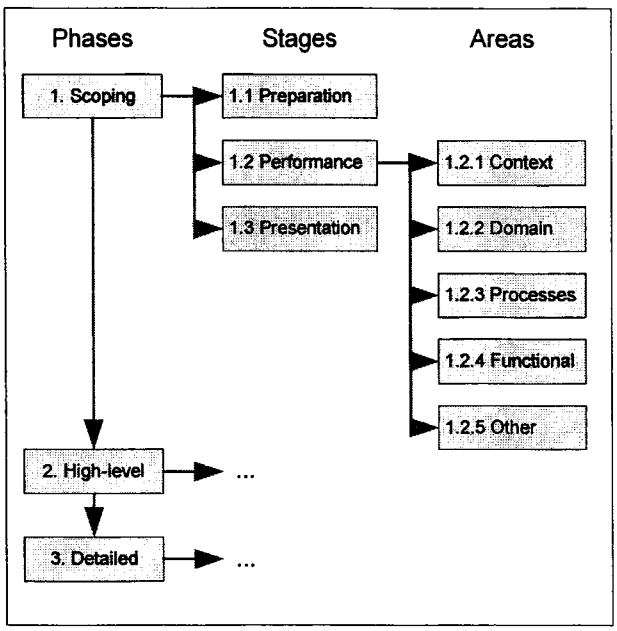

Figure 4: Process framework

In direct response to this, and in order to satisfy the key goal of this research to support novice analysts and projects without a defined requirements elicitation process, we have developed and incorporated into the approach guidelines what we have called 'ready-made' requirements elicitation methods. These represent pre-selected, pre-assembled, and pre-constructed methods for typical, common, and often occurring combinations of the 3Ds in practice. The ready-made methods currently available within the approach have been developed as a result of the review of the state of the art, and survey of the state of practice, in requirements elicitation previously mentioned. That is to say that the method repository was populated, and method components assembled, from direct references found in the literature to specific instances of the meta-types and their relationships. The ready-made methods were then constructed from these method components with reference to a number of example requirements documents from real world projects with the same situational characteristics according to our 3Ds classification. An overview of one of these 'ready-made' methods is demonstrated by the example in Section 5.

4.5 Step 3: Method Execution

The third and final step of the approach is Method Execution. Once the method has been constructed in accordance with the process framework using method components made up of method fragments from the method repository, the requirements elicitation part of the project can then be executed with the results of the method components being stored in the project repository. The Project Repository can be represented as a set of detailed templates for the different types of elicited and supporting information.

During execution, each method component task of the method is addressed utilizing the associated techniques to elicit the required info types from the available sources in support of or within a workshop environment. Each of the Performance

stages prescribed by the process framework are facilitated by the analyst, and may be completed over a number of workshop sessions depending on the complexity of the project, and the accessibility of relevant stakeholders. It is probable that the same info type may be addressed by more than one task at different stages of the method. In these cases the level of detail investigated and the attributes elicited for the info type are usually different. Normally each task has at least one available technique, and likewise each info type has at least one possible source. The exact steps necessary to conduct a specific task using a particular technique within a requirements elicitation workshop are also contained in the approach guidelines.

All method components can be repeated, removed, or reconfigured dynamically during the project by the analyst based on preferences or changing situations. This is referred to as 'tailoring' of the method. Tailoring can take place either before the method is enacted, as in the case of adding or removing method components after construction, or during the execution of the process itself, such as selecting which of the requirements elicitation techniques to use for a given task. Like those constructed from scratch, ready-made methods can be tailored by modifying the info types, tasks, techniques, and sources in the method repository before or after construction. This is important as it enables organizations and individuals to develop their own 'template' situational methods for different sets of project characteristics. It also creates a feedback and validation mechanism into the approach, adding a further level of flexibility and customization. An example of dynamic tailoring would be if the analyst believes insufficient information has been elicited for a particular info type during a session, a new task or technique for that info type can be selected from the repository, and added to the existing method, or utilized there and then.

