Published in IET Software
Received on 10th August 2010
Revised on 2nd May 2011
doi: 10.1049/iet-sen.2010.0096

ISSN 1751-8806

Using soft systems methodology to improve requirements practices: an exploratory case study

N. Niu[1] *A.Y. Lopez*[1] *J.-R.C. Cheng*[2]
[1]Department of Computer Science and Engineering, Mississippi State University, Box 9637, MS 39762, USA
[2]Information Technology Laboratory, US Army Engineer Research and Development Center, Vicksburg, MS 39180, USA
E-mail: niu@cse.msstate.edu

Abstract: Soft systems methodology (SSM) should offer substantial benefits in managing expectations and requirements for a software-intensive system, but the benefits have not yet been examined empirically. This study reports an exploratory case study investigating the hypothesis that 'soft systems approach would identify all the flaws in requirements practices and suggest improvements suited to an organisation's context'. The authors analysed problematic requirements practices in an ongoing software-intensive socio-technical project, modelled potential changes and asked the project team to assess the organisational fit of these changes. The authors further monitored the requirements engineering improvements that the project team made according to the case study. The authors conclude that SSM could indeed uncover a relatively complete set of flaws in requirements practices. Although not all suggested changes were regarded as necessary, the implemented changes had contributed positively to the organisation's requirements engineering improvements.

1 Introduction

It is argued in the software requirements engineering (RE) literature that completely formalising the requirements is impossible because they cannot be fully separated from the stakeholders' intentional, organisational and social context [1]. RE, therefore, is a 'soft' human-centred activity concerned with identifying and communicating the purpose of a software-intensive system, and the context in which it will be used. Key activities in RE include plan and elicit requirements, model and analyse requirements, communicate and agree requirements, and realise and evolve requirements [2]. This activity list is by no means exhaustive, for example, validating requirements is not explicitly mentioned and communicating requirements can be refined to negotiating and prioritising requirements. In practice, an organisation selects a set of RE activities that are critical to its business success.

Soft systems methodology (SSM) [3] aims to deal with 'fuzzy' problem situations like RE, where multiple stakeholders' diverse objectives exist [4]. In this context, the requirements engineers must cooperate with the users in understanding the problem situation. However, users have yet to be fully empowered in deciding the RE processes and the actual requirements [5]. Moreover, software engineers are seen to deliver what is thought of as the users' requirements with minimal reference to the users [6]. As a result, a large number of software projects failed. For instance, a survey of the empirical literature [7] estimated system failure was between 25 and 90% because of the following generic issues:

- Correspondence failure – the delivered system does not correspond to what was required.
- Process failure – a system is not forthcoming within time or resource constraints.
- Interaction failure – systems, as implemented, which fail to satisfy the users.
- Expectation failure – systems that are unable to meet stakeholders' expectations or values.

Although the literature survey [7] was conducted a while ago, the above failure types are still relevant today. A recent study finds these four failure types are broad enough to represent a wide range of critical factors that impact the delivery of information systems projects [8]. Another example comes from a domain-specific literature review [9], which uses the above generic types to characterise the failure of e-government projects. These generic failure types are also mapped well to the RE literature. A study of 56 projects within 29 software organisations worldwide confirmed that the goal of RE process improvement was to address the failure types listed above [10, p. 543]. In a study of 12 software companies, Hall *et al.* [11] found that a majority of the requirements problems were organisational issues, characterised by the correspondence, process, interaction and expectation problems. An industrial case study pointed out four RE challenges [12, p. 51] that could result in the four types of failures, if not addressed properly. Thus, we use the generic failures types [7] as a baseline when discussing RE flaws.

RE is well recognised in industry to be critical to the success of any major development project. Several authoritative studies have shown that RE defects costs 10–200 times more to correct once fielded than they would if they were detected during requirements development

[13, 14]. RE addresses the defects more economically early on in the software life cycle only if the problems that lead to system failures can be identified. In this context, SSM is effective if it provides improvements to all the problem issues, which we refer as a key premise of applying SSM in RE. In theory, the premise is supported by the holistic thinking and the systematic modelling of SSM [3, 5, 6, 15]. However, we are aware of no empirical studies that investigate this basic tenet, nor the scope of its applicability.

To shorten this gap, we conducted an evaluation of applying SSM to an organisation's RE practices. We set out to test the underlying hypothesis that SSM would find all the RE flaws and suggest improvements suitable for the organisation's context. Our goal was to explore what differences the use of SSM made to RE. We also wanted to examine the mechanisms (how) and rationales (why) behind these differences. To that end, we used SSM to model a small-sized company's requirements practice for carrying out a socio-technical project, identified practice areas that needed improvement, interviewed people with different roles in the company in order to assess the organisational fit of potential changes and monitored some of the RE changes that the company implemented based on our analysis. The purpose of our study is to allow the software organisation to assess and extend its RE practice via SSM.

The remainder of the paper is organised as follows. Section 2 lays the background of our work and compares SSM with other RE process improvement approaches. Section 3 presents our study's context. Section 4 details the case study design. Section 5 discusses our main findings. Section 6 addresses threats to validity. Section 7 draws some concluding remarks and outlines future work.

2 Background and related work

2.1 RE and SSM

The RE process is inherently 'soft' since the needs that drive requirements are embedded in the social, cultural and organisational contexts [1]. It has also been argued that the validation of requirements must remain a social process, in which the users are to be convinced of the system value [2]. Checkland's SSM [3] explicitly involves users in system development. His representations deliberately avoid analytic methods, favouring human views of the problem [5]. SSM as a methodology lends itself particularly well for dealing with situations where there exist many different perspectives, values and beliefs around what aspects of the situation are most important and how to address them.

SSM has been used to study systems characterised by the activities where the human behaviour is a key factor that determines the result (satisfactory or not) of these activities, such as requirements elicitation and validation. SSM was originally seen as a modelling tool; however, in recent years, it evolved to an organised learning tool [4]. SSM views models as a means of thinking about reality rather than models of reality [6].

