Towards a Computerized Infrastructure for Managing Experimental Software Engineering Knowledge

Paula Gomes Mian, Guilherme Horta Travassos, Ana Regina Cavalcanti da Rocha and Ana Candida Cruz Natali

COPPE / UFRJ – Computer Science Department - Caixa Postal: 68511 - CEP 21945-970 - Rio de Janeiro – Brazil
{pgmian,ght, darocha, anatali}@cos.ufrj.br

Abstract. The growing interest in experimental studies in software engineering and the difficulties found in their execution had led software engineering researchers to look for ways to (semi) automate the experimental process. This paper introduces the concept of experimental Software Engineering Environment (eSEE) – an infrastructure capable of instantiating software engineering environments to manage knowledge about the definition, planning, execution and packaging of experimental studies in software engineering.

1 Introduction

No science can advance without experimentation and measurement [1]. Progress in any discipline involves building models that can be tested, through empirical studies, to check whether the current understanding of the field is correct. [2].

For instance, there is an increasing agreement in the software engineering community that experimental studies are necessary to develop or to improve software development and maintenance processes, methods and tools [3]. The classical method for identifying cause-effect relationships is to conduct controlled experiments where only a few variables can vary. Controlled experiments in software engineering often involve students solving small pen and paper tasks in a classroom setting. A major criticism regarding such experiments is their lack of realism, which may defer technology transfer from the research community to industry. The experiments would be more realistic if it is run on real tasks, on real systems, with target population software professionals' representative of the technology, using their usual development technology in their usual working environment [4].

In this context, one of the great interests in the experimental studies is the increasing viewpoint that Software Engineering must have strong foundations of a scientific engineering discipline, and that the techniques to improve software development and to evolve processes must be available to professionals. However, executing an experimental study, especially in Software Engineering, is hard, time consuming and generates great volume of information and knowledge that are complex to manage [5].

Hence, such studies depend strongly on a computerized infrastructure, especially when we consider science in wide scale. Areas such as nuclear physics and biomedical computer science also depend on an integrated and distributed internet infrastructure to manage the experimentation process [6].

Attempts to provide computerized support to experimental SE were made, such as SESE (Simula Experiment Support Environment) [7]. Despites the relevance of those works, a more substantial infrastructure is not completely defined and available, yet.

Therefore, it is necessary to define and build a computational infrastructure to support the management of knowledge involved in the experimentation process in Software Engineering.

This paper introduces the concept of ***e**xperimental **S**oftware **E**ngineering **E**nvironment* (eSEE) – an infrastructure able to instantiate software engineering environments to support knowledge management throughout the SE experimentation process, including definition, planning, execution and packaging of experimental studies in software engineering.

This paper is organized as follows: in section 2 we discuss works related to this research. Section 3 presents the eSEE's infrastructure, services and tools. An extension of the experimentation process is discussed in section 4. In section 5, we present a classification of eSEE's features. Finally, in section 6 we describe our conclusions and future work.

2 Initial Works Towards eSEE

The Experimental Software Engineering (ESE) team is a research group of the Systems Engineering and Computer Science Program (PESC) at COPPE/UFRJ. Its goals include the improvement of software engineering by applying the scientific method (experimentation) for the construction of new methods and techniques to support software development, and researching of new models and approaches for planning, execution and packaging of SE experimental studies.

The ESE team has used the experimentation model defined by [8] to perform different experimental studies. This work proposed a model for packaging experiments, which defines a document taxonomy to represent the necessary information for experiments execution. It also proposed a process for SE experiments packaging, defining its stages and activities, the products generated by each one of those stages and activities, and the roles and responsibilities played by the involved people throughout the experimentation process. This approach was built in Hyperwave infrastructure (http://www.haup.org) and was already used to package different types of experiments accomplished by the ESE team.