5 An Example 'ready-made' Method

The following subsections provide an overview example of how one of the existing 'ready-made' methods would be implemented and enacted in a real-world project within the context of the general requirements elicitation approach presented in this paper. Although currently only a relatively small number of these 'ready-made' methods have been developed for different combinations of the 3Ds, this example aims to illustrate how the basic principles of the approach can be applied to most requirements elicitation project contexts. The example project has the following situation characteristics, i.e. values for the 3Ds, as defined in project initiation phase (preliminary planning) and identified by the participating analyst for the Project Characterization step:

1. Definition: Custom Development
2. Domain: Information System
3. Deliverable: Requirements Specification

Other project definitions currently supported by ready-made methods include COTS Selection, Call for Tenders, and Feasibility Study. Likewise, ready-made assistance exists for Group Support, Data Warehouse, and Embedded Control application domains. The production of project deliverables including Software Requirements Specifications, Concept of Operations Documents, Feasibility Reports, Tender

Documents, and Vision & Scope Statements are also available as ready-made methods for the mentioned definitions and domains. The 'ready-made' method for the 3Ds values identified above in the Project Characterization step has the following summary details:

- Number of Info types: 22
- Number of Tasks: 80
- Number of Sources: 18
- Number of Techniques: 14

Since each method component is based on an individual task, this means that there are a total of 80 core method components in the pre-selected, pre-assembled, and pre-constructed ready-made method for the production of a requirements specification for the custom development of an information system, according to the approach. These method components, within the context of the process framework, and with the assistance of the approach guidelines, provide the basis for the project execution step of the approach, and the conduct of the collaborative and combinational requirements elicitation workshops as described by phase and stage in the subsections below. Due to space limitations it is not possible to detail all the individual method components used in this ready-made method, or their descriptions and individual steps, however we have attempted to illustrate the general sense of the resultant workshops by presenting a concise summary and simple examples.

5.1 The Scoping Phase

The most important part of the Preparation stage of the Scoping phase (6 method components) involves the task of identifying the relevant stakeholder sources for participation in the Scoping workshop. Typically these sources would include the project sponsors, i.e. the people paying for the project, upper management of the same organization, and key members of the project team such as the project manager. Furthermore, any and all available external documentation sources relevant to the project should be reviewed such as the organizations marketing material and website. An example of a method component for this phase and stage can be seen in Table 2.

Table 2: Scoping Preparation Method Component – 'Gather preliminary info'

Class	Details
Task	Gather preliminary info
Info Type	All info types
Source	Project sponsors, Executive management, Project manager, External documentation
Technique	Informal discussions, Unofficial observations, Document analysis

During the Performance stage (11 method components), the participating stakeholder sources perform tasks to describe the problem, the mission, and the vision for the project, in addition to defining the boundaries for both the project and the system, i.e. what is in and out of scope. Furthermore high-level goals for both the project and the system are established, and key assumptions and constraints are identified. It is also at this point that the major risks, opportunities, and challenges are identified, and stakeholders for the High-level and Detailed workshops are determined. A number of techniques are prescribed for these tasks including Brainstorming, Questionnaires, and Goal Decomposition.

Presentation for the Scoping phase (6 method components) consists of documenting the results of the workshop sessions, checking these informally for quality, and then distributing the resultant Vision & Scope document (Wiegers, 2003) to the workshop participants for review and feedback either as a group as a walkthrough, or individually as an inspection.

5.2 The High-level Phase

In addition to reviewing all the available high-level internal documentation sources relevant to the project such as organization charts and departmental reports, the Preparation stage of the High-level phase (6 method components) also requires the analyst to observe and take notes at a high-level on the existing work processes and system operations relevant to the target system and the established scope. This enables the analyst to achieve a basic understanding of the business processes and therefore guide the subsequent workshop more intelligently that knowledge.

The Performance of the High-level workshop (15 method components) involves firstly the tasks of reviewing and refining the information elicited from the Scoping workshop. In addition to the project team, High-level workshops also typically include domain experts (or subject matter experts), middle management, and key user representatives as sources. During this stage, the system environment is examined in detail, and the main work processes, features, and capabilities of the target system are identified and described. This can be performed using a variety of techniques including Brainstorming, Questionnaires, Domain Analysis, Viewpoint Definition, Repertory Grids, Card Sorting, and Laddering. An example of a method component for this phase and stage can be seen in Table 3 below.

