SOME ISSUES ON RESEARCH ESSENTIALS IN THE FIELD OF SOFTWARE ENGINEERING
Simplified Look on Scientific Method for Bachelor Level Research

Oksana Nikiforova
Department of Applied Computer Science, Riga Technical University, Kalku 1, Riga, Latvia

Marite Kirikova
Department of System Theory and Design, Riga Technical University, Kalku 1, Riga, Latvia

Renate Strazdina
Department of System Theory and Design, Riga Technical University, Kalku 1, Riga, Latvia

Keywords: Research methodology, software engineering versus scientific method, bachelor level research.

Abstract: Not every software engineering task will qualify as scientific research, because the solution of the problem should contribute to the body of scientific knowledge. This means that, on one hand, the research work should correspond to the state of the art of scientific body of knowledge and give a particular, small, contribution to it, and, on the other hand, the work of the student shall clearly contribute to at least one general phase of software development. Taking into consideration that bachelor thesis are an initial research and the simplest scientific research at the university, the minimum requirements for the scope of bachelor thesis, thus, should not exceed one software development phase and one research phase, still keeping the requirement that the bachelor student must be able to position clearly his work in both, namely, scientific and software development life cycles. Therefore the life cycle of scientific research is analyzed from the perspective of software engineering life cycle and the main activities of both are mapped into single schema. 16 types of bachelor thesis in software engineering proposed in the paper are a helpful tool for bachelor thesis developers and advisers to meet the above mentioned requirements.

1 INTRODUCTION

Strong traditions and requirements for the process of scientific research exist in the fundamental fields of science such as physics, mathematics, chemistry. The same applies to social and economical fields of science. Since science appearance and first scientists researches in the ancient days (Kuhn, 1962) there are developed different guidelines and methodologies for application of scientific methods as well as for their selection for certain kind of scientific solution development. However, this does not apply fully to the research in the area "Software Engineering".

Software engineering is an engineering discipline concerned with the problem of system development, where *system* is understood as a software system. Software engineering is not just programming nor just computer science. (Sommerville, 1992). From research perspective software engineering is a relatively new and heterogeneous field of science, where generally accepted norms for application of scientific methods are yet not established. It is quite enough complicatedly to define a boundary between the required level of problem formalization and simplicity of problem solution. In addition to that, researches in the field of system engineering in general and in software engineering in particular requires to take into consideration not only technological aspects of problem solutions, but also to be concerned with economical and social aspects of application domain. There are many and varied classifications of sciences (Bunge, 1967), based on different criteria. Depending on the kind of science, different research methods are used. None of these

48 Nikiforova O., Kirikova M. and Strazdina R. (2007).
SOME ISSUES ON RESEARCH ESSENTIALS IN THE FIELD OF SOFTWARE ENGINEERING - Simplified Look on Scientific Method for Bachelor Level Research.
In *Proceedings of the Second International Conference on Evaluation of Novel Approaches to Software Engineering* , pages 48-55
DOI: 10.5220/0002584700480055
Copyright © SciTePress

methods, however, seems to be totally suited for scientific research in the field of Software Engineering. As it is proposed in (Marcos, Marcos, 1998), the branches of engineering do not fit perfectly into the classifications proposed in the literature, although they are related with most of the disciplines appearing in them. For this reason, the search for the method appropriate to research in the field of software engineering is becoming a research field itself (Dobson, 2001.), (Glass, etc., 2002), (Gregg, etc., 2001), (Marcos, Marcos, 1998).

Especially the above stated problem is actual for the first scientific research, which in the most general case is development of a bachelor thesis. The bachelor thesis usually is the first scientific research for student and often problems arouse to understand that the scope of the research, its level of detail and suggestions on formal foundations for problem solutions are adequate to the requirements for scientific thesis. The following problems are identified in bachelor thesis development:

1) disorientation in the collection of scientific methods and inability to select the required method for certain task solution;

2) weak skills in usage of bibliography references and appropriate literature search in scientific data bases and further its analysis and application;

3) lack of understanding of what is scientific research and failure of understanding its process in progress.

Normative acts of Riga Technical University define bachelor thesis as "student research, where he present his abilities to use acquired knowledge, to make analysis of problem and its solutions presented in concerned bibliography, to propose problem solution based on analytical results". But it is not enough let to understand the process of scientific research especially in so "extraordinarily" field of science as it is system engineering.