At the heart of soft systems thinking is the principle that whole entities exhibit emergent properties which are meaningful only when attributed to the whole, not to its parts [16]. In this sense, SSM utilises holistic thinking rather than breaking the whole into smaller, more manageable fragments (reductionism). Such a holistic view is particularly useful for understanding stakeholders' perceptions of the problem situation, examining 'regularly patterned' activities [17] such as eliciting and communicating requirements, uncovering flaws in current RE practices and enabling resolutions in an organised manner.

Many connections exist between SSM and RE. SSM is commonly referred to as a requirements elicitation technique because of its established role in knowledge acquisition [18]. Using SSM as a theoretical basis for RE is challenged in [17] since SSM seems to deny technical aspects of software's intelligibility. However, it is important to note that SSM is not a replacement for, but a complement to, 'hard' methodologies such as the structured systems analysis and design method (SSADM). In fact, SSM is often used as a front end to SSADM to understand soft problem situations and make incremental organisational changes [15].

SSM involves users in software design [5], and Mathiassen *et al.* [16] demonstrate the usefulness of applying SSM in the technical domain of designing software applications. In contrast, we explore the possibility of applying SSM in the soft domain of RE that identifies and communicates the purpose and the context of a software-intensive system.

2.2 Related work

Software process improvement (SPI) emerged during the late 1980s and its influence on the industry's practice has been growing ever since. It is believed that SPI generally delivers substantial benefits. Research results show that companies with high level of maturity benefit from increased product quality, improved customer satisfaction and reduced risk [19]. SPI models and standards, such as capability maturity model (CMM) [20] and ISO/IEC 15504 international standard [10], prescribe processes for software organisations to assess their capabilities and achieve high-quality software. An important area of SPI is RE-related process improvement. As argued by Sawyer *et al.* [21], no software process, whatever its 'capability', can keep delivery times, costs and product quality under control if the requirements are poorly formulated or unstable. RE-related SPI incentives, therefore, aim to capitalise the benefit of identifying and resolving errors during the requirements process, rather than later on [22].

Similar to the benchmarking paradigm in the overall SPI, several RE-related capability models emerge. These models involve codifying what are believed to be good RE practices, typically including an extensive catalogue of processes and practices organised in a recommended order of implementation [10]. For example, Sommerville and Sawyer [23] defined a framework for 66 requirements practices derived from existing SPI models and reports of practical experience. As exhaustively evaluating an organisation's process against such a comprehensive checklist can be time consuming and ineffective, lightweight and pragmatic RE SPI frameworks were proposed. In [24], a nine-step document/template-driven method was developed. In [25], a framework with four abstract RE practices was presented.

Although the codified RE practices have much value, their derivation depends mostly on anecdotal evidence and *ad hoc* reviews of general SPI models. Two classes of empirical studies have been conducted to systematically develop and validate software process assessment models: correlational studies and case studies [26]. With correlational studies, one collects data from a number of organisations or projects and investigates relationships about process capability. Quantitative product and project metrics (e.g. defect density, cost and schedule performance indices etc.) are a prerequisite for correlational studies. El Emam and Birk presented one of the most comprehensive correlational studies, where they evaluated the predictive validity of requirements analysis capability for 56 projects within 29 organisations over 2 years [10]. The study confirmed a strong relationship between the implementation of requirements analysis practices and the productivity of software projects. For organisations with less than or equal to 50 IT staff, evidence of predictive validity was rather weak [10].

Although correlational studies use quantitative data to draw statistical generalisations, case studies aim to obtain theoretical generalisations by thoroughly investigating the experiences of a single organisation or a small number of selected organisations [27]. Case studies are particularly useful when objective process and product data are not available. In these situations, qualitative data collection and analysis techniques can be used. Damian *et al.* [12] conducted a case study at the Australian Center for Unisys Software (ACUS). Following a CMM mini-assessment, ACUS formed a focused initiative to improve its RE practices with a view to learn from mistakes made on previous projects. The upper management at ACUS was highly committed to SPI and undertook significant changes in the RE process, including the introduction of group session approaches to requirements analysis and a structured method for writing requirements [12]. Based primarily on questionnaires and interviews, the case study confirmed the benefits resulted from the RE-related SPI activities at ACUS.

A case study of the problems experienced by 12 software companies in their requirements process was reported in [11]. Specifically, 45 focus groups (each with four to six people) were conducted to develop a more holistic understanding of the requirements process. Such a holistic understanding is different from SSM's holistic thinking used in our work. In [11], the authors generalised a holistic RE practice model from 12 organisations, namely the R-CMM (Requirements Capability Maturity Model) [28]. In our work, SSM's holistic thinking is manifested in explicitly considering multiple stakeholders of one single organisation and comprehensively modelling fine-grained requirements processes. Hall *et al.*'s [11] study, in contrast, intentionally ignored customer and user roles when structuring focus groups.

We believe that SSM complements all previous empirical studies on RE SPI in that the holistic thinking can be directly applied into one project within a single organisation. This not only helps to tease out project-specific requirements process, but also facilitates an organisation-specific assessment of the potential changes. In this way, the process assessment based on SSM is truly conducted in situ, and more importantly, before applying any changes to the organisation's RE process, the staff perception of potential changes is solicited and evaluated. This is advantageous to the previous investigations, in which the activities to improve the company's RE process are either prescribed based on SPI standards (e.g. [10, 21, 24, 25]) or predetermined by highly motivated upper management (e.g. [12, 29]).

3 Study context

The subject in our study is a research organisation that specialises in scientific computing and government service. It is located in Starkville, Mississippi, USA and has approximately 50 employees as of 2009. To honour confidentiality agreements, we will call it SrvU. SrvU's mission is to plan and develop software to assist scientific communities and government agencies.

