PhD in Information Technology and Electrical Engineering

Università degli Studi di Napoli Federico II

PhD Student: Vincenzo De Simone

XXX Cycle

Training and Research Activities Report – Third Year

Tutor: Anna Rita Fasolino

Table of Contents

Information

I am Vincenzo De Simone and I obtained a Ms.Sc. Degree (*Laurea Magistrale*) in Computer Engineering (*Ingegneria Informatica*) at the University of Naples *Federico II* in December 2013.

I am attending the 30th cycle of the Information Technology and Electrical Engineering (ITEE) PhD Program with a **P.O.R. F.S.E.** grant. I am carrying out my research activity within the ReVERSE Group (Research Group of Software Engineering) under the supervision of **Prof. Anna Rita Fasolino**.

Study and Training activities

Courses

During the third year of the PhD Program, I attended one Occasionally Provided Course. ***Table 1*** reports the information about the attended course. Moreover, I attended the Cambrish English CAE Advanced C1 Course at the CLA (Centro Linguistico di Ateneo). I obtained the C1 certificate.

Table 1 – List of attended courses

Course type	Course Name	Provided by	Date	Number of Credits
Occasionally Provided	**Le imprese e la ricerca: Gestione strategica dell'innovazione**	Dr. Marco Frizzarin	2/03/2017	4
	Cambridge English: CAE Advanced C1	Geraint Ross Thomas	06/07/2017	-

Seminars

During this year, I also attended the seminars reported in ***Table 2***.

Table 2 - List of attended seminars

Title	Presenter	Date	Credits
Exploiting machine learning techniques in software development processes	Domenico Amalfitano	18/01/17	0,6
IBM Cognitive Computing: Challenges and Opportunities in Building an Artificial Intelligence Platform for Business	Pietro Leo	17/02/17	0,4

Research Activity

Assessing and Improving Industrial Software Processes

The main objective of the proposed research activity, continuing the one carried out last years related to software processes, is to define methods, techniques and tools aimed at supporting the assessment and the improvement of real industrial software processes.

Nowadays, software has acquired a great relevance in our society. Different domains relay in some ways on software systems. As an example, web applications, mobile applications, software for embedded systems, software for safety-critical systems (power plants, railways, automotive, avionics, etc.), just to list a few, govern our everyday life. Since that, software processes that define how the software is developed, are gaining more and more relevance [1] [2].

In the last decades, Software Process research community focused on different aspects related to Software Process. Following the evolution of software systems, new approaches and development paradigms are being defined like the Agile, the model-based or the model-driven ones. Moreover, new methodologies and new enabling technologies are being proposed in order to improve their quality and productivity and to increment their efficiency and effectiveness. In particular, the Software Process research community has been tackling different problems related to the description, the documentation, the monitoring, the automation and the enactment as well as the assessment and the improvement of the activities related to the development of software systems. Moreover, the need to produce software with strict time-to-market and in compliance with regulations and quality standards still makes the problem of assessing software processes and improve them of great interest for the software engineering community.

Software development process is a complex phenomenon that involves several different aspects: the roles of the actors involved in the process, the artifacts that should be used and produced, the flow of activities that should be carried out and the tools that should be adopted. Moreover, methods and guidelines should be defined describing the rules to be followed in the fulfillment of the development process as well as tools supporting its execution and its automation (where and when is possible). To support the improvement of software process, several aspects need to be taken into account, since software processes crosscut the different development activities that are needed to realize a software product.

During the first year of the PhD program, I focused my attention on the issues related to the management and comprehension of the artifacts that my involved in software processes. More in detail, I considered spreadsheet based artifacts, since they are widely involved in several domains for supporting different phases of software development, such as requirements and test cases definition and analysis, etc. More in detail, in this context I proposed a Reverse Engineering process and a tool, named EXACT (Excel Comprehension Tool) for supporting the analysis and the comprehension of this kind of artifacts. The reverse engineering process was based on specific heuristic rules that were crafted according to similarities identified in the structure and formatting of spreadsheet artifacts belonging to the same artifacts corpus. The EXACT tool was designed and implemented in order to provide both analysis and visualization features aimed at easing the understanding and the comprehension of spreadsheet artifacts.

During the second and the third years, I focused my attention on proposing approaches and techniques as well as tools for supporting the assessment and the improvement of software processes that can be applied in real industrial contexts.

