A Model-Driven Approach to Formalize and Support Controlled Experiments in Software Engineering

Marília Aranha Freire[1,2]

[1]Federal University of Rio Grande do Norte, Natal-RN, Brasil
[2]Federal Institute of Rio Grande do Norte, Natal-RN, Brasil
marilia.freire@ppgsc.ufrn.br

ABSTRACT

In this paper, we present the main goals, questions and hypotheses of our research related to provide automated support environments to conduct controlled experiments in software engineering (SE). Here, we introduce a model-driven approach that has been developed for supporting the formalization and execution of SE experiments. It consists of: (i) a domain specific language (DSL) for process specification and statistical design of controlled experiments; (ii) model-driven transformations that promote the generation of workflow models specific to each experiment participant and according to the experiment statistical design; and (iii) a workflow execution environment that allows the monitoring of participant activities in the experiment. In addition, we present the planning of empirical studies for this approach. The research was divided in phases and each phase has empirical studies defined in order to validate it and answer our research questions. Each phase adopts cycles of successive approximations (generate-and-test cycles).

Keywords

Controlled experiments, experimental support environment, model-driven development, domain specific language.

1. INTRODUCTION TO THE RESEARCH

All knowledge areas need to transfer new scientific theories from academy to industry. The successful transfer of knowledge of a technology usually takes around 18 years [1] [2]. However, systematic studies allied to perform experiments may accelerate this process and thus reduce the time to obtain the expected gains by scientific innovations produced by the academy in the context of real projects being developed by industry. The software engineering community have been requiring the conduction of empirical studies in order to make software engineering a science more exact. In this scenario, hundreds of controlled experiments have been performed in the software engineering area over the last years [3]. A controlled experiment is a technique that allows scientists to test a research hypothesis and the cause and effect relationship between variables (dependent and independent) involved in the study environment.

However, despite all this recognized importance, there is still no adequate computational support for the design, analysis, execution and replication of controlled experiments in software engineering, especially with regard to large-scale studies. We believe that an approach to support the formalization and monitoring of a controlled experiment can help the researchers to conduct and replicate controlled experiments in SE.

IDoESE '13, Baltimore, Maryland, USA

1.1 Background

The process of a controlled experiment is typically composed of the following phases: definition, planning, execution, analysis, and packaging [4]. Each one of these phases is associated with the execution of different activities that consume and produce artifacts related to the study in question. However, the planning, execution, analyzing and packaging of a controlled experiment as well as its replication are currently still complex, costly and error prone activities. One reason for this is the lack of computerized environments to support the modeling and execution of such experimental studies. These environments can help in supporting and monitoring the activities involved in an experiment, giving more systematic execution, and facilitating its conduction in large scale.

The recent practical experience in the conduction of controlled experiments in software engineering highlights recurring problems that are mentioned in the literature [5] [6] [7] [8], many of them are a consequence of: (i) many experiments employ "paper & pencil" to collect data on the response variables; (ii) many experiments use and adapt existing tools to run and manage experiments; or (iii) researchers develop new specific tools to assist the implementation of the experiment in question.

All these strategies present problems and make difficulties to the efficient execution and replication of controlled experiments. The use of manual data collection (paper & pencil) during the execution of an experiment is well discussed in the literature [9] [5] [7] [8], and its problems are clear: it is an error prone activity, which can cause inaccuracy in the collected data, and it also requires much more time for experimental data collection and analysis.

The use of existing tools adapted to the context of experimentation can assist the execution of the experiment, but it also creates problems. Firstly, the need to adapt experimental procedures according to the tool procedures (activities and terminology). Secondly, unnecessary features from existing tools can confuse the experiment participants. Finally, it also requires more detailed training on the utilization of the tool, consuming more time in the experiment execution.

The development of a new tool to support the modeling and execution of the experiment was also adopted in [10]. This alternative allows the development of an environment to support the execution of an experiment according to the its needs. On the other hand, this is a very time consuming, it is usually specific to the domain of the problem being treated, and in many cases it is not a reusable solution.

Besides that, there is the problem of between-researcher information exchange. Therefore, in order to replicate an experiment the relevant information (context, raw data, etc.) is not readily available.

1.2 Problem Statement

The above weaknesses of the experimental process have evidenced the problem that our research will address. In a general way, it is necessary an approach to support the formalization and execution (monitoring, statistical analysis, packaging, etc.) of controlled experiment in SE. Although the software engineering (SE) community has proposed approaches over the last years, they have a lot of limitations and potential for future improvements, as presented in the systematic literature review [11] that we have conducted regarding existing environments for supporting experimental software engineering.