Fig. 1 shows SrvU's simple organisational structure where the head and the project coordinator manage the projects. The research scientists gather and analyse the information of the projects. The media laboratory coordinator manages the technological infrastructure, and in conjunction with the database administrator (DBA), acts as an intermediary between the clients and SrvU's project team. In addition, the DBA is responsible for data management. The programmers develop the software system required for the project.

The case (unit of analysis) in our study is a traffic and transportation project, whose goal is to provide software support for analysing and reporting daily activities the customer collects. Key project features include information visualisation and predictive road traffic modelling. Table 1 lists five sample requirements of the project. Note that the identifier (ID) of each requirement in Table 1 is artificially devised for the discussion of this paper only, and does not reflect the actual identification. Note also that we have selected examples that are easier to understand than average. The project under study experienced failures in the past, mainly because the delivered software-intensive system had not satisfied the user requirements. For this reason, the project has been redone several times and is currently within a new development cycle.

SrvU is aware of SPI models like CMM, but no formal process assessment initiatives to date have ensued. SrvU has not systematically collected historical product and/or process data (e.g. on defect levels, or even keep accurate effort records). These ruled out the possibility of any correlational study [26] of SrvU. We selected SrvU's traffic and transportation project as an ideal case study [27] to explore the use of SSM in RE for a number of reasons. First, the project is a representative socio-technical system that involves multiple stakeholders with diverse objectives. This not only fits SSM's scope of applicability, but also indicates that the lessons learned from our case are informative about the experiences of the typical situation. Second, SrvU's development of this project represents both a revelatory and a longitudinal case, in that several iterations were taken for SrvU to better meet the user requirements. This is critical to SSM because one cannot really be certain about the SSM findings (i.e. flaws in SrvU's RE practices and improvements) except perhaps retrospectively. This also allows the project team to implement some of the changes that we suggest so that the effects of the changes can be monitored. Finally, SrvU's

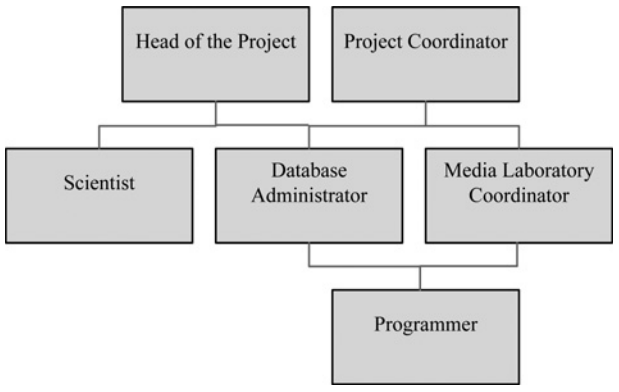

Fig. 1 *Organisational chart*

Table 1 Sample requirements

ID	Description
R.1	the user shall manage (add, update and delete) any daily activity data such as vehicle mileage, accidents, violations and ticket number per day
R.2	the user needs a unique ID to manage each personnel in the traffic control department
R.3	the user wants a visualisation tool to show the trend of information of the traffic data by category (accidents, injuries etc.), by personnel (using a unique ID), by time, and by district and region
R.4	the user requires the software system to generate a file with the information specified in R.3 so as to compare and filter data according to different criteria
R.5	the user requires a PDF file for the information requested in R.3 and R.4 for printing and reporting purposes

project members were highly motivated to participate in our study in order to make organisational-wide changes that can help to generate the right product required by the users. Therefore we anticipated a high degree of access to key stakeholders and project's data.

4 Methodology

Case studies [27] are an important empirical method, suitable for investigating 'how' and 'why' questions that cannot be addressed through controlled experiments. Essentially, the benefits of the use of SSM in RE are only likely to be evident for ongoing socio-technical software projects, under conditions that cannot be replicated in the laboratory. In particular, the study of applying SSM in RE cannot be separated from the organisational context, and the effects may take weeks or months to appear.

We use an exploratory case study [27] as the basis for our research design. Exploratory case studies are appropriate for preliminary studies in which it is not yet clear which phenomena are important, or how to measure these phenomena. In our case, we were particularly interested in understanding how the use of SSM would affect the RE process. Although SSM promises RE improvements in theory, the benefits have not yet been observed empirically. We know little about how exactly SSM is best deployed, nor how the expected benefits arise. For these reasons, it would be premature to try to measure the cost/benefit trade-off. For this study, our intention was to explore how SSM affects RE practices.

4.1 Study design

We derived a central hypothesis to guide our study design: 'SSM would find all the RE flaws and suggest improvements suitable for SrvU's context.' To investigate this, we adapted Checkland's conventional model [3], which allows the systems practitioner to build up the 'richest possible' (i.e. multiple perspectives) picture of the 'problem situation' in the real-world setting. Only then is the practitioner expected to formulate his or her conceptual models.

We adapted the conventional SSM structure to a more succinct model, as suggested in [16, 30]. Fig. 2 shows the adapted SSM stages. Stage 1 examined SrvU's current RE practices. Stage 2 formulated root definitions via the mnemonic CATWOE, one of the best-known SSM tools, used to define the customers, actors, transformation, weltanschauung (the worldview), ownership and environmental constraints [3]. Stage 3 builts the conceptual model and identified potential organisational changes. Stage 4 allowed the stakeholders to debate the consequences of the proposed changes, so that the culturally feasible improvements to SrvU's RE practices could be selected. Stage 5 applied the changes to improve SrvU's requirements practices.

We tested our central hypothesis primarily via two evaluations, both of which were intentionally carried out in the 'real world' (see Fig. 2):

- Objective assessment: We used the four generic failure types [7] discussed in Section 1 (correspondence, process, interaction and expectation) as a baseline to check whether SSM's holistic thinking would identify a relatively complete set of RE flaws. This was done at stage 1 of Fig. 2.
- Subjective assessment: We used questionnaires and semi-structured interviews to check whether SSM's conceptual modelling would generate RE improvements suitable for SrvU. This was done at stage 4 of Fig. 2.