The quality of software products is affected by the processes followed to produce them. For this reason, *Standards* and Process Quality Evaluation Frameworks defining processes, best practices, methods and tools to be followed in order to reach certain quality attributes are being defined. Companies are demanded to comply with these Standards in order to certify the quality of their software development processes and, consequently, of their products. This demand is even higher in context where safety critical features are involved. In these cases, companies are needed to certify their process with Safety standards in order to place their products on the market. Understanding if companies are correctly implementing all the requirements defined in these Standards is not a straightforward task and both research and industrial communities are devoting a great effort in proposing solutions related to this topic.

In this context, I proposed a new approach aimed at supporting companies in effectively understanding what are the gaps with respect to the requirements imposed by a Standard or Process Quality Evaluation Frameworks they are willing to introduce. According to the information they gain with this assessment they will be able to plan the needed improvements. The approach I proposed for supporting Gap Analysis processes with respect to given Standards defines both the process to be followed and the needed tool support in order to carry out this kind of task. The approach exploits the concepts defined by the Model Driven Engineering paradigm [3] and it uses the Application Lifecycle Management (ALM) [4] as enabling technology. It guides the definition of a questionnaire able to identify the current gaps of a company with respect to a standard and allows the automatic generation of a software infrastructure that support its completion and analysis. The approach relies on metamodels we defined for supporting the definition of models related to the considered standard and of its related questionnaire. Models are then exploited to automatically produce a tool supporting the execution of gap analysis that is based on the features offered by Application Management Systems. For easing the definition of the required models and for automatically generating the ALM-based tool supporting the Gap Analysis execution I also implemented a MDE based tool, named GADGET (Gap Analysis Definition and Generation Tool).

The complexity of software processes may also be influenced by the complexity and variability of the software systems to be produced. Often companies are demanded to develop families of software systems that share a common structure and that have different variable characteristics. As another aspect, I focused on during these years was the definition of cost-effective software development processes in order to manage the variability of different software products to be developed. Handling variability of software products is a well-known issue and case-by-case basis approaches, where the software variability is managed at the end of the development process, cannot be considered suitable to resolve it. Since that, more systematic solutions need to be introduced [5]. Software Product Lines (SPL) approach has proven to be a successful solution for handling the complexity and variability of the software developed in different domains as shown by several success stories reported in [6]. According to Clements and Northrop [7], "a software product line is a set of software-intensive systems that share a common, managed set of features satisfying the specific needs of a particular market segment or mission and that are developed from a common set of core assets in a prescribed way". More in detail, during these years, I developed a

methodology to define an SPL infrastructure able to semi-automatically produce software architectural models related to different software products starting from specification documents in the context of Model Based Design software process. The methodology relies on the identification of the features of the software product and on the definition of its Product Line Architecture (PLA). PLA provides the design that is common to all SPL products, hence it describes all the mandatory and varying features of the SPL domain. Then there is the design and implementation of the SPL infrastructure allows to produce a different Product Architecture (PA) by tailoring the PLA for a specific product. The tailoring is defined by means of a Feature Profile (FP) representing the configuration of the overall architecture in terms of the features to provide and the values for configuring them. The methodology I proposed for enabling the introduction of SPL in real software development processes was supported by AutoMative, a software infrastructure I implemented for the semi-automatic generation of Product Architectures from specification documents

Moreover, I also focused on the Traceability among the artifacts involved in software processes and how it affects the quality attributes of enacted software processes. To this aim, I defined and evaluated a new approach aimed at improving the visibility, the acceptability and controllability of software processes by improving the management of traceability among the artifacts. To reach improvements of these quality attributes I proposed an approach aimed at coordinating different tasks related to a software process through the integration of the tools involved in their execution. In these situations, there is the need to correctly manage the artifacts involved in the considered tasks and to generate the correct traceability links between them as well as to coordinate the execution of the different tools needed to carry out them. More in detail, in order to correctly integrate different tools, I developed a Tool Integration architecture for coordinating the different activities of the considered software process and automatically generating the appropriate trace links among the involved artifacts. In order to developed the call connectors and data connectors for realizing the integration architecture I also had to exploit reverse engineering techniques. The proposed architecture developed to handle integration of different tasks in the context of software development processes was designed and implemented by exploiting the features offered by Application Lifecycle Management (ALM) and Continuous Integration (CI) systems.

