Prototyping RE Experiments in the Classroom: An Experience Report

Birgit Penzenstadler
Institut für Informatik
Technische Universität München
Munich, Germany
penzenst@in.tum.de

David Callele
Department of Computer Science
University of Saskatchewan
Saskatchewan, Canada
callele@cs.usask.ca

***Abstract*—In this work we investigate the feasibility of prototyping industrial requirements engineering experiments within an educational environment, *i.e.* conducting a prestudy with students before performing the experiments in industry. We identify a set of constraints on the experimental design intended to make research participation more rewarding for our industrial partners and investigate the complexities of meeting both research and learning objectives within the same experiments. We report our observations and conclude that designing effective requirements experiments for an industrial environment, sensitive to industrial constraints, is a very difficult problem. Specific educational recommendations in visualization, prioritization, and customer interaction are also presented.**

Keywords: Requirements elicitation, experimental design, education, pedagogy.

I. Introduction

Designing requirements engineering experiments is challenging and taking them into the field can be very expensive. In this work we investigate the feasibility of designing RE experiments and then testing them within the classroom environment before performing them in an industrial setting. We propose the use of this prestudy approach to evaluate the experiments as a whole and in their parts – identifying elements that work and those that need improvement.

However, using the classroom implies the potential for a confounding factor: Given that students are participating, we also have soft-goals [7] that the experiments have defined learning outcomes that benefit the students. This can complicate the experimental design and this work investigates whether these potentially competing needs can be satisfactorily addressed in practice.

We focus here on experiments that investigate aspects of the assumption underlying the field of requirements engineering – that the stakeholders can effectively communicate regarding the artifact. Of particular interest is the communication of intent during an iterative requirements elaboration process 1.

We define requirements elaboration as the (typically iterative) process of requirements elicitation, capture, and representation. We distinguish between these elements based upon our experience. We are most interested in participative elicitation techniques such as collaborative sessions and ethnographic techniques, techniques used to bring forth the requirements during human interactions [14]. We further distinguish between capturing and representing requirements – capturing requirements is performed by the practitioner using a variety of means such as scribbled down notes and roughly drawn diagrams, as notations on a white board, or even as audio and video recordings. We consider captured requirements to be an intermediate step in the requirements elaboration process. These captured requirements are then analyzed and edited, usually in a *post hoc* fashion, and represented in a cleaned-up, and often more formal, state. These requirement representations are then used as the basis for efforts in other requirements tasks such as negotiation and prioritization.

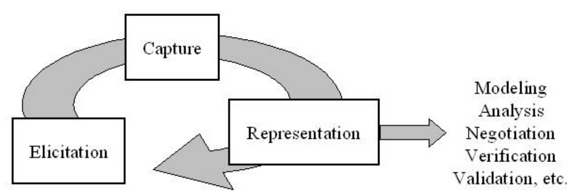

Figure 1. Requirements Elaboration

The distinction between capture and representation is made based on our observations of industrial practices. The early stages of the requirements process often delve into design and implementation issues as the consequences of proposed requirements are explored. These meetings address more than software issues and techniques for capturing these iterations are needed, especially for traceability and for capturing the rationale behind various decisions. Once this information is captured, it must be translated into a form that can be used for performing other requirements tasks and for forming part of the permanent record for the project.

In the remainder of this work we review the related work on requirements elicitation then discuss the design guidelines for our experiments and the constraints thereon. We describe each experiment in detail, identifying the hypothesis for each experiment, the criteria used to evaluate the results and the threats to validity we identified. We report the results and offer commentary first on the individual experiments then upon the set of experiments as a whole – identifying the strengths and weaknesses of the experiments and the lessons learned about designing requirements engineering experiments subject to our self-imposed constraints. Finally, we present our conclusions and recommendations for future work.

II. Related Work

This work is motivated, in part, by the authors' personal observations that communication with some customers during requirements engineering elicitation efforts, using a formal representation, can lead to difficulties. Formal representations can be intimidating[1], and even the best visualization techniques [21] can not always break down the communications barriers. Our domain wisdom insists that requirements engineering efforts should be practiced in the language of the customer – yet how many of our customers use representation techniques like i*, temporal logics or predicate logics in their daily discourse? This observation is not intended to diminish these techniques in any way – they have proven the strength of their contributions in theory and we may well rely heavily upon them in practice but they are not always part of the stakeholder's regular communications regime.

Requirements engineering is predicated, in part, upon the premise that the probability of success in a development effort is directly correlated with the quality of requirements (as defined by their suitability for defining the task at hand [2], [1]). Requirements engineering techniques have evolved from simply identifying what the customer wants to something far more complex. Today, requirements engineering efforts usually determine, at a minimum, the customer's wants and their needs and attempt to balance them against their resources to define the target for the development effort.