In addition, we monitored the effects of the change (stage 5 of Fig. 2) by interviewing SrvU's project members and by observing their requirements practices. As the project team is incrementally implementing the changes during the current software development cycle, our evaluation for stage 5 relies on the observations we made in situ and the feedbacks we received from the project members.

4.2 Data collection and analysis

In our study, the data were collected by means of field study, artefact analysis, interviews and questionnaires, and e-mails. We paid a few on-site visits for ethnographic observation. We exchanged e-mails with the project members to collect feedbacks. Artefacts in our analysis included project management report, functional requirements, meeting minutes etc. The interviews and questionnaires involved project coordinator, DBA, and programmer, spanning SrvU's organisational hierarchy shown in Fig. 1. All

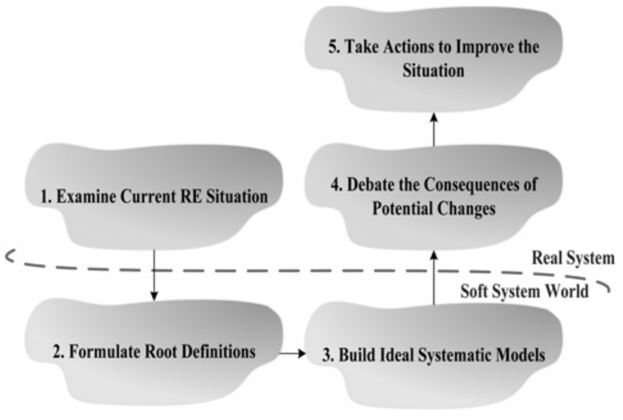
5. Take Actions to Improve the Situation
1. Examine Current RE Situation
4. Debate the Consequences of Potential Changes
Real System
Soft System World
2. Formulate Root Definitions
3. Build Ideal Systematic Models

Fig. 2 *Adapted SSM stages*

interviews were recorded and transcribed with the consent of the interviewees. Complete anonymity was assured in interview transcripts and questionnaires. Note that we took the actual project duration into account when collecting data because some experiences and answers were only obvious in hindsight.

In our evaluation, we used qualitative methods [31] to analyse the collected data. Qualitative research seeks to make sense of the way themes and meanings emerged and patterned in the data records built up from observations, interviews and questionnaires. It is particularly suitable in our study since our collected data consisted of records of observation and interaction that were complex and contextualised.

5 Results

5.1 Understanding RE practices

A unique feature of the project under study is that an intermediary representative, instead of the group of actual users, participated in requirements meetings with SrvU's project team. SrvU's DBA occasionally talked to end users by phone and devised requirements for them. The project team primarily used weekly meetings to elicit requirements, supplemented by DBA's phone interviews. Every team member was mandated to attend the weekly meeting, coordinated by upper level personnel like the head of the project. Requirements validation was carried out by actual users' visits to SrvU at various time intervals, resulting in approval or rejection of the software system developed.

We described SrvU's RE situation in terms of a series of activities and their relations to external entities, as shown in Fig. 3. One of the authors of this paper built this model that reflected the requirements practices in the previous project cycle that ended roughly in April 2010. The model was validated through a walk-through meeting with SrvU's project members. Three activity groups, along with some flaws, emerged from our analysis:

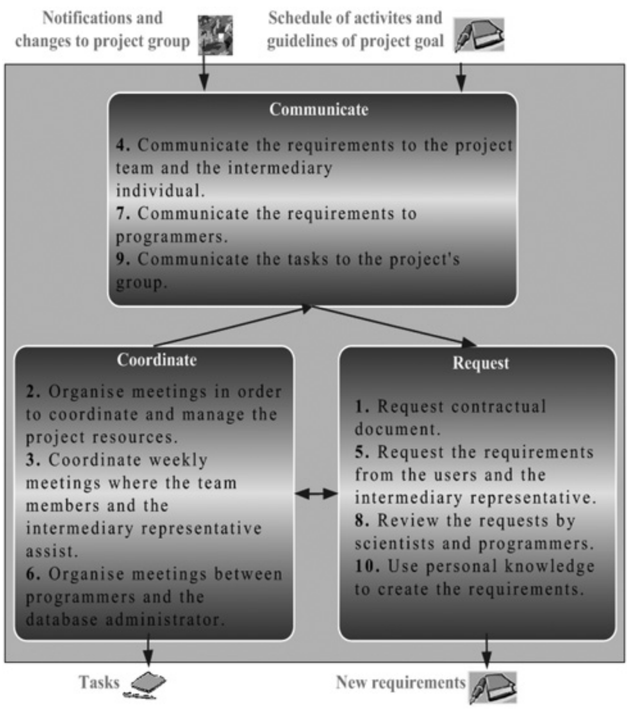
Notifications and changes to project group
Schedule of activites and guidelines of project goal
Communicate
4. Communicate the requirements to the project team and the intermediary individual.
7. Communicate the requirements to programmers.
9. Communicate the tasks to the project's group.
Coordinate
2. Organise meetings in order to coordinate and manage the project resources.
3. Coordinate weekly meetings where the team members and the intermediary representative assist.
6. Organise meetings between programmers and the database administrator.
Request
1. Request contractual document.
5. Request the requirements from the users and the intermediary representative.
8. Review the requests by scientists and programmers.
10. Use personal knowledge to create the requirements.
Tasks
New requirements