To overcome these challenges, this paper proposes a model-driven approach to develop support environments for formalizing and conducting controlled experiments in SE. More specifically, the problems to be solved are:

- There is not any common terminology to describe controlled experiments in SE;
- There are no automated experimental support tools for running and monitoring experiments, and their replications; and for guiding execution according to the statistical experimental design used.

2. RELATED WORK

Although the number of controlled experiments in software engineering has increased in recent years, there is still very limited number of proposed solutions to support conducting of this kind of empirical study [11].

Travassos et al. [7] present the experimental eSEE (experimentation environment to support large-scale experimentation) environment. It provides support to the management of various kinds of empirical studies in software engineering. Its conceptual model is composed of three levels of knowledge organization about the experimental process: (i) meta level – deals with the common knowledge to any experimental study; (ii) configuration level – specific knowledge for each type of experimental study; and (iii) instance level – deals with the knowledge to one specific experimental study. The environment has a prototype and an initial set of tools to populate the infrastructure being built. The work is restricted to detailing the proposed approach, it explores a brief comparison with other approaches and presents no empirical study to evaluate the proposal.

Sjøberg et al. [6] present SESE (Simula Experiment Support Environment), a web-based environment that supports the management of participants in an experiment, captures the time spent during the experiment, enables the collection of artifacts, and monitors the activities of participants. The participant is guided through the activities of a software process and during this execution the artifacts are collected. Activities related to the experiment are performed as part of the routine activities of the developer. This process cannot be tailored according to the experiment being run. The tool is compared with some web tools to support surveys. An empirical study to drive the replication of a controlled experiment using the tool is presented in [9].

Hochstein et al. [5] describe a framework as a set of integrated tools to support experiments in software engineering, the Experiment Manager Framework. The framework was applied to experiments in high-performance computing (HPC). The environment guides the software development steps and applies heuristics to infer the developers' activities. The study presents a brief comparison with other existing approaches and cites the conducting of several experiments on HPC in various universities where some of them were carried out using the framework and others were made before the development of the tool allowing the comparison of results and allowing the analysis of the tool use.

Although these solutions have brought support for many activities involved in a controlled experiment, it may have limitations and potential for future improvements. In general, they do not meet some among the following features: (i) complete experiment definition; (ii) extensibility and adaptability according to the needs of the experiment being carried out; (iii) automatic organization of the experiment according to the statistical design; (iv) support for statistical analysis; and (v) monitoring with online execution and history.

3. RESEARCH OBJECTIVES

The general objective of this work is to propose an approach that allows the generation of automated environments that facilitate the formalization and conduction of controlled experiments in software engineering. In particular, it is explored and investigated the use of techniques of model-driven engineering and domain-specific languages to meet this goal.

The project has the following specific goals:

G1: To investigate the state of the art for automatic or semi-automatic environments for the definition, execution and analysis of experiments in software engineering [11];

G2: To define a set of guidelines for the design and implementation of an infrastructure to support software engineering experiments;

G3: To adopt model-driven technology to transform the modeling of experiments in executable workflows [12] [13];

G4: To model different experiments, using the proposed infrastructure to assess their effectiveness [14];

G5: To analyze qualitatively and quantitatively the proposed approach in the conduction of controlled experiments, highlighting its advantages and limitations.

The development of this work is guided by three research questions, which are described below. In order to answer these questions will be elaborated studies that generate evidence to enable the formulation of responses, even if linked to specific contexts.

RQ1: What tools have been proposed to support the conduction of controlled experiments in software engineering? What are the benefits and limitations of these tools?

RQ2: What is the feasibility of using DSL (domain specific languages) and MDD (model-driven development) in the development of support environments for conducting controlled experiments in SE?

RQ3: What are the benefits and limitations of the approach?

In addition, we derived RQ3 into more concrete research questions (RQ3.1 to RQ3.4) in order to answer them in an easier manner during our empirical studies. In particular, we intend to conduct studies for the evaluation of the benefits of the proposed environment, in relation to:

RQ3.1: Is the approach DSL expressive enough to specify a variety of controlled experiments?

RQ3.2: Does the approach facilitate the formalization and information communication among researchers?

RQ3.3: Is it easy to conduct/replicate an experiment using the proposed approach?

RQ3.4: Does the approach reduce the difficulty to use statistical techniques for designing and analyzing controlled experiments?

Based on these main research questions, we formulate our research hypotheses as presented next:

RH1: There are few tools to support the conduction of controlled experiments in software engineering**.**

RH2: It is the feasible to use DSL and MDD in the development of support environments for conducting controlled experiments in SE.

RH3: The proposed approach provides a completeness and conciseness formalization of an experiment.

RH4: The approach facilitates the formalization and information communication among researchers.

RH5: The use of approach eases the experiment conduction and replication.

RH6: The approach reduce the difficulty to use statistical techniques for designing and analyzing controlled experiments.