A long list of elicitation, capture and representation techniques have been developed [24], [17], [19], [12], [25] to support these efforts. Dieste *et al.* [9] present a systematic review of the range of software requirements elicitation techniques while Hickey and Davis [14] describe the elicitation process and how experienced analysts select elicitation techniques for a given context. Research findings on general empirical evaluation of requirements specifications approaches are reported by Condori-Fernandez *et al.* [8] and Easterbrook *et al.* [10] report on experiences using lightweight formal methods for requirements engineering. This prior work provides significant qualitative guidance on the suitability of given techniques within particular scenarios. However, we are unaware of any prior work that attempts experimental validation of some of the most basic assumptions in requirements elaboration such as the quality of stakeholder communication.

Nuseibeh and Easterbrook presented an overview of RE and key open research issues in their RE roadmap [22] identifying "Bridging the gap between requirements elicitation approaches (. . .) and analysis techniques" as a key research issue. Nearly a decade later Cheng and Atlee [5] provided a followup for that roadmap wherein they recommend placing more emphasis on academic RE education and training as well as on evaluation and empirical research [5, p. 32/33]. Both serve to facilitate the use of RE techniques, one of the identified "RE research hotspots" [5, p. 22/23]. This work is positioned at the intersection of these recommendations and observations.

[1]Even when they are stated to be "very simple", as Lamport described temporal logics [18, p. 872]

III. Designing The Experiments

In this section we present our self-imposed design constraints for the experiments, the justification for these constraints, then describe the environment within which the experiments were run.

A. Design Constraints for Industrial Experiments

This set of experiments is intended to explore aspects of requirements elicitation, capture, and representation. Typical requirements engineering experiments expect that the experiment continues until a software artifact is delivered. The overall development effort, and the software artifact, are then inspected to determine the success of the requirements effort. However, this methodology is very time-intensive and requires significant investment. More timely, lower-cost alternatives are desirable if we are to make participation more attractive to industrial participants. We have placed the following constraints upon our experimental designs in an effort to move toward achieving this goal.

Minimize time commitment for study participants: We find it significantly easier to receive commitments to participate on the part of our industrial contacts if we can reassure them that the experiment is "small" and will only take up the time equivalent to a typical meeting. Participation in larger experiments, requiring commitments for half-day or full-day workshops has been much more difficult to negotiate. Given that setup time, contextual training, *etc.* can take 30 to 60 minutes, we have set a goal of one hour for the experiment itself.

Obtain (relatively) rapid feedback: Many requirements engineering experiments require that software be implemented and that the project has proceeded to the validation phase. A set of experiments whose results can be interpreted as soon as possible are necessary to be able to demonstrate return on investment for study participants – ideally the

978-1-4244-8786-8/10/$26.00 ©2010 IEEE

experiments are independent of other software development process activities and have quantifiable, or qualitatively demonstrable, benefits.

The focus of the experiment (dependent variable) is independent of the context: The researchers should be able to change the accompanying contextual setting without affecting the validity of the experiment. This facilitates deploying the experiment in a variety of contexts, potentially enabling industrial participation from a wider variety of subject domains.

Experimental design can be evaluated within the classroom: This returns to the focus of this work: As researchers, we tend to start, or even perform all of, our experiments in the classroom or some other academic setting. Gaining access to practitioners and projects not only increases the cost of performing the research, it may put the experimental subjects at risk by contributing to the failure of their projects: not all theories work in practice and we are very cognizant of this in our work. By designing the experiments so that they can be prototyped and tested for their effectiveness as experimental tools within an academic setting, we minimize the risk of flaws in the experimental design before investing in field research. We may also be able to be able to offer some supporting evidence to potential industrial participants to help induce them to participate in further studies.

Modeling multi-disciplinary stakeholders: Stakeholders, in practice, can have widely divergent perspectives and needs – we often refer to this heterogeneity as *multi-disciplinary*. When experiments are run in an academic environment, students typically take the roles of the stakeholders and of the requirements practitioners. It may be difficult to generalize the experimental results to the domain given that the students may not be representative of this heterogeneity. Further, the fact that the students are all taking the same class implies a commonality within their training, and this may introduce bias in their perspectives on issues. However, careful management of the students that reinforces the role-playing nature of their participation can mitigate some of these issues.

B. Study Context for Student Experiments

In this first phase of our research, four experiments were designed to explore the validity of assumptions underlying aspects of requirements elicitation practices. The specific hypothesis for each experiment is detailed in the relevant sub-section.

Setting and Motivation: The experiments were situated within a post-graduate (Masters) class that provides a general introduction to requirements engineering. From the students perspective, the experiments were conducted within the tutorial session associated with a weekly lecture on requirements engineering and were simply part of the course material, although they were informed that we were investigating the matters reported here. The term is 13 weeks in duration and both lectures and tutorials are 90 minute sessions, each held once a week. Students also receive take-home exercises that are expected to require 2 to 3 hours to complete. Solutions to the exercises are discussed in the next tutorial session.

Students: There are 15 students in the class, ranging in age from approximately 20 years of age to approximately 40 years of age. The student backgrounds are very diverse – there were students from both the "ordinary" Informatics Master's program and the newly designed Automotive Software Engineering Master's program. While the majority of the students are German, there were students from three other countries in the class. None of the students reported that they had received anything more than a cursory introduction to requirements engineering in their prior experience.