Another problem that I was interested in was the documentation of enacted software process. People involved in the execution of a software process need to understand it, in order to execute it effectively. The adoption of systems able to manage the execution of software processes may make people unaware of some important aspects of the underlying processes, making their work even harder and more ineffective. It is important that the people involved in the execution of a software process are aware of the process and how it should be carried out. To this aim, a MDE based technique and a tool, named SPiDER (Software Process DocumEntation generator), aimed at automatically producing documentation for Software process enacted exploiting ALM systems was defined. The documentation was produced according to a documentation model I defined according to a survey I conducted in real industrial settings. The main aim of the produced documentation is to ease the work of people exploiting the tool for executing their assigned software development tasks giving them a better understanding of the underlying process.

During the years of the PhD program, I was able to apply the methodologies, the techniques and the tools proposed and developed during my research activities in the context of the research project APPS4Safety carried out by the ReVERSE (Research Group on Software Engineering) at DIETI in collaboration with Fiat

Chrysler Automobile (FCA). This project aimed at proposing new methodologies to support the development of safe vehicles. More in detail, in order to evaluate the feasibility and effectiveness of the proposed approaches and tools in real industrial scenarios, I exploited the Case Study Research methodology, following the guidelines provided by Runeson *et al.* [8] . Moreover, since I was interested in better understanding the impacts of the proposed methodologies on the actors involved in the software processes I exploited qualitative research methods that permit to obtain more information from the collected data [9].

In this context, I had the opportunity to investigate the problems related to the introduction of the software processes recommended or prescribed by a quality standard in a company. In particular, the development of software of electronic control units (ECUs) should be done in accordance with the ISO 26262 standard [10] in order to guarantee its functional safety. In order to support the systematic introduction of the processes and practices prescribed by the standard, the capabilities of the company to execute what is requested by the standard should be preliminarily verified by a gap analysis. To this aim we applied our proposed approach to support the execution of the gap analysis between the software development process carried out by the company and the one prescribed by the part 6 of the ISO 26262 standard. The proposed approach allowed to evaluate what are the elements prescribed/recommended by the standard the company is not capable to comply with and to define and evaluate the improvement actions needed to reduce these identified gaps.

Moreover, I had the possibility to evaluate both the reverse engineering process and the EXACT tool defined for supporting the analysis a comprehension of spreadsheet based artifacts in real industrial scenarios with case studies aimed at demonstrating their feasibility. These studies involved different end user actors involved in software development processes carried out in the automotive domain to evaluate whether the proposed solutions actually supported them in performing their assigned tasks.

The methodology defined to support the introduction of SPL was also evaluated during this project. The proposed methodology was exploited to define an SPL infrastructure that was adopted in some decoupled project for the development of IPC embedded software related to different vehicle models. Thanks to the introduction of this infrastructure we were able to achieve an improvement in the considered software development process in terms of required man-months.

The proposed tool integration approach was applied in the context of testing development processes carry out in the automotive domain. The application of our approach allowed to correctly coordinate the entire software testing process, by integrating the different tools needed for its execution. We were able to perform an initial case study to evaluate the improvements reached by applying the approach. An initial case study related to the testing activities of different Software Components developed by an FCA Software Factory allowed us to show improvements on the visibility and acceptability of the considered processes thanks to the adoption of our proposed approach.

I am performing an empirical evaluation of the proposed methodology for the automatic documentation of software processes enacted with the use of ALM systems. Case studies with software developer involved in

embedded software development in different domains have been planned and are going to be executed to evaluate the feasibility and effectiveness of the proposed automatic documentation approach.

I was also involved with the ReVERSE Group in activities for defining more effective approaches in the context of software testing. More in detail, I supported the definition of an infrastructure for testing the lifecycle of mobile applications (for the Android ecosystem) exploiting automated exploration techniques. Moreover, I am collaborating in the definition of a hybrid testing techniques that combines automated and capture and replay testing approaches. The technique aims at thoroughly exploring and testing mobile apps.

I was also involved in the definition of the Thing-in-the-Loop approach for supporting the Verification and Validation of Internet of Things (IOT) systems, focusing in particular on their safety. The approach was defined translating the xIL approaches for embedded systems verification and validation [11] toward the IoT domain. The approach exploits context models in order to safely test also failure and harmful scenarios, that will be dangerous to be tested on the field. Starting from models and scenarios representing the structure and behaviors of the IoT system as well as models of its context our approach generates appropriate test cases that are executed in accordance with Model-in-the-Loop, Software-in-the-Loop and Hardware-in-the-Loop techniques. In order to show the feasibility of the approach we applied it to a Smart Mobility case study.