4. PROBLEM-SOLVING APPROACH

In order to fill the existent gap evidenced by the practice state and the systematic literature review done this work proposes a model-driven approach to generate support environments for formalizing and conducting controlled experiments in SE. The proposed approach [14] consists of: (i) a domain-specific language (DSL - Domain Specific Language) to model the different perspectives of a controlled experiment; (ii) model-driven transformations that maps abstractions of the experiment DSL for workflow models for each participant in the experiment; and (iii) a workflow execution environment that guide the experiments execution and allows their monitoring.

The controlled experiment conduction is a complex process composed of many phases and activities (with their respective artefacts and roles) and its replication is even more complex and generate a large volume of information that is hard to manage. Therefore, the fact of there is not a pattern way to formalize an experiment is a problem that complicates the experiment replication because in a collaborative experimental research it is needed to transfer knowledge between the involved researchers in a common terminology. In this context, to provide a domain-specific language (DSL) to formalize an experiment and its replicas is the first objective of the proposed approach. It works as a controlled vocabulary and should facilitate information communication and exchange among experimenters, contributing to meet the gap related to provide a complete experiment definition.

Another important point is that during the planning phase of the experiment, the researcher has to know how to deal with statistical skills needed to organize and analyze a controlled experiment. This is intrinsically linked with the choice of the statistical design of experiment, since the kind of design constrains the analysis that can be performed, and therefore the conclusions that can be drawn [15]. Although there are simple statistical experimental designs, it is not easy to a software engineer researcher that is unfamiliar with statistic to know how to set it up. This highlights the problem of abstraction mismatch and the need for a support environment that does not require additional statistical expertise. In the context of large-scale experiments, this task is even more complex and difficult. For this, based on the chosen statistical design and the specification of the factor under investigation and experiment controlled variables, the proposed approach generates customized procedures to be executed in a workflow for each different subject according to the selected experimental design. In that way, we believe that our approach contributes to help the researchers in the application of statistical techniques by releasing them of the manual organization of the many combinations of alternatives unitary experiments that they have to deal with, contributing to meet the gap related to provide automatic organization of the experiment according to the statistical design.

Besides that, to manage an experiment execution is a hard task. In the context of large-scale experiments, this task is even more complex. The care in which such data is reported and collected greatly affects the accuracy of the process. Consequently, the gathering of poor self-reported data can distort the results [5] and invalidate the experiment. In order to deal with this challenge, the proposed approach enables researchers to monitor the activities of the subjects and the data produced during the experiment execution. It contributes to the experiment control because the researchers can get early information about the data accuracy and can detect misunderstandings by the participants. Each participant does not need to explicitly register and measure the time spent for each activity, for example. We believe that this can directly contribute to minimize the overhead of collecting such data and increasing the accuracy of the collected data, contributing to meet the gap related to provide monitoring with online execution and history. It is performed by the generated workflows that provide a step-by-step customized procedure for each different subject, helping the registering and storing of his/her activities. The use of model-driven technology contributes to meet the gap related to provide extensibility and adaptability according to the needs of the experiment being carried out;