The diversity of the packaged experimental studies made us able to observe that the conventional two-staged SE experiments taxonomy (in vivo/in vitro) was not enough to classify some of them. Therefore, a four-staged experiment taxonomy has been described [9]:

- *In vivo* experiments: such experiments involve people in their own environments. In software engineering, experiments executed in software development organizations throughout the software development process and under real circumstances can be characterized as *in vivo*;

- *In vitro* experiments: such studies are executed in a controlled environment, such as a laboratory or a controlled community. In software engineering, most *in vitro* experiments are executed in universities or among selected groups of a software development organization;
- *In virtuo* experiments: such experiments involve the interaction among participants and a computerized model of reality. In these experiments, the behavior of the environment with which subjects interact is described as a model and represented by a computer program. In software engineering, these studies are usually executed in universities and research laboratories and are characterized by small groups of subjects manipulating simulators;
- *In silico* experiments: these studies are characterized for both the subjects and real world being described as computer models. In this case, the environment is fully composed by numeric models to which no human interaction is allowed. Due to the need of a large amount of knowledge, *in silico* studies are still very rare in software engineering, being limited to areas where subject participation is not an experimental study issue or intelligent agents can replace human subjects. For instance, we can find *in silico* studies applied to software usability experimentation, such as software performance characterization.

All the works performed by the ESE team have reinforced, despite their importance, how hard is to deal with knowledge regarding experimental studies' planning, execution and packaging. Besides, in the field of experimental software engineering, a central challenge is to efficiently share experimentation knowledge between replicators in order to facilitate replication comparability.

To deal with these issues, [10] proposed an adaptation of Nonaka–Takeuchi knowledge sharing model [11] in the context of software engineering experimentation that was called the experimentation knowledge-sharing model (EKSM). The four phases of the original model, as shown in Figure 1, were mapped to share tacit knowledge through socialization among replicators, making tacit knowledge explicit through laboratory packages[1], improving explicit knowledge through lab package evolution, and internalizing experimentation knowledge by using the lab packages.

Socialization involves the discussion of potential experimental study designs and their threats to validity, variables, procedures, data collection methods, and data analysis procedures. It assumes experimenters are using a common terminology and are communicating common cultural themes. Externalization is the recording of notes, tips, experimental design, raw data, analyses, and results. It involves explicit and implicit choices of what should be written down. It is in these implicit choices that a lot of important tacit information is lost. Combination is the decision of what should go into the lab package for other experimenters to use. This process often involves more externalization, such as changes to the process that occurred during the execution of the experiment. Internalization involves the reading, abstraction,

[1] A laboratory package describes an experiment in specific terms and provides materials for replication, highlights opportunities for variation, and builds a context for combining results of different types of experimental treatments.

interpretation, and execution of what was in the lab package. The execution process often opens questions and points to holes in the lab package [10].

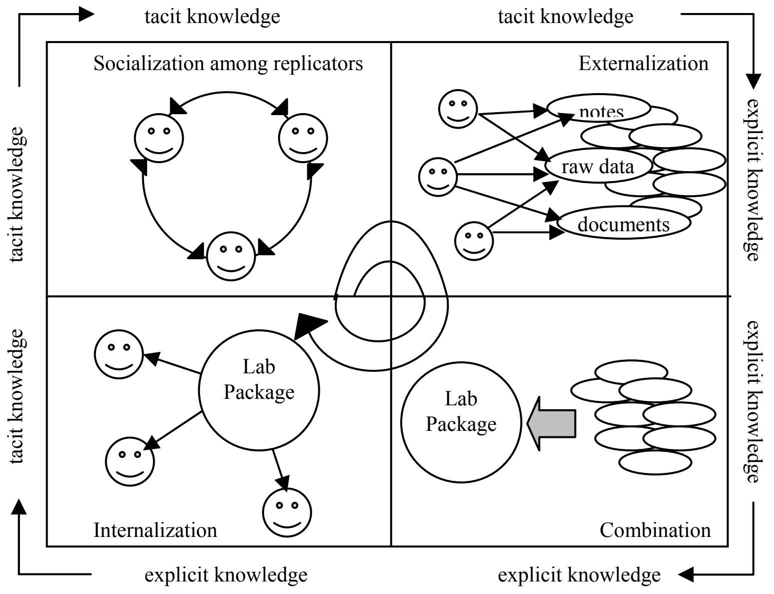

Fig. 1. Experimentation knowledge-sharing model – adapted from [10]

The efforts to support meta-analysis, experiment improvement, knowledge evolution and sharing demand new technological support for managing experiments and knowledge produced by them. As mentioned before, an infrastructure that explores the packaging process approach defined in [8] was built in Hyperwave.