Instructors: The first author of the paper, one of three tutorial session leaders for the class, managed the experiments. The experiments were held within the tutorial sessions. A colleague from the Faculty of Mechanical Engineering participated as a "real life customer" for two of the experiments. The colleague is an experienced mechanical product engineer but does not have any experience in software engineering – in other words, a typical stakeholder with strong subject matter expertise.

IV. THE EXPERIMENTS

Details of the four experiments are presented in this section. For each experiment, we describe the context for the experiment, identify the assumption that we are investigating, the learning outcome and the experiment itself. We state the hypothesis, the criteria used to evaluate the results, the threats to validity and close with a discussion of the experimental results.

A. Experiment 1: Capture Spoken Requirements

Software requirements are typically generated during discussion and prototyping sessions attended by project stakeholders. Functional requirements are usually captured as prose but aspects such as look and feel are more readily captured as visualizations. Short annotations added to a (jointly developed) image (*e.g.* a rich picture [20] or frameworks such as UML or i*) can be very effective mechanisms for capturing these requirements. However, the long-term efficacy of these annotated images is unproven – the annotations are highly context-dependent and may have meaning only to those individuals who were present at their construction or so long as they remember the context.

The purpose of this experiment is to investigate the assumption that verbally delivered requirements can be captured in a satisfactory manner. As a corollary to this assumption, practitioners assume that the elicit-capture-represent process can be as effective at generating requirements as when stakeholders generate requirements themselves.

The associated learning outcome for this experiment is that the students learn awareness of requirements that exist external to the documented requirements and that requirements engineering efforts must address these external requirements.

The students were given a set of 26 written requirements and were asked to structure them in a feature tree [2] visualization [16]. The feature trees were then discussed with the students and in a second exercise the students were required to document use cases and scenarios. During the second exercise, seven more requirements were verbally added during the discussions. The feature tree is the basis for visualization efforts and the verbal requirements must be captured and added to the feature tree as annotations.

- *Hypothesis:* Requirements can be extracted from discussions and captured as extensions to existing requirements.
- *Evaluation criteria:* All oral requirements are identified and added to the feature tree.

Results: Some of the students did document that there were additional requirements, others proceeded directly to the use case and scenario documentation without annotating the feature tree. Not all solutions explicitly included the new requirements; the participants were asked to write down use cases and scenarios and some chose scenarios that did not necessarily show the additional requirements.

- *Participants: 12*
- *Correctness of results:* 9 participants included the additional requirements in some way.

Some, or all, of the additional requirements were included only in the scenarios of the students that did annotate the additional requirements on the feature tree. However, given that some of the other scenarios did not necessarily interact with the additional requirements, it can not be stated for sure that the participants would have forgotten to include the additional requirements – but it may be assumed that some of the students would have forgotten.

In order to be able to draw appropriate conclusions, we have learned that the experimenters need to have very tight control over the wording of the directions to the participants. We received feedback from the participants that they did not understand that the additional requirements needed to be annotated on the feature tree first, or at all, and we are unable to conclusively state whether this was the fault of the participants, a weakness in their academic instruction, or the specific instructions during the experiment.

Independent of the reason for not capturing the requirements as expected, participants received further instruction on traceability and rationale, and guidance that additional requirements should always be explicitly documented before moving on to use case and scenario description or other tasks.

[2]Feature trees were used as the visualization paradigm because the students had recently completed formal instruction in the technique.

B. Experiment 2: Prioritization

Another issue that arises in software development is prioritization in the face of rampant creativity. Requirements efforts may delve in and out of a lightweight design phase and can produce an excessive number of options that must be reduced and prioritized to meet production constraints.

In this experiment, participants are presented with 10 written requirements that have been divided *a priori* into the categories of low, medium, and high priority to ensure consistency of response to questions. However, the participants are not directly informed of the stakeholder's priorities. Instead, the participants must infer the priority from the stakeholder materials or directly query the stakeholder for the information.

This experiment investigates the assumption that requirements priority can be successfully elicited in a multi-disciplinary project. As a corollary to this assumption, practitioners also assume that requirements priority can be successfully negotiated in a multi-disciplinary project. The student is expected to learn that stakeholders set priority, not requirements practitioners, that the *customer's* prioritization is more important than their own and that attempting to deduce priorities, without stakeholder confirmation, is dangerous. Further, the practitioner must learn to discriminate between "customer priority" and "impact upon the project".

- *Hypothesis:* Participants will take the assignment as given and not challenge the stakeholder for priorities.
- *Evaluation criteria:* All requirements are prioritized and classified (completeness), with supporting rationale and/or argument (correctness).

Results: The classification was a difficult task for the requirements were taken from a real case study [11] and we did not invent and insert completely irrelevant requirements. In the absence of asking the stakeholder, the prioritization does in fact depend on the point of view: a test engineer might prioritize a specific requirement differently than a sales person.

- *Participants:* 12
- *Correctness of results:* 11 participants delivered well-reasoned arguments in support of their prioritization efforts. However, none of the students attempted to confirm the priorities with the stakeholder.

As can be seen in Table I, the number of requirements classified into each rank diverged significantly. The correlation with the *a priori* rankings was insufficient to draw any conclusions.