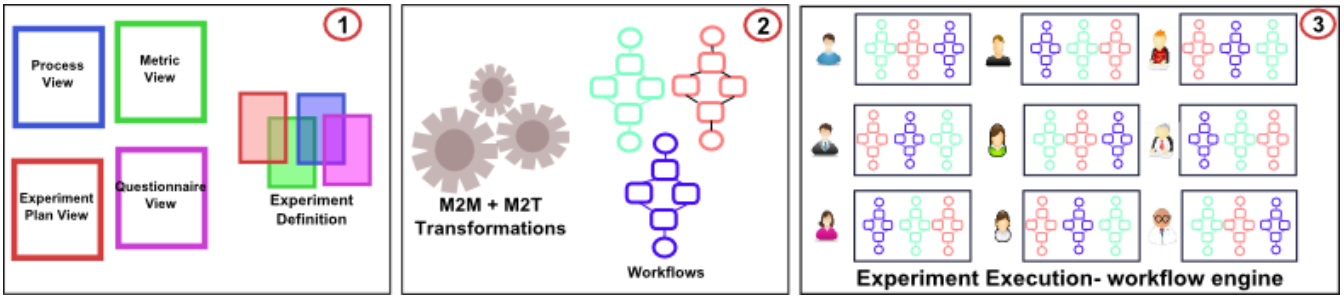

Figure 1: Approach Overview

The Fig. 1 illustrates an approach overview. **Approach first stage - experiment definition and planning**: For this stage we have proposed a domain specific language that allows specifying different perspectives from an experiment [14]. The development of software systems using domain specific languages (DSLs) has increased over the last years. Our experimental DSL is called ExpDSL. It is a textual DSL developed using the xText[1]. ExpDSL is composed of four perspectives, as we can see in the box one from Fig. 1: (i) **the Process View** – allows defining the procedures of data collection from the experiment participants. It is similar to a software process language; (ii) **the Metric View** – is used to define the metrics that have to be collected during the experiment execution. The metrics are responsible to quantify observations of interest (dependent variables, etc.) in the study; (iii) **the Experimental Plan View** – is used to define the experimental plan. It allows setting the factors to be controlled (treatments, noisy variables, etc.), as well as a statistical design of experiment (DoE) used to organize treatment allocation and the experiment execution; and (iv) **the Questionnaire View** – allows defining questionnaires in order to collect quantitative and qualitative data from participants of the experiment. The questionnaires can be applied before or after all the experiment activities, or before or after procedures specified in the process view.

Fig. 2 shows a ExpDSL modeling fragment from the experimental plan view related to an experiment that compare two black-box testing techniques to software product lines [16]. It define the factors and treatments, the parameters of the experiment, and the statistical design chosen for the controlled experiment from a list of three possibilities: (i) completely randomized design - CRD, (ii) randomized complete block design - RCBD, and (iii) Latin square - LS.

The *Parameter* element is used to specify any characteristic (qualitative or quantitative) of the experiment context that has to be invariable throughout the experimentation. In this case, we have the specification of parameters such as "Participants have to pause time for asking questions". The *Factor* "TestTechnique" is the element that represents the investigated techniques, therefore the attribute "isDesiredVariation" is "True". There are two treatments (levels or alternatives) specified for this factor, which are: (i) the *Level* "Generic" (Generic Technique); and (ii) the *Level* "Specific" (Specific Technique). The second Factor is

```
Experimental Plan Design "TestDesign"
    type LS - Latin Square  {
        Parameter "Participants have to pause time for asking questions"
        Parameter "The LPS complexity is low"
        Parameter "Artefacts are written in the participant source language"
        Parameter "Participants are undergraduating students"

        Factor "TestTechnique" isDesiredVariation True
            Level "Generic";
            Level "Specific";
            ;
        Factor "Feature" isDesiredVariation False
            Level "F1";
            Level "F2";
            ;
         Factor "Subject" isDesiredVariation False
            Level "Subject1";
            Level "Subject2";
            ;

         Internal Replication 4
```

Figure 2: Fragment from a experiment modeling

"Feature" and the levels are named "F1" and "F2", representing each feature to be used in the experiment. The last factor specified is "Subject" with levels "Subject1" and "Subject2" (in our 2x2 Latin square, two subjects are required per replica of the square). Finally, the experimental design selected from the list of designs is the "LS – Latin Square". The "Internal Replication" element that is used to indicate the number of applications of each experiment treatment. This parameter has a different meaning according to the experimental design chosen. In a Latin square, it represents the number of square replicas. For the experiment showed, we indicate four internal replicas, which means to have eight subjects in the experiment.

Approach second stage - the workflow generation: The second stage of our approach is the model-driven transformations that receive as input the experiment definition specified in the ExpDSL (our textual DSL) and generates as output customized workflows for each experiment participant according to the specified statistical design. The workflows are responsible for guiding each participant through the activities and tasks needed to conduct the experiment and to extract data related to the specified metrics. Fig. 1, box 2 illustrates this approach stage.

The proposed approach defines two transformations to be executed in this stage: (i) a M2M (model-to-model)

Table 1: Studies relating to research question, phase and metrics

Research Phase	Goal	Research Question	Hypothesis	Study	Criteria/Metrics
Problem Statement	G1	RQ1	RH1	Systematic Literature Review (ES1)	Experimental process phase supported, main features, environment evaluation type
Approach Definition and Approach Implementation	G2, G3	RQ2	RH2	*Experiment Modeling* (ES2) and Exploratory Study (ES3)	Expressiveness
Approach Evaluation	G4	RQ3.1	RH3	*Experiment Modeling (ES4)*	completeness and conciseness
	G5	RQ3.2	RH4	*Survey (ES5)*	Comprehensibility
		RQ3.3	RH5	*Case Studies by the research community (ES7)*	Easy of use
		RQ3.4	RH6	*Case Studies by graduate students (ES6)*	Questionnaires, Comparative analyses

[1] h

transformation that receives as input an Ecore model (generated from the ExpDSL specification) and generates as output the workflow model according to the defined processes related to the experiment factors and treatments, and the statistical design of experiment chosen by the researcher during the planning. The design of experiment is important to organize the participants and the experiment execution arrangement; and (ii) a M2T (model to text) transformation responsible for generating the web forms code and configuration files needed to execute the workflows of the experiment in a web engine.

The M2M transformation is implemented using QVTo (Operational QVT) language and the M2T transformation using the Acceleo template language. The workflow metamodel is based on the jBPM Process Definition Language (JPDL) schema, but it has more elements than the original one, which are specific to the experimental software engineering domain.