The paper discusses several issues relevant in bachelor thesis development. The first issue we discuss is the similarity between research the process as such and software development process in software engineering. We deal with this issue in sections 2 and 3. The second issue concerns variety of options in choosing bachelor thesis topics and their positioning in the map of possible topics. The map of possible topics and 16 bachelor thesis types are introduced in Section 4. In section 5 we conclude with the third issue, namely, how different software development paradigms and different study programs can shape the research done by bachelor level students in their bachelor thesis development.

2 GENERAL FOUNDATIONS OF RESEARCH METHODOLOGY

Research methodology based on scientific method application is a process for making observations, recording data, and analyzing data that can be duplicated by other scientists. In 1637 René Descartes published his "Discours de la Méthode" in which he described systematic rules for determining what is true, thereby establishing the principles of the scientific method. (Wilson, 1952). The simple version of it looks like this (see also Figure 1):
1) observe some aspect of the universe;
2) invent a tentative description, called a *hypothesis*, that is consistent with what you have observed;
3) use the hypothesis to make predictions;
4) test predictions by experiments or further observations and modify the hypothesis in the light of the results;
5) repeat steps 3 and 4 until there are no discrepancies between theory and experiment and/or observation. (Wilson 1952),

When consistency is obtained the hypothesis becomes a *theory,* which is then a framework within which observations are explained and predictions are made. (Barrow, 1991).

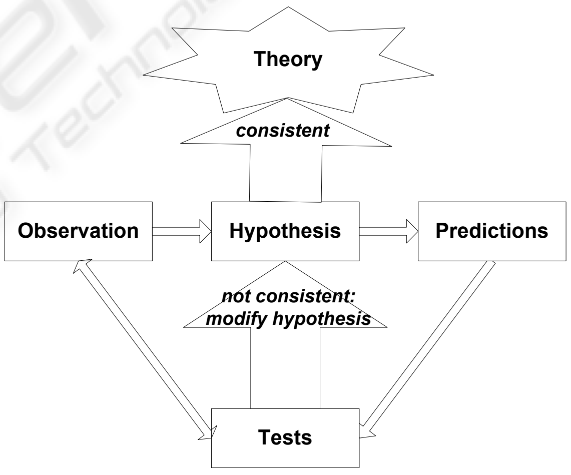

Figure 1: Simplified life cycle of scientific research.

3 MAPPING OF SOFTWARE DEVELOPMENT ACTIVITIES INTO RESEARCH LIFE CYCLE

For the first scientific research usually it is difficult to understand what is hypothesis, how to advance it and how to predict its consistency and so on. But taking in advance high enough students skill in software lifecycle implementation for software development it is possible to describe steps of scientific methods in terms close to students understanding.

Marcos in (Marcos, 2005) assumes that the object of study of the branches of engineering (and software engineering in particular) differs from the object of study of the formal and empirical science (be they sociological or natural). While these subjects deal with the study of existing phenomena or objects, engineering sciences deal with the study of the methods and techniques necessary for the creation of new objects and even with the creation of such methods and techniques. Therefore it has important similarities with the application of software engineering, which is also concerned with creation (of software products). Thus, it is possible to establish an analogy between the process of research in the field of software engineering and process of software development. (Marcos, 2005).

Every software development effort goes through a lifecycle, a process that includes all activities in the development cycle that take place up to initial release. The main function of a lifecycle model is to establish the order in which a project specifies, implements, tests, and performs its activities.

Software development is a process for organized production of software, using a collection of predefined techniques and notational conventions. (Nikiforova, 2001). In general, software development life cycle defines activities for software development and in spite of difference between life cycle models, that can be presented as waterfall (Royce, 1970), spiral (Boehm, 1986), fountain (Hedrerson-Sellers, 1993), as well as its advanced combinations in Rational Unified Process (Jacobson, 1999) and Microsoft Solution Framework (MSF, 2006), it is possible to identify the following stages of software development: analysis, design, implementation and testing (shown in Figure 2). (Nikiforova, 2002).

Usually, software development starts with an idea. System analysis includes careful acquisition and examination of the requirements for a software system with the intent of understanding them, exploring their implications, and removing inconsistencies and omissions. System design presents overall system architecture. During system design the target system is organized into components based on both the analysis structure and the oncoming architecture. The end product of analysis and design is system representation that corresponds to the requirements and is used for further software implementation, which is a direct translation of system model into a programming language (Kleppe, et. al, 2003). Testing is applied for implemented code verification and validation according the preliminary requirements.