This experiment confirmed our hypothesis that the participants would not confirm their prioritization results with the stakeholder. This experiment is not really 'fair' to the students since the hypothesis expects the students to fail to

Table I
REQUIREMENTS PRIORITIZATION IN STUDENT EXPERIMENTS

Req.	High	Medium	Low	(A Priori)
R1	7	2	3	high
R2	6	6	0	high
R3	2	5	5	medium
R4	5	6	1	medium
R5	2	2	8	low
R6	3	5	4	medium
R7	5	3	4	high
R8	6	3	3	high
R9	5	3	4	medium
R10	11	0	1	high

take the initiative and ask the stakeholder for their priorities. However, the students were not evaluated on this point so we were able to perform the experiment and teach the lesson without any negative impact on the participant's perceived academic performance.

Fortunately, the results for this experiment identified weaknesses in their instruction and subsequent practice. The participants do not yet understand that they must subordinate their judgment to the stakeholder's needs and desires, at least during the elicitation phase. This result provided us the opportunity to apply corrective action early in the course, before the misconceptions had a chance to become ingrained.

It also became obvious that the students did not understand the difference between the requirements engineering definitions of *prioritization* and *impact*. In almost all cases, the students assigned priorities based upon what they perceived to be the impact of the requirement upon the development process and not upon the customer's priorities. This is a common misconception that can lead to the requirements practitioner substituting their judgment for that of the customer. This observation led to further emphasis, with the participants, on the differences between the concepts.

If we assume that the participants were asked to evaluate potential impact on the project rather than priority, we would then have been faced with the difficulty of choosing between the submissions. In the absence of objective metrics, we would be forced to rely upon the submitted arguments. In general, the arguments were sound but there was still significant divergence of interpretation and opinion among the students.

C. Experiment 3: RE Top Model Contest "Reloaded"

Many design practitioners assert that highly visual techniques, such as rich pictures, are an effective tool for communicating between stakeholders. The requirements engineering literature contains many works that assume that these highly visual techniques will also succeed at capturing requirements during elicitation efforts. Attendees at RE'09 had the opportunity to attend the "RE Top Model" session, organized by Gotel and Huang [13], an exuberant and entertaining session where various teams 'competed' in their attempts to demonstrate that their technique was the 'best'. The authors used rich pictures [20] as their technique and demonstrated their potential for effectively capturing complex models during the elicitation phase.

Inspired by this contest, we attempted to engage the participants in a similar exercise. The participants were divided into four teams, two teams employing text and UML methods and two teams employing rich picture methods. Our goal was to evaluate the methods for their suitability for communication with a non-software customer during requirements elicitation. All teams received similar training in all techniques.

A researcher from another department agreed to act as the client. She described her fictitious industrial background, market situation and her vision of the system to be developed. Immediately after the customer presented their concept pitch and their requirements, the students (as a whole) were allowed to question the customer for approximately 10 minutes. The questions were well constructed and focused on issues like determining the exact scope of the project, hardware interface details, marketing intentions, exceptions, and exact procedures.

The students then had 15 minutes to sketch their thoughts and structure their notes. Two teams modeled using traditional requirements engineering techniques (*e.g.* natural language plus UML) and two teams modeled the "colorful" way – with rich pictures. Each team was then asked to present their requirements to the client.

The intent of this experiment was to expose students to a (simulated) real-life client without software engineering experience who describes their system vision. The students are encouraged to interact with the client to ensure their understanding of the problem (elicitation and capture) prior to documenting the requirements (representation) to show the client they have understood their idea and problems.

In this experiment we are attempting to understand how practitioners might employ highly visual techniques, such as rich pictures, with multi-disciplinary projects and their attendant requirements. The students were expected to gain direct experience with the effectiveness of visualizations as means for communicating with the customer during an initial requirements elicitation session. The customer does not have to learn a formal representation, the rich pictures approach is easier to understand than a formal representation, and it is expected to perform better than natural language text as a mechanism for supporting interactive discussion.

- *Hypothesis:* Rich pictures are superior to the "traditional" approach of natural language and UML for communication with the customer during requirements elicitation.
- *Evaluation criteria:* How well did the students understand the customer's problem? How well did the students capture the problem? Criteria include correctness, completeness, conciseness, and understandability.

How well can the customer understand the documented requirements? Does the customer feel their problem is well understood and well represented?

Results: In general, the participants did not adopt the rich picture technique in an appropriate manner. The participants are all engineers, trained in methodical, refinement and decomposition-oriented techniques and they found it difficult to adopt such a free-form, exploratory and expressive technique. The visualizations that the rich pictures teams used were actually feature trees – a visualization that they were familiar with and which was strongly related to their prior training.

The participants were excessively concerned with the fidelity of the (visualized) model and not concerned enough about communicating with the customer or about expressing and capturing concepts in a manner that the customer could understand.

It appears that RE practitioners may require more extensive training in informal visualization techniques in order to better be able to utilize them. We are currently investigating this issue and can recommend that interested readers begin their investigations into some of the available visualization techniques at the Open University, Systems Thinking website [27]. Practitioners employing these techniques may be assisted by employing artistic abstractions such as metaphor and simile in their work [4], [15], in addition to our traditional engineering abstractions and modelling techniques [26], [6], [23].