Notwithstanding its relevance, the infrastructure built in Hyperwave doesn't cover all aspects of knowledge management in experimental software engineering, mainly those one concerned with experimental studies execution. Therefore, we identified the need to develop a computerized infrastructure such as eSEE. It must provide an experience base and a standard layer for integrating tools that support experimental studies execution, meta-analysis and knowledge sharing.

To support the building of such infrastructure, the experience and knowledge acquired to build the TABA Project (www.cos.ufrj.br/~taba) are going to be reused. TABA is a meta-environment that allows the instantiation of Software Engineering Environments (SEEs) according to different application domains and technologies. TABA also includes a knowledge dimension that makes possible external CASE tools' integration into the SEE [12].

3 eSEE's Infrastructure

Usually, a lot of clerical activities take part in the whole experimentation process, making it tedious and repetitive mainly for novice experimenters. It led us to look for ways to support some of these SE experimentation process activities by providing automated (or semi-automate) computer based tools integrated into a software engineering environment, highlighting the aim at identifying and defining an infrastructure, such as eSEE, that can allow instantiation of SE environments for experimentation.

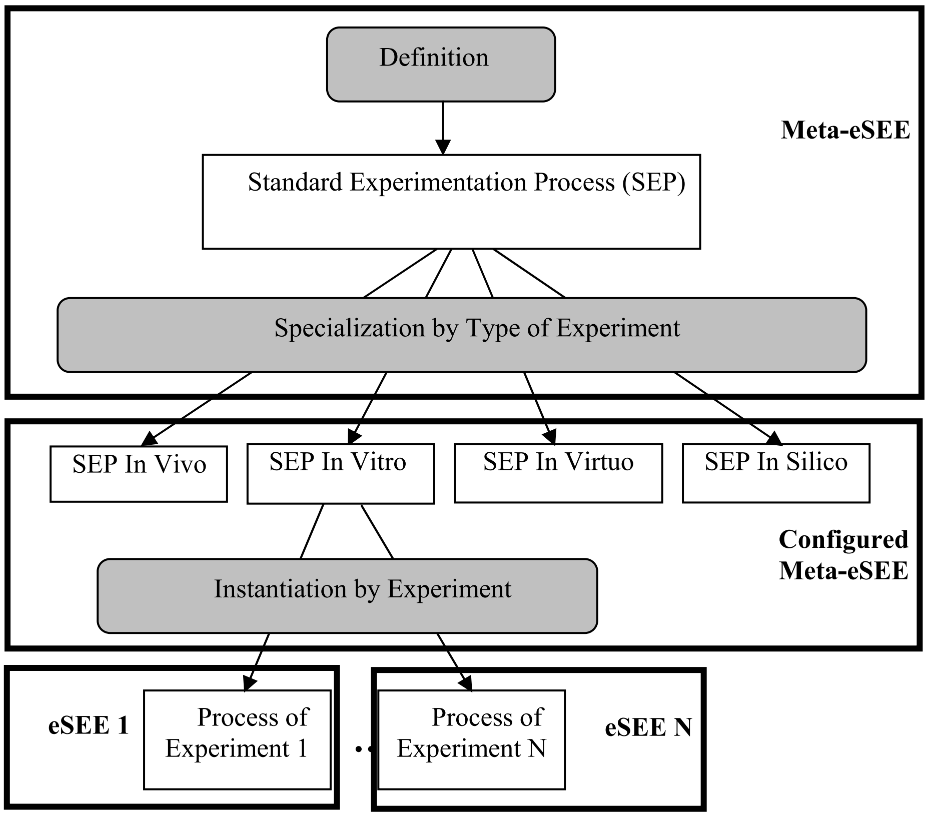

Fig. 2. eSEE's Architecture

An eSEE must manage the software engineering experimentation process, including knowledge acquired when defining, planning, executing and packaging experiments.