Approach third stage – experiment execution: The third and last stage of our approach is the experiment execution in our web workflow engine (Fig. 1, box 3). It is performed as a web application in a workflow engine where each participant can follow the instructions for her/his tasks, and the researchers running the experiment can monitor these activities. Our research group has developed a workflow engine in order to support this approach stage.

In order to execute an experiment, the researcher needs to upload the generated files (workflow specification, web forms and configuration files), which represents the whole experiment, in the web workflow engine environment. Then, they have to perform as the register of participants and upload the input artifacts specified for the processes tasks. Finally, they can enable the experiment execution. From that point on, each participant can start the experiment execution following the activities described in his/her workflow. During the execution, the metrics will be collected according to the specification (we are initially collecting metrics related only to the time spent by the experiment) and the collected time will be available for the experiment analysis phase.

In addition, during the experiment execution, the researchers can monitor all the activities in progress as well as to see details about the executed activities for each experiment participant, such as the uploaded artifacts, time spent for different tasks and activities, and comments added. Thus, the researcher can get early information about the data accuracy and can detect misunderstanding by the participants. Besides that, each web workflow form presents a pause button that must be used by the participant if she/he needs to shortly interrupt the current experiment activity to do some action that can affect the quantified time for that activity. This interruption can be to perform any extra activity (such as, to answer the cell phone or to go to the restroom) that can affect the activity time resulting, according to the experiment rules

5. RESEARCH METHOD

We plan to run studies in order to answer the research questions. We have planned at least one study for each research question. The objective of this section is to describe these studies. The research has been divided into phases. Each phase tries to generate results to answer the research questions. The research phases are:

1 – Problem Statement

2 – Approach Definition

3 – Approach Implementation

4 – Approach Evaluation

Within each phase, we will adopt cycles of successive approximations (generate-and-test cycles) where in each approximation news evaluation studies are executed in order to identify news features or improvements points, resulting in a more realistic approach. The Fig. 3 shows the research phases and related empirical studies and the Table 1 shows the studies relating to research question, phase and metrics

5.1 Description of the empirical studies

Phase 1 – Problem Statement: The objective of this stage was to study the current situation to gather in-depth knowledge of the problem in question. The product of this first stage was an understanding of the current situation, expressed as both the state of the art and the state of practice. To do this, we carried out the following study:

Empirical Study 1 (ES1) – Systematic Literature Review: We have performed a systematic literature review [17] [18]. It aimed to bring an overview of how the issue is being addressed by the academic community, and confront these results with the practical experience of our research and partners groups in conducting controlled experiments. Preliminary results obtained allowed the identification of deficiencies in environments to support the formalization and execution of controlled experiments in ES. Freire et al [11] present the results of this SLR that aims to answer the research question 1 (RQ 1).

Phase 2 – Approach Definition: The objective of this phase was to define a model-driven approach to specify one experiment and that allows the experiment execution as a workflow. This phase comprises: (i) the specification of domain specific language (ExpDSL) for the modeling and execution of controlled experiments; and (ii) the workflow model definition that support the execution of the participants' activities

Empirical Study 2 (ES2) – Experiment Modelling: The model was built and evaluated iteratively by modeling several experiments from the literature e from our research group and partners groups . Such approach was designed based on results of preliminary research to define strategies for monitoring software processes [12] [13]. (RQ2).

Phase 3 – Approach Implementation: The goal behind this phase was to build a technology tool to formalize and execute (monitoring) controlled experiments in SE. The approach was implemented in to support the generation of specialized workflows for the execution and monitoring of different participants' activities from the experiment. The workflows execution and treatments distribution is done according to the statistical design of experiment (DoE).

Empirical Study 3 (ES3) – Exploratory Study: An exploratory study was performed to evaluate the feasibility and expressiveness of the proposed approach. In this study, a real and complex controlled experiment was modeled to evaluate the expressiveness of the different elements of the approach, which include the ExpDSL, the generated workflows, and the runtime environment responsible for the workflows execution and monitoring. During the exploratory study, some limitations were identified and remedied by proposing improvements for each approach element.

Freire et al [14] present the results of this exploratory study that aims to answer the research question 2 (RQ 2). Two other studies have also used the DSL approach in this proposal as a case study [19] [20].