Table 3: High-level Execution Method Component – 'Determine system goals'

Class	Details
Task	Determine system goals
Info Type	Goals
Source	Project sponsors, Executive and middle management, Project manager and team, Domain experts, Key user representatives
Technique	Brainstorming, Questionnaire, Goal decomposition

Like the previous phase, the High-level Presentation stage (6 method components) consists of documenting the results of the workshop sessions, informal quality checking, and then distribution and review, except this time in the format of a Concept of

Operations (ConOps) document (IEEE, 1998a). Approval (review and update) of the High-Level document, as with the Scoping and Detailed documents, may required several iterations depending on the effectiveness of the workshops, commitment of the stakeholders, and the complexity of the project.

5.3 The Detailed Phase

A major task of the Preparation stage for the Detailed phase is not only to review detailed internal documentation sources such as work instructions and system manuals, but the analyst is also required to observe and take detailed notes on the existing work processes and system operations identified in the High-Level phase. As the participants for the Detailed workshops will typically include supervisors, end users, and developers, it is important that the analyst is very familiar with the specific tasks that must be supported by the target system.

During the Performance stage (18 method components), it is necessary for participating stakeholders to review the results from the Scoping workshop in order to understand the objectives and constraints of the current project and target system. Then the work processes, features, and capabilities identified in the High-level workshop are examined in detail with the relevant stakeholder and user sources. Each work process is decomposed into individual steps with exceptions and extensions using Use Cases or Scenario Analysis. Likewise each feature and capability is further decomposed into individual functional and non-functional requirements once again using a combination of Questionnaires, Goal Decomposition, and Viewpoint Definition requirements elicitation techniques.

The same process is followed once again for the Presentation stage of the Detailed phase (6 method components), however the format is that of a full System Requirements Specification document (IEEE, 1998). Given that this document is substantially more detailed than the previous documents, and involves many project stakeholders, approval can often take considerably more iterations and time. This is particularly the case when the document is to be used as part of a contractual agreement between a customer and supplier. An example of a method component for this phase and stage can be seen in Table 4 below.

Table 4: Detailed Presentation Method Component – 'Validate info quality'

Class	Details
Task	Validate info quality
Info Type	All info types
Source	All relevant stakeholders, All workshop participants
Technique	Inspection, Walkthrough

6 Discussion

It is important to remember when developing a situational approach for an activity like requirements elicitation that a number of delicate balancing acts naturally take place. One of these is between the flexibility and the rigor within the approach. Another is the risk of being too specific, and hence limiting the applicability of the approach to only a small number of situations, against the risk of being too abstract, and

therefore reducing the ability of the approach in providing the necessary support. Furthermore, requirements elicitation in particular is not a stand-alone process, but interrelated and interleaved with other development activities such as system design.

As a result we have endeavoured to provide the analyst with several ways of customizing the approach. This not only includes the ability to engineer situation methods for requirements elicitation from scratch, but also the capability to tailor these and ready-made methods throughout the process. These are supported by a process framework and detailed guidelines for each step of the approach, which the analyst can adopt completely, partially, or disregard altogether. Where possible we have attempted to conform to, or at least not conflict with, recognized ME practices, such as the research identified in Section 3, to guarantee a level of consistency. Although our focus has been on the early stages of requirements elicitation, we have been careful to ensure that the fundamental ME concepts used can be adapted and applied to other software development activities and range of process models that acknowledge the iterative and incremental nature of requirements elicitation.

Within the approach, the actual execution of the requirements elicitation project is mainly task driven, in order to optimize the process through structure and sequence. However, construction of the method is largely info type driven for the sake of completeness of the process. Rather than causing a conflict, these task driven and info type driven situations actually ensure that the overall approach is both effective (all the required info types are included and elicited) and efficient (tasks are performed productively and systematically). Although the tasks, info types, sources, and techniques of the workshops in the three recommended phases are significantly different, the entire activity is closely integrated and concentrated on the common objective of producing a complete, correct, consistent, and clear documented set of requirements for the target system with supporting information.