eSEE makes available experimentation process models, experiment packages, data representation standards, knowledge management facilities, tools, services, quality and computerized models (simulators) for the identified experiment's types [13].

The proposed computerized infrastructure for eSEE is based on the experimentation process and package models described by [8]. To instantiate an eSEE for a specific study, the experiment type intended to be performed must be defined. Three levels of knowledge organization about the experimentation process have been identified: knowledge for any kind of experiments (meta level), knowledge for each type of experiment (configuration level), and knowledge for a specific experimental study (instance level). These levels reflect themselves at eSEE's architecture, which was inspired by the TABA's metaphor. All the information generated throughout the experimentation process must be packaged for future use by researchers.

The layers composing eSEE's architecture, as shown in Figure 2, are:

- Meta-eSEE: the meta-level contains common knowledge regarding SE experimental studies, including software engineering knowledge. This knowledge may be captured by software engineering ontologies [12] integrated with an experimentation ontology. At this level, a standard experimentation process (SEP) is defined. The SEP represents the basis to instantiate standard processes for each type of experiment;
- Configured Meta-eSEE: the configuration level contains knowledge regarding specific experiments types. At this level, specific eSEE's instantiation can be accomplished by choosing experimental study type and adding specific experimental study's study characteristics. The instantiated environment, then, is generated with the specific knowledge to that study type;
- eSEE: environment supporting for the definition, planning and execution of the specific experimental study type. For each new experimental study type, one eSEE should be instantiated to manage the experimentation process and knowledge.

3.1 eSEE's Tools and Services

Once the instantiated environments are created, some facilities must be available to support the accomplishing of experimental studies. For instance, to allow process tracking in each eSEE, a XML based process execution machine [14] will be integrated to the environment. Nevertheless, experimental studies should be packed into experiment packages repositories throughout the experimentation process [8].

Some of the data necessary to perform experimental studies in SE may come from CASE tools. Therefore, these tools must be extended to collect experimental data [15]. Once CASE tools collect the data, data tool integration facilities based on XML and ontologies will allow data importing to eSEE's repository.

Besides supporting experimental studies accomplishment, other services and tools (shown in Figure 3) are necessary to support eSEE activities, such as:

- Knowledge Management: the eSEE infrastructure must make available knowledge about the experimentation process, methods, techniques and tools to assist the software engineering researcher managing the experimental study. Besides, it allows lessons learned to be stored and reused for future experimental studies` planning activities (to prevent errors and to identify opportunities of experimentation process improvement);

- eCASE Tools: supporting experimentation life cycle activities, such as definition of questionnaires, assignment of participants, data collecting or data analysis;
- Data Visualization Tools: must be enclosed to the experimental environment to easy data analysis;
- Environment, Objects and Subjects Computerized Models: to perform *in virtuo* or *in silico* experiments, it is necessary to build object, participants and environments models (simulators) used in the experimental study.