Phase 4 – Approach Evaluation: In this phase, we go ahead in the approach evaluation and improvements. This phase is planned to answer the third research question (RQ3): *What are the benefits and limitations of the approach?* Specifically we want to answer the derived questions*: Is the approach DSL expressive enough to specify a variety of controlled experiments*? (RQ3.1), *Does the approach facilitate the formalization and information communication among researchers?* (RQ3.2), *Is it easy to conduct/replicate an experiment using the proposed approach*? (RQ3.3) and, *Does the approach reduce the difficulty to use statistical techniques for designing and analyzing controlled experiments*? (RQ3.4)

Empirical Study 4 (ES4) – Experiment Modeling: we have been modelling different controlled experiments in SE to assess the experiment formalization quality. The aim is to evaluate if the DSL is actually effective in its intended domain, that is, if experiments in software engineering are expressible with the DSL. This can be defined as the *completeness* of the DSL or the *coverage* of the domain. We also want to evaluate if all the information gathered in the DSL is useful. Therefore, we intend to evaluate the ExpDSL completeness and conciseness. (RQ3.1)

Empirical Study 5 (ES5) – Survey: the purpose of this survey is to identify characteristics and weakness from the ExpDSL in the perspective of the expert researchers of empirical software engineering area. We will apply a questionnaire to approximately 20 experimenters of different test experimentation teams. We intend to make a video of no more than 5 minutes, showing the main environment features. The survey participants will be asked if they are interested in being contacted to use the environment, thereby fueling the study ES7. (RQ3.2)

Empirical Study 6 (ES6) –Studies by graduate students: we intend to calibrate and use the approach for 3 months, during the conduction of controlled experiments in graduate courses. The main objective here is to verify if our proposed approach can reduce the difficulty to plan and run designed experiments. All the students will use the approach infrastructure and the professor will compare the students' performance in relation to previous classes that have used the manual approach (RQ3.4).

Empirical Study 7 (ES7) – Case Studies by the research community: we will make the tool available for use by other research groups. The invitation will be sent for groups from different countries, based on their historical use of controlled experiments, including replications. After the usage of the tool, researchers will have to answer questions about the approach benefits and limitations. (RQ3.3).

5.2 Definition of metrics

We define here the set of metrics to be collected during the empirical studies to answer our research questions in a quantitative and qualitative way.

Completeness – it refers to the extension, degree, amount or coverage to which the information in a domain specific language covers the information of the real world. The completeness of a definition depends on the level of granularity agreed to in the whole DSL. We can say that a definition is complete if different issues involved in the definition of the experiment can be modeled using our DSL. Therefore, we want to identify the amount, types and complete list of issues that are not possible to model with the DSL.

Conciseness – it refers to if all the information gathered in the DSL is useful and accurate. Conciseness does not imply absence of redundancies. Sometimes, some degree of controlled redundancy can be useful in the definitions. We want to identify the quantity, types and complete list of DSL elements that are not considered useful or accurate.

Comprehensibility – it refers to the researchers understanding perception using the proposed approach. We want to identify the quantity, types and complete list of misunderstanding in the experiment modeling using the DSL.

Easy of use – it refers to the users' opinion about the facilities of using the approach. They will answer question choosing their perception according the rating scale: strongly agree, agree, do not know, disagree, or strongly disagree. We want to evaluate the user perception in relation to the learning curve, need of training or statistics knowledge, DSL and approach usability, etc.

Table 2 shows the GQM definition related to the metric definition for the proposed research.

Table 2: GQM definition

Goal	Question	Metrics related to
Evaluate the approach quality in order to model an experiment from the perspective of a experimenter	Is the approach DSL expressive enough to specify a variety of controlled experiments?	Completeness Conciseness
Evaluate the approach comprehensibility from the perspective of the researchers	Does the approach facilitate the formalization and information communication among researchers?	Comprehensibility
Evaluate the approach easy of use from the perspective of the researchers	Is ease to conduct/replicate an experiment using the proposed approach?	Easy of use

5.3 Limitations and threats to validity

In this section, we list the main threats to the validity of our empirical studies and how we plan to control them.

Selected Metrics – In order to evaluate the approach quality we need to specify quality attributes to be measured**.** Most quality attributes are subjective and cannot purely be determined based on objective metrics. To reduce possible bias, we consider the opinion of researchers from different groups. We also do not measure effort for using the approach in our studies. This is because we believe that the time spent for planning real experiments are interactive and long (several days, weeks or even months), taking much more time to elaborate an appropriate design rather than formalizing it in the proposed tool. Hence, our empirical studies focus on evaluating the approach based on the quality benefits and user satisfaction.

External validity – The results from the studies that we have executed or planned to execute are based on different experiments in SE area, making generalizations difficulty. However, it is disputable if these results can be generalized to modeling/execute any experiment in SE. Most part of the applied experiments were selected due to its available documentation, making it representative to a large class of experiments in SE, but also due the experimenters available to contribute with us given information and knowledge about the experiments. In order to draw global conclusions, a wider variety of experiments needs to be modeled and analyzed. We also intend to contact active experimentation researchers in order to enlarge the approach feedback.