Table II
ASSESSMENT OF REQUIREMENTS DOCUMENTATION.

Team	Solution/Result	Customer's Opinion
Color 1	The result was not really a rich picture but rather a feature tree for customer and service personnel.	"Very well understandable, it was the only solution that included the maintenance service for the coffee machine and the interaction with other services, function-oriented, but different types of relationships were mixed and could not be distinguished."
Color 2	Not really a rich picture, rather kind of a scenario tree that represented how the problem was understood.	"Good because it did not yet imply a specific solution."
Text 1	Collection of loose notes about what the system shall be able to do which were roughly sorted but not yet structured due to lack of time.	"It was good that they used my exact numbers, but more explicit structuring would have been helpful."
Text 2	Scenario description with customer Bob using the machine.	"General idea is well captured but this is only one of many scenarios. Missing completely is, for example, the maintenance service part. Strongly function-oriented but other elements are completely left out."

Conclusion: The results and the comments of our customer are described in Table II. Unfortunately, neither of the "colorful" student teams actually used rich picture-like techniques. Instead, they stayed with what they had seen before, using representations like feature trees and scenarios. They were reluctant to use the given freedom (emphasized in the lecture) for the visual representation. Our client did express a preference for the "close-to-picture" approaches relative to the textual approaches.

D. Experiment 4: Traceability & Rationale

Given the characterization of media and discussion-rich requirements sessions as described in the prior section, requirements traceability and accompanying rationale are particularly challenging to maintain in such a fluid environment. Requirements traceability and rationale are considered difficult to achieve in a project that emphasizes functional requirements. Will the techniques work even in relatively uncertain environments such as those employing agile software development techniques?

The students were required to refine the initial requirements according to the schedule presented below. At the end of the experiment, the students were asked to identify the source of the changes and the rationale.

Experiment: This experiment was performed as follow-up to Experiment 2 (Sec. IV-C). On day 1, the initial requirement definition was generated in discussion with the customer. On day 2, we originally planned to have further discussion with stakeholders, wherein the requirements changes would be inserted. Instead, we actually performed two updates via email in which one completely new requirement was added and one change to an existing requirement was made (functional requirements were used to make the experience more tangible to the participants). The participants were requested to submit their requirements documentation within 5 days. On day 7 the students finalized and submitted their requirements. The results were discussed with the students on day 8.

All traceability techniques assume that requirements practitioners can maintain acceptable records of requirements evolution and accompanying rationale during a requirements elicitation discussion. Numerous proposals have been made but they all rely upon the practitioner's diligence for success. The students are expected to learn that practices such as editing requirements during a meeting can be very dangerous if the project state must be rolled back or if the rationale for earlier-stage decisions must be identified.

- *Hypothesis:* Participants will fail to capture traceability information and rationale information for their requirement.
- *Criteria:* Evaluate presence of prior versions, revision history. Check for presence of rationale, validate rationale as captured.

Results: Classroom timelines required dynamic changes to the experimental design. Rather than the planned discussion leading to changes, two change requests were given to the participants via email. One change request was a modification of an existing requirement, i.e. to different payment options. The second change request was the introduction of a new requirement, i.e. to two more alternative selection options. While all solutions did include the requirement for supporting the payment options, only half of the solutions documented changes to the requirement of additional selection options (via updates to the requirements). None of the submissions captured rationale associated with either change.

- *Participants:* 4 teams
- *Correctness of results:* 2, but we have to differentiate: Two teams explicitly included a new requirement for additional selection options, the other two teams only documented that there were "different types for selection". All four teams noted that there was a change and captured another new requirement, but none explicitly captured the rationale.[3].

In general, we can say that the participants attempted to capture new requirements but were less likely to formally note changes in the requirements or to capture the reasons for the changes. The participants did not seem to understand that *why* a decision was made can be at least as important as the results of the decision. It may be that this lesson is usually learned only with the experience of something going badly and then trying to trace the rationale to identify responsibility.

It appears that stakeholders may find it easier to understand the value of capturing the requirements than the value of capturing the rationale. The requirements are relatively tangible since they relate to aspects of the product focus. Rationale embodies the wisdom of the decision-making process and participants may not comprehend why decisions were made. Given that the participants were all in a requirements engineering class, we would expect them to be particularly sensitive to requirements issues – members of other software development teams may exhibit even weaker performance.

V. Observations and Lessons Learned

We look first at the self-imposed design constraints, then the results of the experiments in the context of their hypotheses. We then review the threats to validity for the experiments, and conclude with comments on the lessons learned from the experiments, how the results inform our pedagogy, and some general comments on the experience.

A. Experimental Design Constraints

Prior experience with designing experiments of this nature led us to believe that designing experiments that met our constraints would be difficult and our expectations were proved correct.

Effort: The participants were able to complete each of the planned experiments within the target time frame of one hour. Therefore, similar experiments might be equally suitable for an experiment session with industrial practitioners. However, experiments that illustrate the importance of traceability, for example, may require that the effort be spread over considerable periods of real time in order to illustrate the points. This ongoing commitment may be an issue for potential participants.