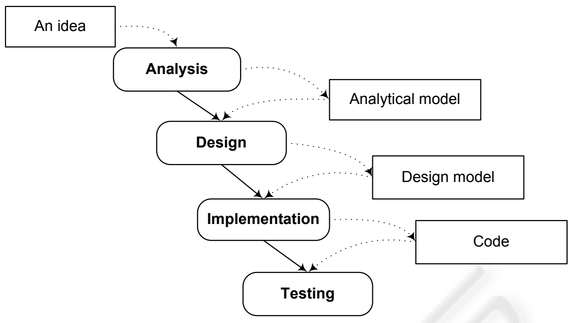

Figure 2: Generalization of software development process.

Therefore from one side taking as a basis schema of scientific method presented in Figure 1 and from other side applied general phases of system development shown in Figure 2 a composite schema for research methodology can be presented as shown in Figure 3.

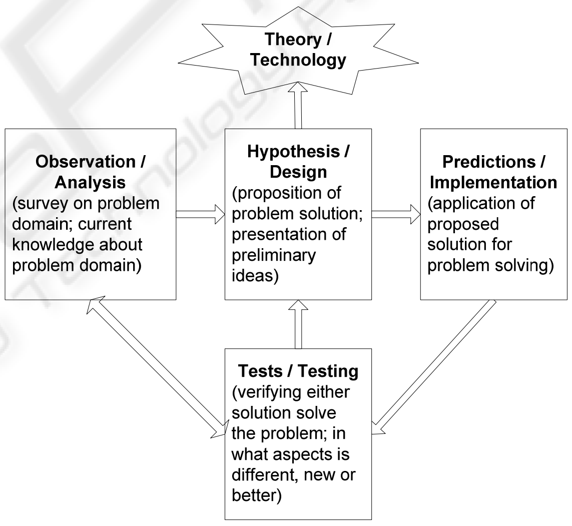
Theory / Technology
Observation / Analysis (survey on problem domain; current knowledge about problem domain)
Hypothesis / Design (proposition of problem solution; presentation of preliminary ideas)
Predictions / Implementation (application of proposed solution for problem solving)
Tests / Testing (verifying either solution solve the problem; in what aspects is different, new or better)

Figure 3: Mapping of stages for software development into schema of scientific method.

Aspect observation can be understood as problem domain analysis and knowledge acquisition about the “science needs” actual at the moment. Proposition of problem solution, which is better understood as solution architecture development, can serve as setting of the hypothesis. Acquirements for application of proposed solution or realization of developed architecture can help for understanding what should be done during scientific research predictions.

Verifying either solution solves the problem in some extent can be based on methods applied for software testing as far as mathematical data analysis methods, which academic background of software specialists, can be applied for discussing of what aspects are better, different or new in the proposed problem solution.

4 VARIETY OF OPTIONS IN CHOOSING BACHELOR THESIS DIMENSION

The result of bachelor level research is a Bachelor thesis. Analysis of bachelor theses topics at the Institute of Applied Computer Systems revealed a wide variety of topics considerably differing in depth and breadth of topics with respect to both software development phases and research life cycle. However, there was a clear tendency to keep a balance between student's contribution to research as such and his/her contribution to software engineering as an engineering discipline.

4.1 Cross Screening of Software Development Phases Via Scientific Research Life Cycle

The situation with variety of bachelor thesis dimensions is illustrated in Table 1 where cross screening of simplified software development and scientific research processes is given. Rows of Table 1 correspond to the phases of simplified research process.

Table 1: Cross screening of software development phases via scientific research life cycle.

Software development phases / *Scientific research life cycle*	1. Analysis	2. Design	3. Implementation	4. Testing
1. Observation	1.1.	1.2.	1.3.	1.4.
2. Hypothesis	2.1.	2.2.	2.3.	2.4.
3. Predictions	3.1.	3.2.	3.3.	3.4.
4. Tests	4.1.	4.2.	4.3.	4.4.

Columns of Table 1 correspond to the phases of simplified software engineering life cycle. Despite both processes having considerable similarities (Fig. 3) there are particular differences with respect to the products to be delivered by activities in each life cycle phase (Marcos, 2005).