Fig. 3 *Description of current situation*

- Communicate: As discussed earlier, an intermediary user representative, rather than the end users, participated in the project's weekly meetings. The lack of communication with the end users posed the risk that the delivered system was unable to meet their expectations (expectation failure) and did not correspond to what was actually required (correspondence failure). For example, the DBA asked the intermediary representative whose ID with respect to R.2 (cf. Table 1) should be used in the project. The representative suggested the batch number, but did not realise that the batch number was not unique for the personnel. This requirements-level mistake cost great re-work in database design and software implementation.
- Request: The fact that SrvU's DBA had to coin, fabricate and even guess the users' requirements would cause correspondence and interaction failures [7]. For example, the DBA assumed that personnel management (R.2 in Table 1) would be done in a DB-batch mode. However, the user/administrator wanted a web form (application code) to add, update and delete personnel information (first name, last name, status, rank, district etc.). This type of missing requirements was not unusual throughout the project.
- Coordinate: The mandatory meetings organised by upper management could result in software engineers' passive participation in the project, thereby causing a loss of motivations to some extent [32]. The lack of requirements analyst's skill set, such as interpersonal and coordinate, could contribute to the process failure [7]. For example, the sample requirements R.1, R.3 and R.4 shown in Table 1 were all concerned with displaying and manipulating traffic data. A specialised requirements analyst might be able to further distinguish stakeholder goals (visualising, predicting, reporting etc.) and allocate the resource more properly to fulfil them.

As a result, the holistic thinking, in which different areas of requirements practices were taken into consideration as a whole, indeed uncovered a complete set of RE flaws, according to the baseline failure types (correspondence, process, interaction and expectation) [7] employed in our study. These RE flaws were also confirmed through our field observations and interviews with SrvU's employees.

5.2 Conceptual modelling in soft systems world

We leveraged the CATWOE mnemonic [3] to formulate the SSM's root definition, a precise description of the emergent properties of a system [16]. The CATWOE elements ensure the root definition is relevant, valid and complete [15]. The results of our CATWOE analysis for SrvU's human-centred RE activities were:

- Customers (C): users of the software project.
- Actors (A): SrvU's project members.
- Transformation (T): producing a software system fulfilling the user's needs.
- Weltanschauung (W): project members whose goals are to elicit, analyse and validate requirements so that the high-quality software can be constructed and delivered.
- Ownership (O): SrvU.
- Environmental constraints (E): restrictions and standards applied to the use of information and technology framework used.

Teasing out CATWOE definitions provided a firm conceptual basis for modelling potential changes that SrvU might apply to improve its RE practices. Fig. 4 shows the suggested changes, given by the new activity series. We grouped the changes into four areas:

- Communication: A more direct communication with the users should adjust the expectations more properly, and greatly reduce the DBA's guessing of what the users really want.
- Elicitation: The requirements should be gathered using multiple elicitation techniques [5], and reviewed by all stakeholders including users, DBA and programmers.
- Role: An explicit 'requirements analyst' role should be created to better coordinate project meetings and to possibly replace the intermediary user representative.
- Documentation: Both SrvU and customers should agree on and maintain a formal document for contractual purposes.

The above areas for improvements were modelled by one of the authors of this paper. It is our belief that these changes stemmed from the soft systems world would overcome SrvU's RE flaws in a culturally feasible way. Fig. 5 shows how the RE areas could address different types of software failures [7].

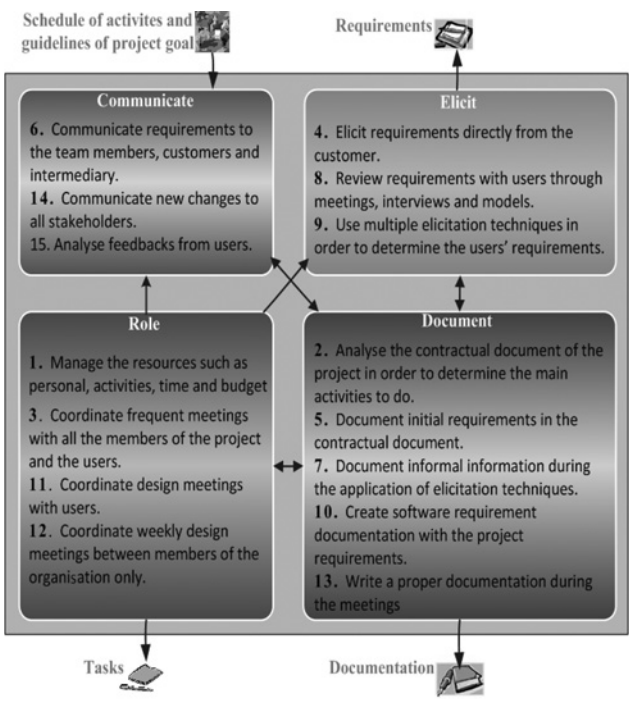
Schedule of activites and guidelines of project goal
Requirements
Communicate
6. Communicate requirements to the team members, customers and intermediary.
14. Communicate new changes to all stakeholders.
15. Analyse feedbacks from users.
Elicit
4. Elicit requirements directly from the customer.
8. Review requirements with users through meetings, interviews and models.
9. Use multiple elicitation techniques in order to determine the users' requirements.
Role
1. Manage the resources such as personal, activities, time and budget
3. Coordinate frequent meetings with all the members of the project and the users.
11. Coordinate design meetings with users.
12. Coordinate weekly design meetings between members of the organisation only.
Document
2. Analyse the contractual document of the project in order to determine the main activities to do.
5. Document initial requirements in the contractual document.
7. Document informal information during the application of elicitation techniques.
10. Create software requirement documentation with the project requirements.
13. Write a proper documentation during the meetings
Tasks
Documentation

Fig. 4 *Potential changes to improve the situation*

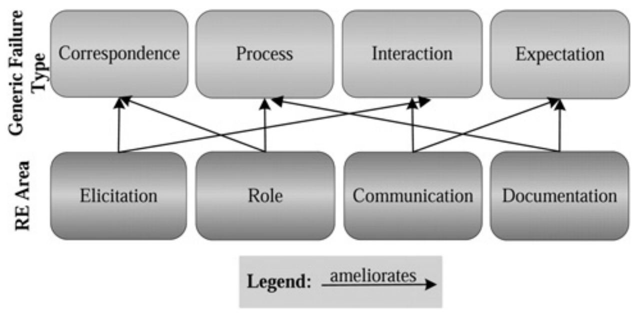

Fig. 5 *Addressing flaws*

5.3 Assessing changes

The suggested changes must be critically evaluated before SrvU's adoption. We designed a questionnaire to guide our semi-structured interviews with SrvU's project coordinator, DBA and programmer. Each of the nine questions covered one area of the RE improvements, as shown in Table 2. The three interviews were conducted separately, ensuring non-interference with other team members' opinions. Each interview lasted approximately 20 min. Note that we conducted the interviews in April 2010 when the previous project cycle came to an end. This enabled SrvU's project members to take into account their experiences when assessing our proposed changes.