Feedback: The designed experiments did not require the implementation of software artifacts and even the longest running experiment could take place within a couple of weeks or real time. Therefore, participants received relatively rapid feedback. However, we have some concerns as to the credibility of the results with industrial practitioners if the projects are not taken to completion – for example, to demonstrate that a particular RE technique actually leads to measurable improvements during, for example, the integration phase or acceptance testing.

Context: The dependent variable (focus) of each experiment was chosen such that it would not affect the validity of the experiment when performing it within a different business application domain. Our experience in this work has lead us to conclude that this approach is feasible for the particular hypotheses that were evaluated and that it shows sufficient promise that other researchers may find it to be a beneficial guiding principle during the design of their experiments.

Experimental design: Using the classroom instead of an industrial setting for the context allowed us to be more relaxed with respect to the outcomes of the experiments – we did not have the potential for inducing project failure as a risk that we had to manage. As a result, we could encourage the students to "go crazy" in using creative visualization techniques (although they were reluctant to use that freedom) without having to fear that we might put a project's success at risk.

Multi-disciplinary stakeholders: The diversity of our students was significant, from senior master students who have already been working in industry for up to ten years and master students who had just finished their bachelor exams. The lively discussions and diversity of opinions during the tutorials were strong indicators that this group of students, at least, had sufficient diversity to mimic an industrial context.

We conclude that we have met our self-imposed constraints on the experimental design and that these constraints do not limit the validity of the experiments. However, we need to perform further work to determine whether other researchers and other practitioners find these self-imposed constraints to be reasonable and beneficial.

B. Relation to Hypotheses

We were able to prove three out of four hypotheses with our experiments and partially prove the final hypothesis.

[3]This was actually a requirement invented / introduced by the students NOT the client. Our client then decided to include that requirement, thereby making the realization of the system significantly more challenging. Such interaction can be judged positively as the students tried to inquire what else the client would probably want, on the other hand the students are not supposed to encourage the client to make their desired system more complex via the addition of new features.

In summary, the responses to the hypothesis from each experiment are as follows.

Successful extraction of requirements from discussion: Partial support. It is possible, but not all students documented the extracted requirements properly.

Failure to capture traceability and rationale: Proven. The students captured neither rationale nor tracing information. Increasing the emphasis on these points in RE classes may be necessary.

Priorities not challenged: Proven. The students did not challenge any of the priorities, they simply assigned them according their estimates of their estimated impact upon development effort. However, the pedagogy may be an issue here and we hope to rerun the experiment with tighter controls.

Rich pictures improve communication: Proven. The client did prefer the solutions that were more visual in style, even though the students did not use the expected visualization technique – instead, choosing to use feature trees.

Whereas we can say that we were successful, in the abstract, we recognize that we still face significant challenges in determining correlation and causality within the results. Modifications to the experimental designs may be necessary.

C. Threats to Validity

1) Internal Validity: In Experiment 1 the number of requirements was small compared to industrial practice. Further, the additional requirements were relatively clearly stated, implying little opportunity for confusion by the participants; the oral requirements were principally additional features and the requirements were either remembered or they were not.

In Experiment 2 there were only 10 requirements so their priorities may have been too obvious (or too subtle). The context and rationale for the prioritization decisions were not shared among the participants. The experiment also employed a degree of misdirection, assuming that the participants would learn the lesson “the hard way”.

In Experiment 3 there were only two student groups per approach and the assessment was the personal impression of only one “customer”. Our ability to generalize is very weak and reflects the ongoing issues with recruiting substantial numbers of participants for RE experiments.

In Experiment 4 the participants may not have had sufficient experience to have the appropriate perspective regarding the importance of the tasks. Further, the penalties associated with failing to capture the requisite information may not have been sufficient to motivate the participants to perform at the necessary level.

2) External Validity:

Statistical validity: There are potential issues with statistical validity with any of our results. Even if there were many student participants, does the student population accurately represent the target population?

Creativity: The students did not want to be artistically creative, they used structured diagrams like scenarios, feature trees. They were focused on the technical issues and how to realize the requirements. They were very analytical but weak in communications. They focused on extracting information that *they* felt was necessary but they did not necessarily suppress their opinion and elicit, from the customer, what the customer felt was necessary. This behavior might be different in another application domain, such as entertainment software, where the practitioners have a greater probability of being artistically inclined.

Application Domain: It is very difficult to determine whether the student populations were representative of the practitioner domains. The student population was sufficiently diverse to be illustrative of the issues associated with stakeholder diversity but may not have been sufficiently homogeneous to illuminate issues representative of traditional development teams.

D. Lessons Learned for Design of Experiments

Prestudy: A staged evaluation of experimental designs, using student volunteers to reduce risk of failure in the field, appears to work well. The technique is particularly helpful in identifying places where miscommunication is occurring, affording us the opportunity to rectify issues before going to the field.

Appropriateness: Experiment 2 was not an appropriate experiment to determine whether the students understand how to evaluate the impact of a requirement upon project scope, there were too many variables to be effective. Table II demonstrates that the student responses were excessively divergent.