6. SUMMARY OF THE CURRENT STATUS OF THE RESEARCH AND PLANNED STEPS

6.1 Current Status of the Research

At the time this paper was written, we have defined the problem and performed our systematic literature review. With respect to the second phase, we have defined and reviewed the domain specific language through the modeling of several experiments in SE. We have obtained a conceptual model based on study - elicit - test - modify cycles. We have defined four different views that comprise an experiment: process view, metric view, experimental plan view, and questionnaire view.

With respect to the third phase, we implemented our model-driven approach composed of a textual DSL (ExpDSL), a workflow model, model-to-model and model-to-text transformations, and a workflow engine as a web-application. The results of this phase have demonstrated that our approach is feasible because the performed preliminary studies have confirmed the usefulness and practicality of the approach by allowing the adequate modeling of the selected studies and the generation of the experiment workflow, that supports its execution. The study also identified improvement points related to the DSL and workflow engine of the approach to model different perspectives during the conduction of an experiment.

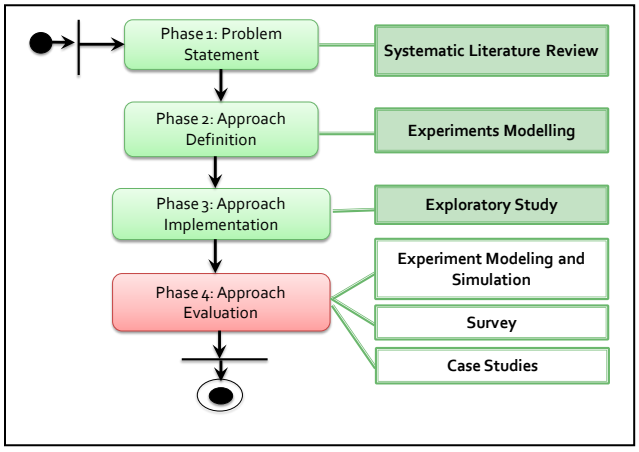

Figure 3: Research current status

As part of the fourth stage, we are now analyzing our approach (expressiveness, completeness, conciseness, effort) in a qualitative and quantitative way through the modeling of other real experiments. We also intend to perform a survey with expert researchers and, finally, we intend to execute case studies to observe our approach in action in order to identify benefits and limitations.

Fig. 3 summarizes the current status of our research. We are in the fourth phase, defining and executing the planned empirical studies.

6.2 Planned Steps

We plan in the immediate future to carry out several activities in parallel in compliance with the planned empirical studies. To be more precise, the tasks to be executed are:

- To continue with the modeling of several experiments evaluating the approach completeness and conciseness;
- To analyze the quality of the approach from the expert researcher perspective;
- To perform case studies to observe the approach in action and collect quantitative and qualitative results that give evidence of its effectiveness;
- To make the approach available to allow its usage in the conduction of experiments for partner research groups and collect feedback about it.

Table 3: Empirical Studies Schedule

Empirical Study/RQ	Apr – Aug 2013	Sep – Dec 2013	Jan–Jul 2014
ES4/RQ3.1			
ES5/RQ3.2			
ES6/RQ3.4			
ES7/RQ3.3			

Therefore, we are focusing in 2013 and 2014 on the empirical studies developing according to the Table 3. Finally, we will put together the final research reports in 2014.

7. CONCLUSIONS

In this paper, we presented the main goals, questions and hypotheses of our research. We introduced a new model-driven approach to support the conduction of controlled experiments in software engineering. We presented the planning of the empirical studies for evaluating this approach.

These studies are based on a systematic literature review, a survey and case studies. We also presented the metrics/criteria that will be used during the empirical studies to evaluate the approach, and how they will help to answer our research questions. Finally, we have presented the current status of our research and the planned next steps.

We believe that the proposed approach cannot only support researchers who are currently working with controlled experiments, as may facilitate their dissemination, replication process, meta-analysis, etc.

8. ACKNOWLEDGMENTS

I thank my advisors Uirá Kulesza and Eduardo Aranha for their helpful support during my research. This study is supported by the program "Ciência sem Fronteiras", a joint initiative from the Ministérios da Ciência, Tecnologia e Inovação (MCTI) and the Ministério da Educação (MEC), through CNPq and Capes. It is partially supported by the National Institute of Science and Technology for Software Engineering (INES), funded by CNPq,

grants 573964/2008-4 and PDI – Great Challenges, 560256/2010-8, and by FAPERN and CAPES/PROAP.

REFERENCES

[1] A. Jedlitschka, M. Ciolkowski, C. Denger, B. Freimut e A. Schlichting, "Relevant Information Sources for Successful Technology Transfer: A Survey Using Inspections as an Example.," em *Empirical Software Engineering and Measurement*, Germany, 2007.