7 Conclusions

The lightweight approach for requirements elicitation presented in this paper provides a number of potential benefits over existing ones. It is both extensible and flexible in that it provides detailed guidelines for each step of the approach, and the ability to engineer and tailor situational methods based on specific project characteristics. Implementation of the approach does not require significant expertise or substantial experience, nor is it depended upon the utilization of any other systems development process. As a result the approach is particularly suited to novice analysts and those projects lacking a defined software development process, as it provides a high-level of guidance and instruction, and offers the necessary framework to ensure an efficient process, and effective results.

The execution of the approach takes advantage of both collaborative elicitation by being workshop centric, and the combination of multiple techniques in support of and integrated within the requirements elicitation workshop environment. As part of the approach we have introduced a number of useful and novel concepts including that of a 'method component' representing a task based method building block, and 'ready-made' methods that provide the analyst with a pre-constructed situational requirements elicitation process for the specific project at hand. Tailoring of the resultant methods can be performed throughout the process, even during performance of the requirements elicitation workshops themselves.

We believe that this research presents both researchers and practitioners of software development projects with an effective, efficient, useful, and useable requirements elicitation approach for this complex and critical activity. Furthermore, we are of the opinion that the approach described produces requirements elicitation methods that are profitable in terms of offering value for effort, therefore encouraging its acceptance and adoption into industry by organizations and analysts.

8 Future Work

The next step of this research, which is currently underway, is the development of a tool that embodies the complete approach, making it even more attractive, useful, and useable to analysts. The tool represents both a Computer Aided Method Engineering (CAME) and a Computer Aided Software Engineering (CASE) software based system, containing database driven method and project repositories. By utilizing web-based technologies, the tool and subsequently the approach can be used to support distributed requirements elicitation, where stakeholder groups are situated in different geographic locations across multiple time zones. Just as important, the approach and tool needs to be evaluated through industrial case studies in a variety of situations. Ideally this will be performed for a number of different application domains, in diverse project conditions, with analysts and stakeholders of varying levels of experience and expertise. It is expected that these evaluations will provide evidence as to the efficiency and effectiveness of the approach, and its applicability to various domains. The results can then be used to refine and expand the approach to better support a wider range of requirements elicitation contexts.

References

BALL LJ and ORMEROD TC (2000) Putting ethnography to work: the case for a cognitive ethnography of design. *International Journal of Human–Computer Studies* 53(1), pp. 147-168.

BEYER HR and HOLTZBLATT K (1995) Apprenticing with the customer. *Communications of the ACM* 38(5), pp. 45-52.

BRINKKEMPER S (1996) Method engineering: engineering of information systems development methods and tools. *Information and Software Technology* 38, pp. 275-280.

BRINKKEMPER S, SAEKI M and HARMSEN F (1998) Assembly Techniques for Method Engineering. *CAiSE'98*, LNCS 1314, pp. 381-400.

CHRISTEL MG and KANG KC (1992) *Issues in Requirements Elicitation.* Carnegie Mellon University Technical Report, CMU/SEI-92-TR-012.

DARDEENE A, VAN LAMSWEERDE A and FICKAS S (1993) Goal-Directed Requirements Acquisition. *Science of Computer Programming* 20(1-2), pp. 3-50.

DAVIS GB (1982) Strategies for information requirements determination. *IBM Systems Journal* 21(1), pp. 4-30.

GOGUEN JA and LINDE C (1993) Techniques for Requirements Elicitation. *International Symposium on Requirements Engineering*, pp. 152-164, January 4-6, San Diego, CA.

GOTTESDIENER E (2002) Requirements by Collaboration: Workshops for Defining Needs, Addison-Wesley, USA.

HENDERSON-SELLERS B (2002) Process Metamodelling and Process Construction: Examples Using the OPEN Process Framework (OPF). *Annals of Software Engineering* 14.

HICKEY AM and DAVIS AM (2002) The Role of Requirements Elicitation Techniques in Achieving Software Quality. *REFSQ'02*, September 9-10, Essen, Germany.