During this third year, I was also involved in teaching activities, helding seminars during ***Software Engineering*** ("Ingegneria del Software") and ***Design and Development of Software Systems*** ("Progettazione e Sviluppo di Sistemi Software") courses. Moreover, I supported students in their thesis activities and I performed tutorship activities with first year students, supporting them in the preparation of first year courses.

Products

The research activities carried out during the three years of the PhD program led to the following products:

Published Papers

[C7] Domenico Amalfitano, ***Vincenzo De Simone***, Anna Rita Fasolino, Stefano Scala: **Improving traceability management through tool integration: an experience in the automotive domain.** 10th International Conference on Software and System Process ICSSP 2017: 5-14. ACM.

[J1] Domenico Amalfitano, ***Vincenzo De Simone***, Anna Rita Fasolino, Porfirio Tramontana: **EXACT: A tool for comprehending VBA-based Excel spreadsheet applications.** Journal of Software: Evolution and Process 28(6): 483-505 (2016). Wiley.

[C6] Domenico Amalfitano, ***Vincenzo De Simone***, Anna Rita Fasolino, Mario Lubrano, Stefano Scala: **Introducing Software Product Lines in Model-Based Design Processes: An Industrial Experience.** 13th Working IEEE/IFIP Conference on Software Architecture WICSA 2016: 287-290

[C5] Domenico Amalfitano, Nicola Amatucci, ***Vincenzo De Simone***, Anna Rita Fasolino, Porfirio Tramontana: **Toward Reverse Engineering of VBA Based Excel Spreadsheet Applications.** 2nd International Workshop on Software Engineering Methods in Spreadsheets SEMS@ICSE 2015: 30-31

[C4] Domenico Amalfitano, ***Vincenzo De Simone***, Anna Rita Fasolino, Vincenzo Riccio: **Comparing Model Coverage and Code Coverage in Model Driven Testing: An Exploratory Study.** 30th IEEE/ACM International Conference on Automated Software Engineering Workshops 2015: 70-73

[C3] Domenico Amalfitano, Anna Rita Fasolino, Porfirio Tramontana, ***Vincenzo De Simone***, Giancarlo Di Mare, Stefano Scala: **A Reverse Engineering Process for Inferring Data Models from Spreadsheet-based Information Systems: An Automotive Industrial Experience.** Communications in Computer and Information Science 178 - DATA 2014 Revised Selected Papers: 136-153. Springer.

[C2] Domenico Amalfitano, Anna Rita Fasolino, Porfirio Tramontana, ***Vincenzo De Simone***, Giancarlo Di Mare, Stefano Scala: **Information Extraction from Legacy Spreadsheet-based Information System - An Experience in the Automotive Context.** 3rd International Conference on Data Management Technologies and Applications DATA 2014 : 389-398

Accepted Papers

[C8] Domenico Amalfitano, Nicola Amatucci, ***Vincenzo De Simone***, Vincenzo Riccio, Anna Rita Fasolino: **Towards a Thing-In-the-Loop approach for the verification and validation of IoT systems.** 1st ACM Workshop on the Internet of Safe Things

Submitted Papers

[S1] Domenico Amalfitano, ***Vincenzo De Simone***, Stefano Scala, Anna Rita Fasolino: **Using Application Lifecycle Management and Model Driven Engineering for supporting Questionnaire-Based Gap Analysis Processes.** Submitted to the Information and Software Technology Journal. Elsevier.

[S2] Domenico Amalfitano, Nicola Amatucci, Vincenzo De Simone, Vincenzo Riccio, Anna Rita Fasolino: **Is This the Lifecycle We Really Want? An Automated Black-Box Testing Approach for Android Activities.** Submitted to the 11th IEEE Conference on Software Testing, Validation and Verification

In Preparation Papers

[P1] Domenico Amalfitano, ***Vincenzo De Simone***, Anna Rita Fasolino: **Exploiting Model Driven Engineering for the Automatic Documentation of Software Processes enacted through Application Lifecycle Management Systems.**

[P2] Domenico Amalfitano, Nicola Amatucci, ***Vincenzo De Simone***, Vincenzo Riccio, Anna Rita Fasolino: **Combining automatic exploration of mobile apps behavior with capture and replay through Machine Learning.**

[P3] Domenico Amalfitano, Anna Rita Fasolino, ***Vincenzo De Simone***, Stefano Scala: **Extended version of Improving Traceability Management through Tool Integration: an Experience in the Automotive Domain.** To be submitted at **Journal of Software: Evolution and Process** – Special Issue: Revised Selected Papers of ICSSP 2017. Wiley.