[2] S. Redwine e W. Riddle, "Software Technology Maturation," em *8th International Conference on Software Engineering*, 1985.

[3] D. Sjoeberg, J. Hannay, O. Hansen, V. Kampenes, A. Karahasanovic, N.-K. Liborg e . A. Rekdal, "A survey of controlled experiments in software engineering," *IEEE Transactions on Software Engineering*, vol. 31, pp. 733,753, 2005.

[4] C. Wohlin, P. Runeson, M. Höst, M. C. Ohlsson, B. Regnell e A. Wesslén, Experimentation in Software Engineering: An Introduction, Boston/Dordrecht/London: Kluwer Academic Publishers, 2000.

[5] L. Hochstein, T. Nakamura, F. Shull, N. Zazworka, V. R. Basili e M. V. Zelkowitz, "An Environment for Conducting Families of Software Engineering Experiments," *Advances in Computers*, vol. 74, p. 175–200, 2008.

[6] D. I. Sjøberg, B. Anda, E. Arisholm, T. Dybå, M. Jørgensen, A. Karahasanovic, E. F. Koren e M. Vokác, "Conducting Realistic Experiments in Software Engineering," *International Symposium on Empirical Software Engineering* , 2002.

[7] G. H. Travassos, P. S. M. Santos, P. G. Mian, A. C. Dias Neto e J. Biolchini, "An environment to support large scale experimentation in software engineering," *13th IEEE International Conference on Engineering of Complex Computer Systems*, pp. 193-202, 2008.

[8] K. T. Stolee e S. Elbaum, "Exploring the use of crowdsourcing to support empirical studies in software engineering," *2010 ACM-IEEE International Symposium on Empirical Software Engineering and Measurement*, 2010.

[9] E. Arisholm, D. I. Sjøberg, G. J. Carelius e Y. Lindsjørn, "SESE – an Experiment Support Environment for Evaluating Software Engineering Technologies," *Nordic Workshop on Programming and Software Development Tools and Techniques*, pp. 81-98, 2002.

[10] P. Accioly, "Comparing Different Testing Strategies for Software Product Lines (Masters Thesis)," Federal University of Pernambuco, Recife, Brasil, 2012.

[11] M. Freire, D. Alencar, E. Campos, T. Medeiros, E. Aranha e U. Kulesza, "Automated Support for Controlled Experiments in Software Engineering: A Systematic Review International *Conference on Software Engineering and Knowledge Engineering*, Boston/USA, 2013.

[12] M. Freire, F. Aleixo, K. Uira, E. Aranha e R. Coelho, "Automatic Deployment and Monitoring of Software Processes: A Model-Driven Approach", International *Conference on Software Engineering and Knowledge Engineering)*, Miami/Florida, 2011.

[13] M. Freire, D. Alencar, E. Aranha e U. Kulesza, "Software Process Monitoring using Statistical Process Control Integrated in Workflow Systems," em *Conference in Software Engineering Knowledge Engineering*, San Francisco/California - USA, 2012.

[14] M. Freire, P. Accioly, G. Sizílio, E. Campos, U. Kulesza, E. Aranha e P. Borba, "A Model-Driven Approach to Specifying and Monitoring Controlled Experiments in Software Engineering," em *Product-Focused Software Process Improvement*, vol. 7983, Paphos, Cyprus, LNCS, 2013, pp. 65-79.

[15] S. L. Pfleeger, "Experimental design and analysis in software engineering: Part 2: how to set up and experiment," *ACM SIGSOFT Software Engineering Notes*, vol. 20, pp. 22-26, jan 1995.

[16] P. Accioly, P. Borba e R. Bonifácio, "Comparing Two Black-box Testing Strategies for Software Product Lines," *SBCARS VI - Brazilian Symposium on Components, Architecture and Software Reuse*, 2012.

[17] B. Kitchenham and S. Charters, "Guidelines for performing Systematic Literature Reviews in Software Engineering," 2007.

[18] B. Kitchenham, R. Pretorius, D. Budgen, O. Pearl Brereton, . M. Turner, M. Niazi e S. Linkman, "Systematic literature reviews in software engineering - A tertiary study," *Inf. Softw. Technol.*, pp. 792-805, August 2010.

[19] 9. E. Campos, A. Bezerra, M. Freire, U. Kulesza e E. Aranha, "Composição de Linguagens de Modelagem Específicas de Domínio: Um Estudo Exploratório," *III Brazilian Workshop on Model-Driven Software Development, CBSoft 2012*, pp. p. 41-48, 2012.

[20] E. Campos, M. Freire, U. Kulesza, A. Bezerra e E. Aranha, "Composition of Domain Specific Modeling Languages: An Exploratory Study," *International Conference on Model-Driven Engineering and Software Development (MODELSWARD 2013)*, 2013.