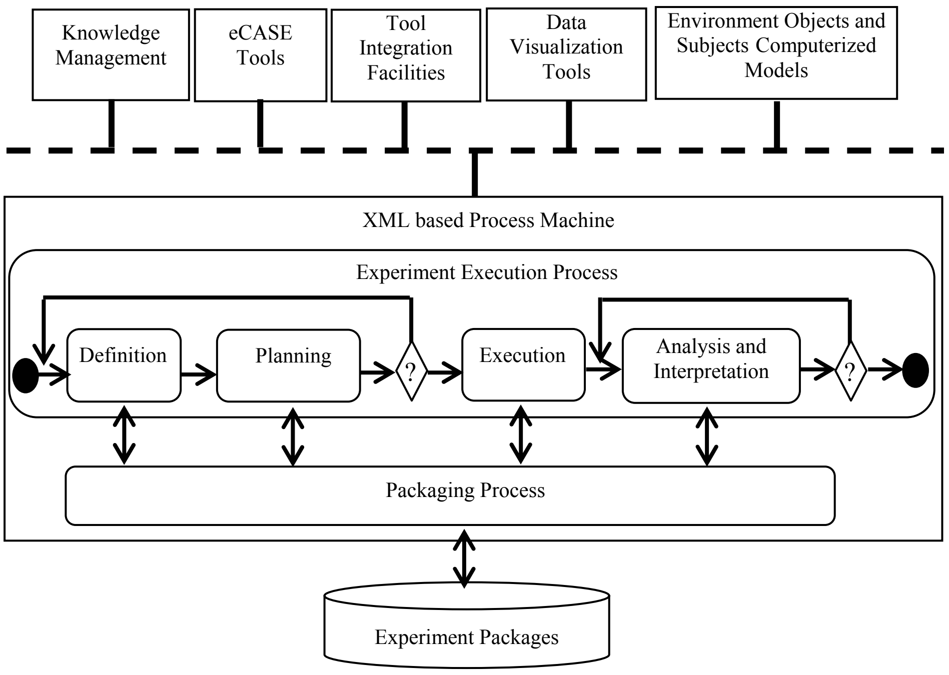
Knowledge Management
eCASE Tools
Tool Integration Facilities
Data Visualization Tools
Environment Objects and Subjects Computerized Models
XML based Process Machine
Experiment Execution Process
Definition
Planning
?
Execution
Analysis and Interpretation
?
Packaging Process
Experiment Packages

Fig. 3. eSEE's Services and Tools – adapted from [16]

Some services and tools that belong to eSEE, such as the XML based process machine and the tool integration mechanism, were already developed, and now, their integration to eSEE's infrastructure is being defined [17].

4 Mapping ISO 12207 into the Experimental Process

Based on the ESE group software development and experimental studies development know-how, we observed that several Experimental SE practices are similar to Software Engineering ones, such as documentation, training, configuration management, process management, and so on. Therefore, we believe that the experimentation process could be improved by incorporating some of the ISO/IEC

12207 processes. The objective is to include Software Engineering characteristics to the Experimental process.