HICKEY AM and DAVIS AM (2003) Requirements Elicitation and Elicitation Technique Selection: A Model for Two Knowledge-Intensive Software Development Processes. *Hawaii International Conference on System Sciences*, January 6-9, Big Island, Hawaii.

HUDLICKA E (1996) Requirements Elicitation with Indirect Knowledge Elicitation Techniques: Comparison of Three Methods. *International Conference on Requirements Engineering*, pp. 4-11, April 15-18, Colorado Springs, CO.

IEEE (1998) *Std 830 Recommended Practice for Software Requirements Specifications.*

IEEE (1998a) *Std 1362 System Definition - Concept of Operations (ConOps) Document.*

KOTONYA G and SOMMERVILLE I (1998) *Requirements Engineering: Processes and Techniques.* John Wiley & Sons, Great Britain.

KURTZ T (2001) Ask Pete, Software Planning and Estimation through Project Characterization. *International Symposium on Requirements Engineering*, pp. 286, August 27-31, Toronto, Canada.

MAIDEN NAM and RUGG G (1994) Knowledge Acquisition Techniques for Requirements Engineering. *Requirements Elicitation for Systems Specification*, July 12-14, Keele, UK.

MAIDEN NAM and RUGG G (1996) ACRE: selecting methods for requirements acquisition. *Software Engineering Journal* 11(3), pp. 183-192.

MAIDEN N, MANNING S, ROBERTSON S and GREENWOOD J (2004) Integrating Creativity Workshops into Structured Requirements Processes. *Designing Interactive Systems*, August 1-4, Cambridge, MA.

OPF (2005) *OPEN Process Framework.* http:// www.opfro.org [Accessed 22 July 2005].

POTTS C, TAKAHASHI K and ANTON A (1994) Inquiry-Based Requirements Analysis. *IEEE Software* 11(2), pp. 21-32.

RALYTE J and ROLLAND C (2001) An Assembly Process Model for Method Engineering. *CAiSE 2001*, LNCS 2068, pp. 267-283.

RALYTE J, DENECKERE R and ROLLAND C (2003) Towards a Generic Model for Situational Method Engineering. *CAiSE 2003*, LNCS 2681, pp. 95-110.

SOMMERVILLE I and SAWYER P (1997) *Requirements Engineering: A Good Practice Guide.* John Wiley & Sons, Great Britain.

SOMMERVILLE I, SAWYER P and VILLER S (1998) Viewpoints for Requirements Elicitation: A Practical Approach. *International Conference on Requirements Engineering*, pp. 74-81, April 6-10, Colorado Springs, CO.

SUTCLIFFE A and MAIDEN N (1998) The Domain Theory for Requirements Engineering. *IEEE Transactions on Software Engineering* 24(3), pp. 174-196.

WIEGERS KE (2003) *Software Requirements.* Microsoft Press: USA.

ZOWGHI D and COULIN C (2005) Requirements Elicitation: A Survey of Techniques, Approaches, and Tools. In *Engineering and Managing Software Requirements* (AURUM A and WOHLIN C, Eds), Springer: US.

Situational Requirements Engineering Processes

Methods, Techniques and Tools to Support Situation-Specific Requirements Engineering Processes

SREP'05

The 1st International Workshop

Paris, France, 29–30 August 2005

Edited By Jolita Ralyté, Pär J. Ågerfalk and Naoufel Kraiem

Organized By IFIP WG8.1 Method Engineering Task Group
in Conjunction with
13th IEEE International Requirements Engineering Conference

Proceedings of the First International Workshop on Situational Requirements Engineering Processes: Methods, Techniques and Tools to Support Situation-Specific Requirements Engineering Processes (SREP'05), Paris, France, August 2005, In Conjunction with the Thirteenth IEEE Requirements Engineering Conference (RE'05).

Edited by:

Jolita Ralyté (jolita.ralyte@cui.unige.ch)
University of Geneva, Switzerland

Pär J Ågerfalk (par.agerfalk@ul.ie)
University of Limerick, Ireland

Naoufel Kraiem (naoufel.kraiem@ensi.rnu.tn)
University of Manouba, Tunisia

Printed in Ireland by the University of Limerick
ISBN: 1-874653-82-8