We analysed interview transcripts and completed questionnaires by two qualitative data analysis methods [31]: coding (relating answer sections to proper subject matters under testing) and categorising (classifying answers to be positive or negative). Table 2 summarises the results. Note that SrvU's project members' quotes are represented italic and cited in double quotation marks (' ').

SrvU's project team found changes in three areas to be acceptable. In particular, '*the communication with the users should be as needed just to make sure they are in the loop if we have questions*'. Although common elicitation techniques were meetings and personal knowledge, the project team realised the importance to incorporate other techniques like rapid prototyping and knowledge acquisition. Proper documentation was crucial not only for contractual purposes, but also for '*keeping everyone on the same page*'. In this regard, flexible formats like feature lists were preferred.

To our surprise, the idea of setting up the 'requirements analyst' role was not appealing to SrvU. According to some project members, SrvU enjoyed the culture of small project teams, supported by their flat organisational structure. They regarded the current division of labour, especially with the intermediary user representative, was effective. They were concerned that adding another role would increase latency and reduce the throughput of SrvU's business processes.

The interviews revealed SrvU's 'few roles' philosophy. Our interviewees pointed out that people on a project must communicate with each other in order for the project to make progress. Yet the overhead of this communication can hinder the very progress it should facilitate. Indeed, the number of possible communication paths among roles increases quadratically with respect to the number of roles [33]. For example, 5 roles have 10 communication paths, but 10 roles have 45 paths. Moreover, 20 roles have 190 possible communication paths. It is clearly not possible for every role to communicate with every other role. Therefore information often reaches roles indirectly through other roles. However, this process increases both delay and overhead. The combinatorics of communication encourages few roles. 'Few roles' means that communication becomes more efficient, both in resources consumed and in speed. This is the main reason that SrvU is hesitant to create a new 'requirements analyst' role.

5.4 Monitoring improvements

Our work has provided SrvU much insight into their RE practices. Although we cannot claim that our results have had direct impacts on all SrvU's RE processes, we can claim to have initiated changes in some areas. Already, SrvU has introduced rapid prototyping to the project's

Table 2 Summary of subjective evaluation

RE area	Question	Answer	Result
communication	how often do you communicate with the user/customer?	all responded 'more than twice a week'	somewhat positive
	what do you think would be the ideal time interval to communicate with the user/customer?	one emphasised 'twice a week' and others commented 'as needed'	
elicitation	what kinds of elicitation techniques have been applied to the project?	primarily 'meetings' and 'personal knowledge and experience'	positive
	what elicitation techniques would help you to identify requirements more effectively? Why?	preferred multiple techniques (meetings, modelling, prototyping etc.)	
role	is there a requirement analyst, or a similar role, in the project? If yes, how many?	'no' and such role does not seem to be necessary	negative
	how do you categorise the role of the intermediary user representative in the current project?	two responded 'very effective' and one responded 'somewhat effective'	
documentation	are the requirements documented for the project?	'(yes,) under contract' and '(yes,) in the initial (project) proposal'	positive, but informal documentation is preferred
	based on your experience, do you find requirements documentation useful? Why?	all responded 'yes' and one commented '(the documents) help keep me focused on what exactly I need to accomplish'	
	in your opinion, what is the best way to document the requirements for the project?	preferred lightweight formats, e.g. feature list, e-mails, diagrams etc.	

current iteration, and also started using feature list to document user requirements.

The project team, under our guidance, has experimented with a couple of knowledge acquisition techniques (card sorting [34] and repertory grids [35]) during new rounds of requirements elicitation. Card sorting is to ask the stakeholders to sort cards in groups, each of which has name of some domain entity. As in [34], SrvU employs card sorting to characterise the nature and origins of requirements change in the most recent software development cycle. Repertory grids help construct an attribute matrix for entities by asking stakeholders for attributes applicable to entities and values for cells in each entity. As in [35], SrvU employs repertory grids to tackle the terminological interference problem, that is, stakeholders sometimes use different terms to refer to the same concept, and use the same term to refer to different concepts.

Note that the traffic and transportation project under our study started its newest development cycle in April 2010. SrvU shows much interest in incrementally implementing some of the changes resulted from our research in order to improve its RE practices. Table 3 summarises four changes that have been applied to the project. The second column of Table 3 refers to the sample requirements listed in Table 1, showing to which specific requirements each change is applied to. The third and fourth columns of Table 3 show which RE areas the change improves (cf. Fig. 4 and Table 2) and which failure type the change addresses (cf. Fig. 5).

Rapid prototyping refers to an iterative testing process based on the principle that it is easier to change a prototype than the final software system. The goal is to quickly identify potential problems and make adjustments iteratively. This allows not only to find issues in the requirements, but also to have a chance to test the solutions. So far, SrvU has applied rapid prototyping mainly to the user interface requirements, such as R.1 and R.3. The prototypical reporting interfaces (tables, grids, charts etc.) help the end users to describe and prove requirements that developers have not considered.

Feature list provides SrvU's project team a flexible way of documenting salient functionalities of the intended software. According to [36], a feature represents not only the system characteristics that customers view as important in describing the software, but also the function that must be implemented and delivered by the software. When coping with R.3, for instance, the features 'querying', 'filtering', 'tracking', 'zooming in/zooming out' and 'sorting' are documented.