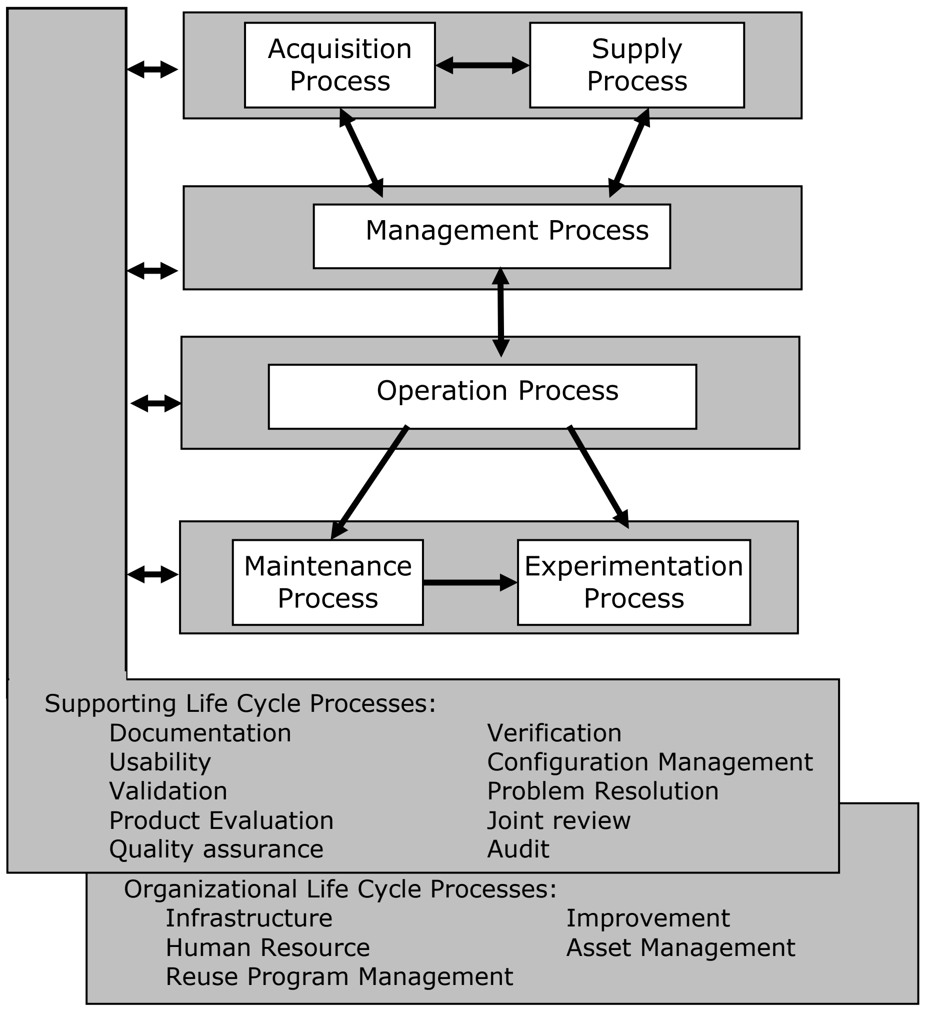

Fig. 4. Extending the Experimentation Process

An initial mapping of ISO's processes into the experimentation life cycle was already made in order to organize the experimental process. Two kinds of mapping can be considered:

- Mapping ISO/IEC 12207's processes or sub-processes into activities or sub-activities of the experimental process: for instance, as defined in [8], an experiment *Planning* stage's sub-activity is *Results Validation Adequacy*. This sub-activity is concerned with the results' validity for the population of interest. In this phase, threads to experimental results must be identified, managed and mitigated. Therefore, this sub-activity was mapped as ISO/IEC 12207's *Risk Management Process* (a sub-process of Management Process).

- Mapping ISO/IEC 12207's processes as supporting processes to the experimental life cycle: the experimental process' assets (documents, experimental studies' plans, experimental packages, and so on) must be under the supervision of a *Configuration Management Process* (one of ISO's supporting life cycle processes) to control their modifications and releases; record and report the status of the items and modification requests; ensure their completeness, consistency, and correctness and control their storage, handling, and delivery. Similarly, it's necessary to ensure that individual knowledge, information and skills are collected, shared, reused and improved throughout the experimental process execution. This task may be supported by ISO's *Knowledge Management Process* (a sub-process of the Human Resource Process).

Therefore, the experimental process defined in [8] has been extended as shown in Figure 4. ISO 12207's *Primary Life Cycle Processes, Supporting Life Cycle Processes* and *Organizational Life Cycle Processes* were mapped into *Experimental Supporting Processes*, which provide the necessary supporting activities and tasks for managing Experimentation Process execution.

5 eSEE's Initial Features

As discussed in the former section, the experimental SE activities have a lot in common with software engineering ones. Therefore, an environment for experimental software engineering may have similar characteristics to software engineering environments, such as the TABA Workstation [12].

The work of [18] defined the integration philosophy for the TABA Workstation and it considers four integration dimensions: *data dimension, control dimension, presentation dimension and knowledge dimension*. [19] considers during tools integration in software engineering environments, besides the data, control and presentation dimensions, also the *platform dimension*.

Inspired on these dimensions, we categorized eSEE's initial set of features (including requirements and quality characteristics) according to them, adding a specific dimension related to experimentation. Thus, six features categories were defined, as shown in Table 1.

Especially regarding knowledge sharing issues, the eSEE infrastructure intends to support the four phases of the experimental knowledge sharing model proposed by [10].