It is also important to take into consideration that a bachelor thesis is not a PhD thesis and is defined as "analytical research with scientific elements", therefore the student is not requested to go through the entire life cycle of scientific research. While such an option is possible for simple software engineering problems in more complex problems the student most probably will perform his / her research work only according to the part of the research life cycle. In these complex tasks, however, it is necessary for the student to understand how his / her activities are placed in the research life cycle and what his / her contribution to the research is. The same principle applies to the software engineering dimension, i.e. software development phases.

The diagonal of Table 1 actually corresponds to Fig. 3 and is an option where performance with respect to both life cycles would be nearly identical. It should be noted here that in this situation every software engineering task will not qualify as scientific research because the solution of the problem should contribute to the body of scientific knowledge, not just to the needs of a particular business organisation that uses students' skills instead of licences of existing products. Thus the contribution to scientific body of knowledge is the main indicator of presence of scientific method (or a part of it) in a bachelor thesis. In other words students should satisfy science needs instead of software requirements which are defined by user in software engineering life cycle.

This means that in at least one cell in Table 1 student's work should correspond to the state of the art of scientific body of knowledge and give a specific, small, contribution to this body of knowledge. Similarly the work of the student shall clearly contribute to at least one general phase of software development.

4.2 Collection of Types for Bachelor Thesis in the Field of Software Engineering

Taking into consideration that bachelor theses are the simplest scientific research at the university, the minimum requirements for the scope of bachelor thesis should not exceed one software development phase and one research phase still keeping the requirement that the bachelor student must be able to position clearly his work in both scientific and software development life cycles (see Fig. 4). The numbers of phases in Figure 4 correspond to the numbers given in Table 1.

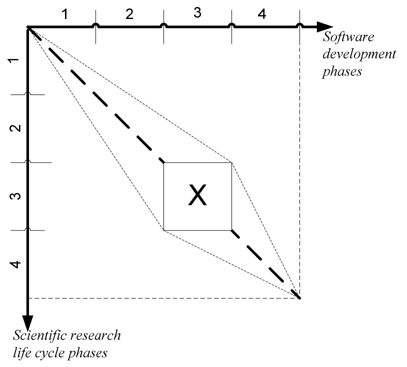
1
2
3
4
Software development phases
1
2
3
4
X
Scientific research life cycle phases

Figure 4: Mapping of stages for software development into schema of scientific method.

In terms of generalisation of software engineering and scientific research lifecycles according to the positioning principle reflected in Figure 4 we come to the 16 basic types of bachelor thesis in software engineering presented in Figure 5. Each type is briefly described in the remainder of this section. The point of main emphasis of the bachelor thesis, in terms of phases of scientific research and software development, is denoted as a scope. Positioning according to the software development phases is denoted by a). The positioning according to the scientific research phases is denoted by b). Each type is illustrated by at least one instance, which is typical for the author's research institution.