Card sorting is used primarily to handle the requirements evolution problem, that is, change requests that are related to user requirements. In the current iteration of the project, generating a PDF version of the report (R.5) receives higher priority because of the newer distributed administration setting. Although requirements priority changes, card sorting helps keep the sequential order of R.3, R.4 and R.5 intact because R.5 depends on R.3 and R.4. When coping with changes to some other requirements, card sorting helps assess change impact so that the process model can be

Table 3 Changes applied to RE practices

Change implemented	Requirements applied	Practice areas	Failure addressed
rapid prototyping	R.1, R.3	elicitation, communication	interaction
feature list	R.1, R.3	documentation	expectation
card sorting	R.3–R.5	evolution	process
repertory grids	R.2	elicitation	correspondence

adjusted, for example, in terms of resources and time constraints.

A repertory grid is built for R.2 in order to crystallize the concepts involved in the requirement. For instance, 'unique ID' indeed has different meanings: the intermediary user representative proposes to use the batch number, the end users think of a combination of the batch number, rank and status, and even the unit's code, whereas the DBA and the media laboratory coordinator have a computer automatically generated number in their minds. Discovering such a terminological interference [35] helps build correspondence among different stakeholders. The project team has resolved the situation by considering the database's primary key which maps to personnel's information as the unique ID to satisfy the requirement R.2.

The feedbacks we received and the observations we made about the implemented changes are very positive in general. The project members clearly '*see the improvements in requirements-related practices*'. Concerns include keeping the user requirements (feature list) consistent and up-to-date, motivating the stakeholders to be more creative when devising requirements [37] and assessing cost–benefit of the newly introduced RE techniques.

6 Threats to validity

Several factors can affect the validity of our exploratory case study. 'Construct validity' concerns establishing correct operational measures for the concepts being studied [27]. The main constructs in our case study are 'SSM', 'all the RE flaws' and 'improvements suitable for SrvU's context'. As for the first construct, our five-stage design, shown in Fig. 3, was in line with many SSM studies, for example, [16, 30]. As for the second construct, we do not feel that using the generic software failure types [7] as the baseline posed a serious limitation. The failure types provide a coverage reference point for a variety of information systems [7–9], as well as RE-related SPI studies [10–12]. Although requirements-specific error types exist, for example Lauesen and Vinter classified requirements defects according to error source, quality factor, related interface, and cost of handling and repair [29], using a generic scheme helps to identify project and organisation-wide success factors. As for the third construct, because of the lack of metrics for organisational fit, our best measure came from the subjective opinions of SrvU's project members.

Regarding 'internal validity' [27], a major limitation of our study design is the SSM modelling skills and experiences of the researcher, which compounds the problem of experimenter bias [27]. We plan to address the threat by involving multiple experts and stakeholders in developing conceptual models. Another likely confounding variable is the interview data. Participants in our current study may have omitted important facts when answering questions or we may have misinterpreted the data. This threat was mitigated by on-site ethnographic observation, as well as applying pre-defined qualitative data analysis methods (coding and categorising) jointly by the two authors of this paper.

The results of this study might not generalise beyond SrvU's organisational conditions and its traffic and transportation project's situational characteristics, a threat to external validity [27]. Nevertheless, the ongoing industrial-strength project, together with the participation of software professionals, provided a firm footing for applying SSM in RE. Finally, in terms of 'conclusion validity' or 'reliability' [27], we expect that replications of our study should offer results similar to ours. Of course, the experience of SSM modellers may differ, but the underlying trends and implications should remain unchanged.

7 Conclusions

This case study was set up to investigate the role of SSM in understanding RE practices. To that end, we discussed some fundamental aspects of soft systems approach in relation to RE. We then studied a socio-technical software system developed by a research organisation. We found that SSM's holistic thinking could indeed identify a relatively complete set of RE flaws. However, we were unable to confirm that all SSM changes would fit in the organisation's context. Monitoring the actually implemented changes clearly showed RE improvements.

From our experience, we feel that SSM has a rich value in scrutinising and improving the human-centred RE activities. In particular, SSM complements information engineering approaches by enabling the holistic thinking about the requirements process. This further leads to an assessment of organisational fit before potential changes are carried out. Our experience shows that SSM can be applied with low cost and used by non-experts. It is also easy to understand by different stakeholders of a software-intensive system. Nevertheless, more in-depth empirical studies are needed to lend strength to the preliminary findings reported here. Our future work also includes helping SrvU to select a proper set of requirements modelling techniques, estimate the cost–benefit of RE techniques, analyse coordination patterns and tackle requirements interaction problems [38].

8 References

1 Jirotka, M., Goguen, J.A.: 'Requirements engineering: social and technical issues' (Academic Press, 1994)

2 Nuseibeh, B., Easterbrook, S.: 'Requirements engineering: a roadmap'. Proc. Conf. on the Future of Software Engineering, 2000, pp. 35–46

3 Checkland, P.: 'Systems thinking, systems practice' (John Wiley & Sons Ltd., 1981)

4 Rosenhead, J.: 'Rational analysis of a problematic world' (John Wiley & Sons Ltd., 1989)

5 Alexander, I.: 'Migrating towards co-operative requirements engineering', *Comput. Control Eng. J.*, 1999, **9**, (1), pp. 17–22

6 Bennetts, P.D.C., Wood-Harper, A.T., Mills, S.: 'An holistic approach to the management of information systems development – a view using a soft systems approach and multiple viewpoints', *Syst. Pract. Action. Res.*, 2000, **13**, (2), pp. 189–205

7 Lyytinen, K., Hirschheim, R.: 'Information systems failures – a survey and classification of the empirical literature', *Oxford Surv. Inf. Technol.*, 1987, **4**, (1), pp. 257–309

8 Yeo, K.T.: 'Critical failure factors in information system projects', *Int. J. Project Manag.*, 2002, **20**, (3), pp. 241–246