Therefore, the initial set of eSEE features considers the EKSM model and it aims at providing computerized facilities to support all EKSM phases. For instance, the Internalization phase may be supported by content publication facilities and the Socialization phase by content distribution facilities.

Table 1. eSEE's Features Categorization

Category	eSEE Features
Platform	Portabilility, Scalability, Support to Extensibility and Support to Ubiquity
Presentation	User Interface and Accessibility
Control	Access Control, Artifacts Version Control, Support to Integration, Support to Workflow and Support to Measurement (metrics)
Knowledge	Support to Content Publication, Support to Content Distribution, Meta-Data Management and Intelligent Assistance
Data	Experimental Packages Security, Support to Artifacts Packaging, Search Mechanisms and Shared Data Storage Model
Experimentation	Support to Families of Experiments, Support to the Experimentation Workflow, Support to Templates Definition, Support to Experimental Processes Activities, Support to Experimental Studies Replication and Support to Experimental Studies Valuation

6 Conclusions and Future Work

In this paper, we have introduced the concept of eSEE, an infrastructure able to instantiate software engineering environments to support knowledge management throughout the SE experimentation process, including the definition, planning, execution and packaging of experimental studies in software engineering.

The working product composing eSEE will be a set of integrated software components, which will be applied to support Software Engineering experimentation processes.

To test these ideas, an initial prototype is being built. This prototype is based on the extended experimentation packaging process developed in [8], exploring a different development platform – Zope+Plone [20].

We intend to instantiate some eSEE examples to perform experimental studies using them. This is necessary to evaluate the use of eSEE's infrastructure. It's also intended that experiments in Software Engineering should be performed using the TABA workstation, and the integration between TABA and eSEE must be defined. We intend to follow the process for developing new technologies proposed by [5] to play and execute the infrastructure evaluation.

We expect that this research can contribute with the following results:

- Making available a set of software components that can compose a computerized infrastructure to allow instantiation of experimentation environments for different experimental studies types in Software Engineering;
- Reducing efforts and consequently the time for defining, planning, analyzing and packaging experimental studies in Software Engineering;

- Making available knowledge and associated tools that can be used by the academia and software industry to improve the development of their software technologies;
- Allowing the replication of experimental studies, what can support the improvement of SE body of knowledge and;
- Extending the experimentation process defined in [7], by mapping ISO 12207's processes into the experimental life cycle.

The next steps of this work include the following activities:

- Experimentation process adaptation, regarding the four staged taxonomy [9] and the definition of a standard experimentation process to support each one of them;
- Experimentation ontology definition to organize knowledge about experimental software engineering in eSEE; used to identify general characteristics of each experimental study type;
- To define an initial set of eCASE tools to populate the eSEE infrastructure.

Further information about the ESE team at COPPE/UFRJ and all the research work that contribute to this research can be obtained at the ESE team's homepage (http://www.cos.ufrj.br/~ese).

7 Acknowledgments

The authors acknowledge CAPES and CNPq for the financial support to this work. We also thank the members of the NSF-CNPq READERS II Project team and the ESE team for their valuable comments regarding this research. We especially thank Tayana Conte and Wladmir Chapetta for they contribution to this work regarding eSEE initial features.