Researcher Bias: The academic researcher’s familiarity with student populations and their behaviors make it relatively easy to design experiments that correct for perceived biases. However, these corrections, themselves, may introduce biases into the experimental results. For example, the pedagogical lessons detailed in Section V-E may lead to biases that accumulate as successive experiments are performed and the design of the experiments is updated.

E. Pedagogical Lessons

The pedagogy associated with each of these experiments has been revised as a result of this work. In particular, there is greater emphasis on communication with the customer, encouraging creativity, and teaching the difference between customer priority and developer priority (impact).

Experiment 1: Teach students that they need to identify accepted / acceptable practices for requirements management and that these practices may be project dependent.

Experiment 2: Ensure that students understand that they have to subjugate their will and knowledge during the elicitation phase. The difference between the concepts of priority (given by the customer) and impact (estimated by the developer) has to be clearly communicated. Further training on interview techniques is necessary to help students learn how to correctly interpret the customer's perception of priorities.

Experiment 3: Further training in unstructured visualization techniques is needed. The students were reluctant to adopt the presented techniques and we observed discomfort during the experiments. The importance of communication with customers outside of the IT domain is sometimes mentioned, but practical techniques may need to be taught.

Experiment 4: Students need to understand the consequences of not capturing the 5 Ws (who, what, where, when, why) so that they can make informed risk assessments. Prior attempts at constructing experiments of this type resulted in dramatic student experiences [3], we continue to search for less painful ways of imparting this knowledge.

Learning without trauma: The old adage that people learn by making mistakes exists because it is generally true. However, as educators we are under significant pressure to ensure that the learning environment is constantly and consistently supportive. Are there ways to teach students topics that have, associated with them, significant negative consequences for failure in practice? Topics such as prioritization errors leading to project failure, failure to capture rationale leading to legal liabilities imply traumatic experiences – do we truly train the students or do we simply pass them along?

F. General Comments

Validate experimental methods: Requirement elicitation, capture, and representation is highly dependent upon human interaction. As a result, controlling all of the parameters is effectively impossible and we can only aspire to do our best. We have found that testing the experiments in the classroom, before proceeding to field tests, can expose unexpected participant behaviors that can be compensated for before proceeding to field tests.

We remain concerned over aspects of experimental control. For example, when we attempt to reduce the number of possible misinterpretations for a given problem description or even a particular requirement, we find that we can reach the point where the information communicated to the participant is so rigidly defined that there is nothing left for them to interpret. How does this affect the learning process – are the students being properly prepared for real challenges? Do our attempts to eliminate these confounding factors actually bias the results? Is requirements engineering really just about teaching communication skills for that can be applied to communication in highly specialized domains?

Validity of experimental results: The size of the experimental populations and the number of confounding factors are significant challenges, particularly if one wishes to attempt to draw quantitative conclusions with statistical support. Resource constraints, including time, are severe compared to the number of variables that we are attempting to manage.

For example, when working with student populations the fact that the students are all taking the same class implies a commonality within their training, and this may introduce bias in their perspectives on issues. However, we rarely see this issue explicitly addressed by the researchers.

To a 'hard' scientist our discipline can seem vague or even lacking in credibility due to an inability to implement rigidly controlled experiments. Wieringa [28] provides an interesting perspective that can help position our work – as "engineers of the engineering cycle". If we accept his position, what effect does this have upon our pedagogy?

Long term visualization: Rich pictures appear to work well for elicitation and capture but we have concerns about rich pictures for long term representation – especially if there is a lack of accompanying documentation of the 5 W's. The (intentional) low fidelity is designed to prompt recall and illustrate relationships – if the viewer was not present when the visualization was generated, there may be issues of misinterpretation. Some of the students were also uncomfortable with free-form sketching techniques, probably experiencing a fear of failure since they are not necessarily trained as artists. Does this imply that formal training in sketching techniques should become part of the expected training for requirements engineers?

Change history: Our students/practitioners appeared to be unaware of the dangers of dynamically editing requirements (without keeping an appropriate change history) during a meeting if the project state must be rolled back or if the rationale for earlier-stage decisions must be identified. Further lessons or experiments are suggested.

VI. Conclusions and Outlook

This work has exposed and illustrated some of the difficulties associated with designing and implementing short-turnaround requirements elicitation experiments. We are able to report our observations and can provide modest qualitative guidance but significant work remains to increase confidence in our observations. Nevertheless, our initial results have convinced us of the worth of performing a prestudy in a relatively controlled environment, providing benefits of improvements to our experimental designs and to the associated pedagogy.

There are challenges. Requirements engineering is most often a group activity. We can work with (small) student groups but meeting this goal comes at the cost of (statistical) confidence in the results – the number of students, and our ability to manage them, are limited. Biases in the participants, particularly (relatively inexperienced) students, can overwhelm the observations we are attempting to make. Observer bias and observer error also become more significant with shorter experiments; longer timelines allow the experimenters to observe participant behavior and to deduce the 'correct' interpretations of their actions. Finally, the content of the experiments and how they are conducted within the classroom must meet classroom content guidelines, human rights legislation, privacy legislation, *etc.* Can these constraints be met and the experiment prototype phase still deliver suitable information as to the experiment's suitability as an experimental instrument?