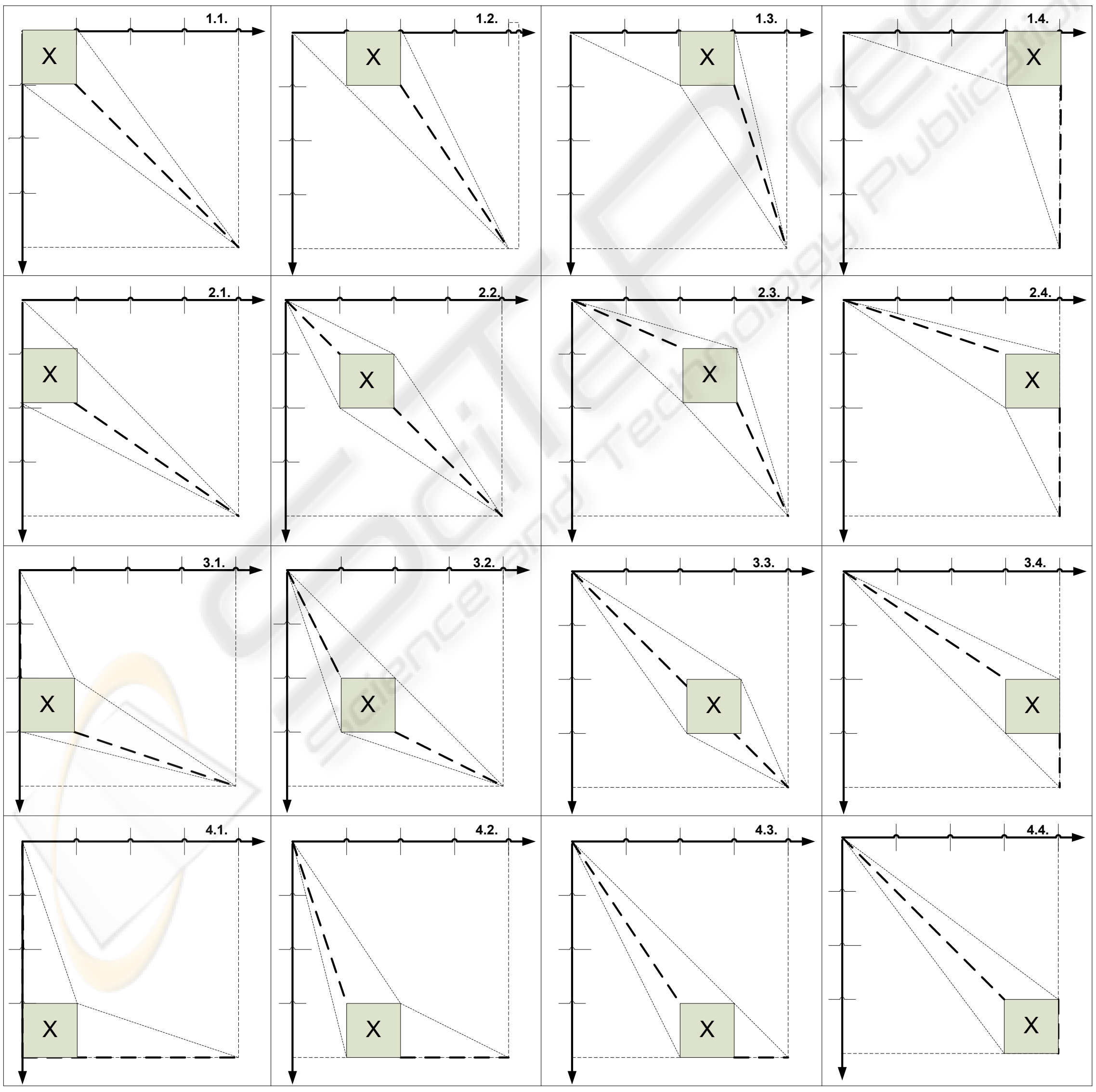

Figure 5: Collection of basic types for bachelor thesis in the field of software engineering.

Type 1.1.
Scope: Observation of issues relevant to analysis phase of software development.
Positioning: a) Describe why and probably how these issues influence further phases of software development, b) Describe how could results of observation influence further phases of the research cycle.
An instance: Analysis of a new approach, statement, concept documents, field of interest, problem.

Type 1.2.
Scope: Observation of issues relevant to design phase of software development.
Positioning: a) Describe why these issues are relevant from the point of view of analysis phase and describe why and probably how these issues could influence further phases of software development life cycle, b) Describe how the results of observation could influence further phases of the research cycle.
An instance: Analysis of one or different design solutions.

Type 1.3.
Scope: Observation of issues relevant to implementation phase of software development.
Positioning: a) Describe why these issues are relevant from the point of view of analysis and design phases of software development and probably how they could influence the testing phase, b) Describe how could results of observation influence further phases of the research cycle.
An instance: Analysis of one or different systems.

Type 1.4.
Scope: Observation of issues relevant to testing phase of software development.
Positioning: a) Describe why these issues are relevant from the point of view of previous phases of software development, b) Describe how could results of observation influence further phases of the research cycle.
An instance: Analysis of different sources describing the same problem.

Type 2.1.
Scope: Finding the solution to a particular problem relevant in the analysis phase of software development.
Positioning: a) Describe how the solution could probably influence further phases of software development. b) Describe why the problem is relevant for software engineering research, where and how would the solution be applicable and probably how could its real value be assessed.
An instance: Analysis or modification of existing hypothesis.

Type 2.2.
Scope: Finding the solution to a particular problem relevant in the design phase of software development.
Positioning: a) Describe why the solution is relevant from the point of view of the analysis phase of software development and probably how the solution could influence further phases of software development. b) Describe why the problem is relevant for software engineering research, where and how would the solution be applicable and probably how could its real value be estimated.
An instance: Defining of new hypotheses (after 1.1. only).