References

1. Pfleeger, S.L.: Albert Einstein and Empirical Software Engineering. IEEE Computer, (1999).
2. Basili, V.R., Shull, F., Lanubile, F.: Building Knowledge through Families of Experiments. IEEE Transactions of Software Engineering, vol. 25, No. 4, July/August (1999).
3. Tichy, W.F.: Should Computer Scientists Experiment More? 16 Reasons to Avoid Experimentation. IEEE Computer, 31(5), 32-40, May (1998).
4. Sjøberg, D.I.K., Anda, B., Arisholm, E., Dybå, T., Jørgensen, M., Karahasanovic, A., Koren, E.F., Vokac M.: Conducting Realistic Experiments in Software Engineering. Proc. of ISESE'2002, Nara, Japan (2002).
5. Shull, F., Carver, J., Travassos, G.H.: An Empirical Methodology for Introducing Software Processes. Proc. of 8th European Software Engineering Conference (ESEC) and 9th ACM SIGSOFT Symposium on the Foundations of Software Engineering (FSE-9), Vienna (2001).
6. Newman, H.B., Ellisman, M.H., Orcutt, J.A.: Data-Intensive E-Science Frontier Research, Communications of the ACM, vol. 46, no. 11, November (2003).
7. Arisholm, E., Sjøberg, D., Carelius, G.J.: A Web-based Support Environment for Software Engineering Experiments. Nordic Journal of Computing, vol 9 (4), 231-247 (2002).

8. Amaral, E. A. G., Travassos, G. H.: Em busca de uma abordagem para Empacotamento de Experimentos em Engenharia de Software. Proc. of 2a JIISIC - Jornada Ibero-Americana De Engenharia de Software e Engenharia de Conhecimento, Salvador (2002).
9. Travassos, G. H.; Barros, M. O.: Contributions of In Virtuo and In Silico Experiments for the Future of Empirical Studies in Software Engineering. Proc. of 2nd Workshop in Workshop Series on Empirical Software Engineering the Future of Empirical Studies in Software Engineering, Roma (2003).
10. Shull, F., Mendonça, M., Basili, V., Carver, J., Maldonado, J. C., Fabbri, S., Travassos, G. H., Ferreira, M. C.: Knowledge-Sharing Issues in Experimental Software Engineering. Empirical Software Engineering An International Journal, USA, v. 9, n. 1-2, p. 111-137 (2004).
11. Nonaka, I., Takeuchi, H.: The Knowledge Creating Company. Oxford, UK: Oxford University Press (1995).
12. Oliveira, K.M., Zlot, F., Rocha, A.R.C., Travassos, G.H., Silva, C.G.M., Menezes, Silva C.: Domain Oriented Software Development Environment, Journal of Systems and Software, v. 72, n. 2, p. 145-161 (2004).
13. Mian, P.G., Travassos, G.H, Rocha, A.R.C.: eSEE: Computational Infrastructure for Experimental Software Engineering. Proc. of IX Workshop de Teses em Engenharia de Software (XIX SBES), Brasília, Brazil (2004).
14. Mafra, S.N., Barros, M.O., Travassos, G.H.: enactPro: Automatizando Processos de Software, Caderno de Ferramentas do Simpósio Brasileiro de Engenharia de Software¸ Brasília, Brazil (2004).
15. Silva, L.F.S., Travassos, G.H.: Tool-Supported Unobtrusive Evaluation of Software Engineering Process Conformance. Proc. of ISESE 2004, Redondo Beach CA, USA (2004).
16. Travassos, G. H., Silva, L. F. S., Spinola, R. O., Kalinowski, M., Chapetta, W.: Tools and Facilities for Experimentation: Tools and Facilities for Experimentation: Can we get Can we get there? Panel presentation on the 11th International Software Engineering Research Network Meeting (ISERN 2003), Italy, (2003).
17. Spinola, R.O., Kalinowski, M., Travassos, G.H.: Uma Infra-Estrutura para Integração de Ferramentas CASE. Proc. of XIX Simpósio Brasileiro de Engenharia de Software (SBES 04), Brasília, Brazil (2004).
18. Travassos, G. H.: O Modelo de Integração de Ferramentas da Estação TABA. Doctorate Thesis, Programa de Engenharia de Sistemas e Computação, COPPE/UFRJ, Rio de Janeiro, RJ, Brasil, (1994).
19. Wasserman, A.I.: Tool integration in software engineering environments, F.Long Ed., Software Engineering Environments, Berling: Springer-Verlag, p. 138-150 (1990).
20. Bernstein, M. R., Robertson, S., Codeit Development Team.: Zope Bible. Hungry Minds, Inc. New York (2002).