In the future, we would like to continue to work toward our goal of identifying requirements engineering experiments that can deliver meaningful results with relatively small time investments by study practitioners. We are also interested in attempting to identify biases in participant populations and on models or arguments that more clearly identify the "return on investment" for participants.

Acknowledgements: We would like to thank Julia Roelofsen for being our "real-life customer" during the experiments, Andreas Fleischmann for his support in conducting the experiments, the students of the RE'09 class for their active participation and Klaus Lochmann for feedback on an earlier version of the paper.

References

[1] Berenbach et al. *Software & Systems Requirements Engineering: In Practice*. McGraw-Hill Osborne Media, 2009.

[2] Barry Boehm. A View of 20th and 21st Century Software Engineering. In *ICSE*, 2006.

[3] David Callele and Dwight Makaroff. Teaching requirements engineering to an unsuspecting audience. In *SIGCSE '06: Proceedings of the 37th SIGCSE technical symposium on Computer science education*, pages 433–437, New York, NY, USA, 2006. ACM.

[4] Julie Cameron. *The artist's way*. Jeremy P. Tarcher/Putnam, 2002.

[5] B.H.C. Cheng and J.M. Atlee. Current and Future Research Directions in Requirements Engineering. In *Design Requirements Engineering: A Ten-Year Perspective, Workshop, Cleveland, 2007*, page 11. Springer, 2009.

[6] C.K. Chua, K.F. Leong, and C.S. Lim. *Rapid prototyping: principles and applications*. World Scientific Pub Co Inc, 2003.

[7] Lawrence Chung. *Non-Functional Requirements in Software Engineering*. Kluwer Academic Publishers, 2000.

[8] Condori-Fernandez et al. Research findings on empirical evaluation of requirements specifications approaches. In *12th Workshop on Requirements Engineering*, 2009.

[9] Oscar Dieste, Marta Lopez, and Felicidad Ramos. Updating a systematic review about selection of software requirements elicitation techniques. In *11th Workshop on RE*, 2008.

[10] Easterbrook et al. Experiences using lightweight formal methods for requirements modeling. *IEEE Transactions on Software Engineering*, 24(1):4–14, 1998.

[11] Feilkas et al. A Top-Down Methodology for the Development of Automotive Software. Technical Report I0902, Technische Universität München, 2009.

[12] M. Glinz. On non-functional requirements. In *Proc. RE*, volume 7, pages 21–26. Springer, 2007.

[13] Olly Gotel and Jane Huang. The next RE top model. In *Intl. Conference on RE*, 2009.

[14] Ann Hickey and Alan Davis. Elicitation Technique Selection: How do experts do it? In *11th International Conference on RE*, 2003.

[15] James M. Higgins. Innovate or evaporate: Creative techniques for strategists. *Long Range Planning*, 29(3):370 – 380, 1996.

[16] Kang et al. Feature-oriented domain analysis (FODA) feasibility study. Technical Report CMU/SEI-90-TR-21, Software Engineering Institute, Carnegie Mellon University, November 1990.

[17] G. Kotonya and I. Sommerville. *Requirements Engineering*. John Wiley & Sons Ltd, 1998.

[18] L. Lamport. The temporal logic of actions. *ACM Transactions on Programming Languages and Systems (TOPLAS)*, 16(3):872–923, 1994.

[19] M. Mannion and B. Keepence. SMART requirements. *ACM SIGSOFT Software Engineering Notes*, 20(2):42–47, 1995.

[20] Andrew Monk and Steve Howard. Methods & tools: the rich picture: a tool for reasoning about work context. *interactions*, 5(2):21–30, 1998.

[21] Daniel L. Moody, Patrick Heymans, and Raimundas Matulevicius. Improving the effectiveness of visual representations in requirements engineering: An evaluation of i* visual syntax. In *17th Intern. Requirements Engineering Conf.*, 2009.

[22] Bashar Nuseibeh and Steve Easterbrook. Requirements engineering: a roadmap. In *ICSE '00: Proceedings of the Conference on The Future of Software Engineering*, pages 35–46, New York, NY, USA, 2000. ACM Press.

[23] DT Pham and RS Gault. A comparison of rapid prototyping technologies. *International Journal of Machine Tools and Manufacture*, 38(10-11):1257–1287, 1998.

[24] James Robertson and Suzanne Robertson. *Mastering the requirements process*. Addison-Wesley, 2007.

[25] S. Steve Easterbrook. Elicitation techniques. lecture at the University of Toronto, June 2005. slides as PDF available at http://www.cs.toronto.edu/ sme/CSC2106S/slides/04-elicitation-techniques.pdf.

[26] Michael Tovey. Styling and design: intuition and analysis in industrial design. *Design Studies*, 18(1):5 – 31, 1997.

[27] Open University. Visualization techniques. http://systems.open.ac.uk/materials/t552/index.htm.

[28] Roel Wieringa. Requirements researchers: are we really doing research? *Requirements Engineering*, 10:304–306, 2005.