Type 2.3.
Scope: Finding the solution to a particular problem relevant for the implementation phase of software development.
Positioning: a) Describe why the solution is relevant from the point of view of the analysis and design phases of software development and probably how it could influence the testing phase, b) Describe why the problem is relevant for software engineering research, where and how would the solution be applicable and probably how could its real value be assessed.
An instance: Evaluation or implementation of defined hypothesis (after 2.2. only)

Type 2.4.
Scope: Finding the solution to a particular problem relevant to the implementation phase of software development.
Positioning: a) Describe why the solution is relevant from the point of view of previous phases of software development, b) Describe why the problem is relevant to software engineering research, where and how would the solution be applicable and probably how its real value could be assessed.
An instance: Testing of a new (from 2.2.) or modified (from 2.1.) hypothesis; testing of an existing hypothesis in different fields.

Type 3.1.
Scope: Application of a particular solution relevant to the analysis phase of software development in different real, experimental or simulated situations.
Positioning: a) Describe how the results of application exercise could probably influence further phases of software development, b) Describe why the solution is relevant for software engineering research, how would the results of application of solution help to evaluate the real value of the solution.
An instance: Developing of a concept document for a system.

Type 3.2.
Scope: Application of a particular solution relevant to the design phase of software development in different real life, experimental or simulated situations.
Positioning: a) Describe why the application exercise is relevant from the point of view of the analysis phase of software development and probably how would the results of application exercise influence further phases of software development, b) Describe why the solution is relevant to software engineering research, how would the results of application of solution help to evaluate the real value of the solution.
An instance: Design of an agent, system, technology (after 3.1. only).

Type 3.3.
Scope: Application of a particular solution relevant to the implementation phase of software development in different real life, experimental or simulated situations.
Positioning: a) Describe why the application exercise is relevant from the point of view of analysis and design phases of software development and probably how would the results of application exercise influence the testing phase, b) Describe why the solution is relevant to software engineering research, how would the results of application of the solution help to evaluate the real value of the solution.
An instance: combining different methods; implementation of a system, agent, technology (after 3.1 and 3.2. only).

Type 3.4.
Scope: Application of a particular solution relevant to the testing phase of software development in different real, experimental or simulated situations.
Positioning: a) Describe why the application exercise is relevant from the point of view of previous phases of software development, b) Describe why the solution is relevant to software engineering research, how would the results of application of the solution help to evaluate the real value of the solution.
An instance: Developing of a testing document of a system

Type 4.1.
Scope: Scientific evaluation of a particular solution relevant to the analysis phase of software development.
Positioning: a) Describe how the results of the evaluation could probably influence further phases of software development, b) Describe why the solution and its evaluation are relevant to software engineering research.
An instance: Evaluation of different approaches, methods, technologies;

Type 4.2.
Scope: Scientific evaluation of a particular solution relevant to the implementation phase of software development.
Positioning: a) Describe why the results of the evaluation are relevant from the point of view of the analysis phase of software development and describe probably how the results of the evaluation could influence further phases of software development, b) Describe why the solution and its evaluation are relevant to software engineering research.
An instance: Evaluation of different designs (things not implemented yet)

Type 4.3.
Scope: Scientific evaluation of a particular solution relevant to the design phase of software development.
Positioning: a) Describe why the results of the evaluation are relevant from the point of view of analysis and design phases of software development and describe probably how the results of the evaluation could influence the testing phase, b) Describe why the solution and its evaluation are relevant to software engineering research.
An instance: Evaluation of different systems.

Type 4.4.
Scope: Scientific evaluation of a particular solution relevant to the testing phase of software development.
Positioning: a) Describe why the results of the evaluation are relevant from the point of view of previous phases of software development, b) Describe why the solution and its evaluation are relevant to software engineering research.
An instance: Approbation and evaluation of a testing document of a system

The collection describes the basic types only, but the research can not only to be positioned in the strongly divided phases of software engineering and not to be framed by division between scientific research phases. Because the phases of software engineering are often overlapped each by another and the research can be positioned as a frontier for several phases. As for selection of implemented steps of scientific research the only requirement is let minimally one phase observation, hypothesis, implementation or scientific testing, would be accomplished for bachelor level of the research.