Conferences and Seminars

During the third year of the PhD program, I attended the **13th TAROT Summer School on Software Testing, Verification & Validation** (26-30 June, Naples, Italy) organized by the Department of Electric Engineering and Information Technologies (DIETI) where I presented the work: "**Improving Traceability Management through Tool Integration: a case study for Software Testing Processes"**.

Moreover, I attended the **10th International Conference on Software and System Processes** (5-7 July, Paris, France) where I presented the paper: "**Improving traceability management through tool integration: an experience in the automotive domain**".

Credits Summary

Student: Vincenzo De Simone
vincenzo.desimone2@unina.it

Tutor: Anna Rita Fasolino
fasolino@unina.it

Cycle XXX

	Credits year 1						Credits year 2						Credits year 3						
	1 bimonth	2 bimonth	3 bimonth	4 bimonth	5 bimonth	6 bimonth	1 bimonth	2 bimonth	3 bimonth	4 bimonth	5 bimonth	6 bimonth	1 bimonth	2 bimonth	3 bimonth	4 bimonth	5 bimonth	6 bimonth	**Total**
Modules		5	3	3	9	0	0	3	3	8	0	0	0	0	4	0	0	0	**38**
Seminars	1,5	1,2	2,5	4,8	0	0	3,6	0,8	0,4	2,2	0	0,5	0	0,6	0	3	2	0	**23**
Research	8,5	3,8	4,5	2,2	4	10	6,4	6,2	6,6	1,8	8	9,5	10	9,4	6	7	8	10	**122**
	10	10	10	10	13	10	10	10	10	12	8	10	10	10	10	10	10	10	**183**

Bibliography

[1] A. Fuggetta, «Software Process: a roadmap,» in *Proceedings of the Conference on The Future of Software Engineering*, Limerick, Ireland, 2000.

[2] A. Fuggetta e E. Di Nitto, «Software Process,» in *Future of Software Engineering, {FOSE} 2014*, Hyderabad, India, 2014.

[3] D. C. Schmidt, «Model-driven engineering,» *Computer,* vol. 39, pp. 25-31, 2006.

[4] D. Chappell, «What is application lifecycle management?,» Microsoft Press, 2008.

[5] S. Thiel e A. Hein, «Modeling and using product line variability in automotive systems,» *IEEE Software,* vol. 19, pp. 66-72, 2002.

[6] A. Apel, D. Batory, C. Kstner e G. Saake, Feature-Oriented Software Product Lines: Concepts and Implementation, Berlin: Springer Publishing Company, 2013.

[7] P. Clements e L. Northrop, Software Product Lines: Practices and Patterns, Addison-Wesley Professional, 2001.

[8] P. Runeson e M. Host, «Guidelines for conducting and reporting case study research in software engineering,» *Empirical Software Engineering,* vol. 14, 2009.

[9] C. B. Seaman, «Qualitative Methods,» *Guide to Advanced Empirical Software Engineering,* 2008.

[10] ISO, «ISO 26262:2011 Road vehicles - Functional safety,» ISO, 2011.

[11] C. M. C. S. N. N. L. Z. a. S. C. G. Tibba, «Testing automotive embedded systems under X-in-the-loop setups,» in *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, 2016.

[12] L. Garcìa-Borgonon, M. A. Barcelona, J. A. Garcìa, M. Alba e M. Escalona, «Software process modeling languages: A systematic literature review,» *Information & Software Technology,* vol. 56, n. 2, pp. 103--116, 2014.

[13] Object Management Group (OMG), «SPEM 2.0 Formal Specification,» 01 04 2008. [Online]. Available: http://www.omg.org/spec/SPEM/2.0/. [Consultato il giorno 31 10 2015].

[14] R. Bendraou, J. Jezequel, M.-P. Gervais e X. Blanc, «A Comparison of Six UML-Based Languages for Software Process Modeling,» *IEEE Transactions on Software Engineering, 36-5,* pp. 662-675, 2010.

[15] H. Lacheiner e R. Ramler, «Application Lifecycle Management as Infrastructure for Software Process

Improvement and Evolution: Experience and Insights from Industry,» in *Software Engineering and Advanced Applications (SEAA), 2011 37th EUROMICRO Conference on*, Oulu, Finland, 2011.