9 Dada, D.: 'The failure of E-government in developing countries: a literature review', *Electron. J. Inf. Syst. Dev. Ctries.*, 2006, **26**, (7), pp. 1–10

10 El Emam, K., Birk, A.: 'Validating the ISO/IEC 15504 measure of software requirements analysis process capability', *IEEE Trans. Softw. Eng.*, 2000, **26**, (6), pp. 541–566

11 Hall, T., Beecham, S., Rainer, A.: 'Requirements problems in twelve software companies: an empirical analysis', *IEE Softw.*, 2002, **149**, (5), pp. 153–160

12 Damian, D., Zowghi, D., Vaidyanathasamy, L., Pal, Y.: 'An industrial case study of immediate benefits of requirements engineering process improvement at the Australian center for Unisys software', *Empir. Softw. Eng.*, 2004, **9**, (1–2), pp. 45–75

13 Boehm, B.W., Papaccio, P.N.: 'Understanding and controlling software costs', *IEEE Trans. Softw. Eng.*, 1988, **4**, (10), pp. 1462–1477

14 McConnell, S.: 'From the editor – an ounce of prevention', *IEEE Softw.*, 2001, **18**, (3), pp. 5–7

15 Platt, A., Warwick, S.: 'Review of soft systems methodology', *Ind. Manag. Data Syst.*, 1995, **95**, (4), pp. 19–21

16 Mathiassen, L., Munk-Madsen, A., Nielsen, P.A., Stage, J.: 'Soft systems in software design', in', in Jackson, M.C., Mansell, G.J., Flood, R.L., Blackham, R.B., Probert, S.V. (Eds.): 'Systems thinking in Europe, basic books' (Plenum Press, New York, 1991), pp. 317–327

17 Probert, S.K.: 'Requirements engineering, soft systems methodology and workforce empowerment', *Requir. Eng.*, 1999, **4**, (2), pp. 85–91

18 Kidd, A.L.: 'Knowledge acquisition for expert systems: a practical handbook' (Plenum Press, 1987)

19 Herbsleb, J.D., Goldenson, D.R.: 'A systematic survey of CMM experience and results'. Proc. Int. Conf. on Software Engineering, 1996, pp. 323–330

20 Software Engineering Institute: 'The capability maturity model: guidelines for improving the software process' (Addison Wesley, 1995)

21 Sawyer, P., Sommerville, I., Viller, S.: 'Improving the requirements process'. Proc. Int. Workshop on Requirements Engineering: Foundation of Software Quality, 1998, pp. 71–84

22 Kaindl, H., Brinkkemper, S., Bubenko Jr, J.A., *et al.*: 'Requirements engineering and technology transfer: obstacles, incentives and improvement agenda', *Requir. Eng.*, 2002, **7**, (3), pp. 113–123

23 Sommerville, I., Sawyer, P.: 'Requirements engineering: a good practice guide' (John Wiley & Sons, 1997)

24 Nikula, U., Sajaniemi, J.: 'BaSyRE: a lightweight combination of proven RE techniques'. Proc. Int. Workshop on Time-Constrained Requirements Engineering, 2002, pp. 69–78

25 Olsson, T., Doerr, J., Koenig, T., Ehresmann, M.: 'A flexible and pragmatic requirements engineering framework for SME'. Proc. Int. Workshop on Situational Requirements Engineering Processes, 2005, pp. 1–12

26 Goldenson, D., El Emam, K., Herbsleb, J., Deephouse, C.: 'Empirical studies of software process assessment methods', in El Emam, K., Madhavji, N.H. (Eds.): 'Elements of software process assessment and improvement' (IEEE CS Press, 1999)

27 Yin, R.K.: 'Case study research: design and methods' (Sage Publications, 2003, 3rd edn.)

28 Beecham, S., Hall, T., Rainer, A.: 'Validating the R-CMM'. Technical report No. 373, Department of Computer Science, University of Hertfordshire, UK, April 2003

29 Lauesen, S., Vinter, O.: 'Preventing requirement defects: an experiment in process improvement', *Requir. Eng.*, 2001, **6**, (1), pp. 37–50

30 Kotiadis, K., Robinson, S.: 'Conceptual modeling: knowledge acquisition and model abstraction'. Proc. Conf. on Winter Simulation, 2008, pp. 951–958

31 Richards, L.: 'Handling qualitative data: a practical guide' (Sage Publications, 2005)

32 Hall, T., Baddoo, N., Beecham, S., Robinson, H., Sharp, H.: 'A systematic review of theory use in studies investigating the motivations of software engineers', *ACM Trans. Softw. Eng. Methodol.*, 2009, **18**, (3), pp. 1–29

33 Coplien, J.O., Harrison, N.B.: 'Organizational patterns of agile software development' (Pearson Prentice Hall, 2005)

34 Nurmuliani, N., Zowghi, D., Williams, S.P.: 'Using card sorting technique to classify requirements change'. Proc. IEEE Int. Requirements Engineering Conf., 2004, pp. 240–248

35 Niu, N., Easterbrook, S.: 'So, you think you know others' goals? A repertory grid study'', *IEEE Softw.*, 2007, **24**, (2), pp. 52–61

36 Kang, K.C., Kim, S., Lee, J., Kim, K., Shin, E., Huh, M.: 'FORM: a feature-oriented reuse method with domain-specific reference architectures', *Ann. Softw. Eng.*, 1998, **5**, (1), pp. 143–168

37 Karlsen, I.K., Maiden, N.A.M., Kerne, A.: 'Inventing requirements with creativity support tools'. Proc. Int. Working Conf. on Requirements Engineering: Foundation for Software Quality, 2009, pp. 162–174

38 Robinson, W.N., Pawlowski, S.D., Volkov, V.: 'Requirements interaction management', *ACM Comput. Surv.*, 2003, **35**, (2), pp. 132–190