5 CONCLUSIONS

It is often difficult to understand the necessary level of details or depth of the research, which is required for bachelor thesis. Especially unambiguous this is for research in the field of software engineering, where it is enough complicated to state the boundary between the necessary level of formalization in the research and the ability to present results clear and perceptible enough. The paper presents the discussion about ability to simplify the scientific research life cycle and to try to map it into the schema of software development life cycle for better understanding of the research performance for bachelor level students. The cross screening of scientific research through the phases of software development life cycle give the ability not only to describe the scientific research process in terms of software engineering, which are closer to students for understanding, but also to define the scope and positioning of the main dimensions of bachelor thesis in the field of software engineering.

The division between sub areas in the field of software engineering depends also on the subdivision of the exact phase of software development. So far one more aspect of scientific research inspection according the software development stages can be the discussion about possible sequences of stages, so called models of life-cycles of software development. Analysis of different paradigms existing in software development let to assume its possible existence in scientific research, like agile versus disciplined, structural versus object-oriented, sequential versus iterative and so on by analogy with the same in software development. The collection of basic types for bachelor thesis dimensions can serve as a framework for research initialising and positioning it in both life cycles – the scientific and engineering ones. Also the discussion about bachelor thesis types can be useful for students in selection of appropriate methods and strategy for research as well as for their advisors to define recommendations on research performance.

ACKNOWLEDGEMENTS

The research reflected in the paper is supported by the research grant No. ZP/2006-05 of Riga Technical University “Development of the Method for the First Level Scientific Research in the Area of Computer Science and Information Technology” and by the European Social Fund within the National Programme "Support for the carrying out doctoral study program's and post-doctoral researches".

REFERENCES

Barrow, J., 1991. *Theories of Everything*, Oxford University Press.

Boehm, B. W., 1986. A Spiral Model of Development and Enhancement. In *Software Engineering Notes, Vol. 11, No. 4, August*.

Bunge, M., 1967. *Scientific Research.* Springer, Berlín, 2 vols.

Dobson, P. J., 2001. The Philosophy of Critical Realism-An Opportunity for Information Systems Research. In *Information Systems Frontiers*, 3:2, pp. 199-210.

Glass, R.L., Vessey, I., Ramesh, V., 2002. Research in software engineering: an analysis of the literature. In *Information and Software Technology*, Elsevier Science B.V. N.44, pp. 491-506.

Gregg, D. G., Kulkarni, U. R., Vinzé, A. S., 2001. Understanding the Philosophical Underpinnings of Software Engineering Research in Information Systems. In *Information Systems Frontiers*, 3:2, pp. 169-183.

Henderson-Sellers, B., 1998. *The OPEN-Mentor Methodology*, Object Magazine.

Henderson-Sellers, B., Edwards, J.M., 1993. The Object-Oriented Systems Lifecycle. In *Communications of the ACM, **33**(9)*.

Jacobson, I., Booch, G., Rumbaugh, J., 1999. *The Unified Software Development Process*, Addison-Wesley.

Kleppe, A.G., Warmer J.B., Bast W., 2003. *MDA Explained: The Model Driven Architecture: Practice and Promise*, Addison-Wesley.

Kuhn, T., 1962. *The Structure of Scientific Revolutions*, University of Chicago Press.

Marcos, E., 2005. Software Engineering Research versus Software Development. In *ACM SIGSOFT Software Engineering Notes*, July 2005, Vol. 30, No. 4, pp. 1-7.

Marcos, E., Marcos, A., 1998. An Aristotelian Approach to the Methodological Research: a Method for Data Models Construction. In *Information Systems- The Next Generation.* L. Brooks and C. Kimble (Eds.). Mc Graw-Hill, pp. 532-543.

MSF, 2006. Microsoft Solution Framework – available at http://www.microsoft.com/technet/solutionaccelerators/msf/default.mspx.

Nikiforova, O., 2001. *Comparison Methodology of Software Development Means*, PhD Thesis, Riga Technical University, Latvia.

Nikiforova, O., 2002. General Framework for Object-oriented Software Development Process. In *Scientific Proceedings of Riga Technical University, Computer Science, Applied Computer Systems,* 3rd thematic issue, pp.132-144.

Royce, W.W. 1970. Managing the development of large software systems. In Proceedings of WESTCON, San Francisco.

Sommerville, I., 1992. *Software Engineering*. Addison-Wesley.

Wilson, E. B., 1952. *An Introduction to Scientific Research*, McGraw-Hill.