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 – Second 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 second year of the PhD Program, I attended one Occasionally Provided Course, two ad-Hoc Courses and a Doctoral School. ***Table 1*** reports the information about the attended courses.

Table 1 – List of attended courses

Course type	Course Name	Provided by	Date	Number of Credits
Occasionally Provided	**Specification and Verification of Multi-Agent Systems**	Prof. Wojtek Jamroga (Polish Academy of Sciences)	24/02/2016	3
ad-Hoc	**Communicating and disseminating your research work**	Prof.Mo Mansuri (Stevens Institute of Technology)	03/03/2016	3
	Scientific Writing	Prof. Paolo Russo	24/05/2016	5
Doctoral School	**International Summer School on Software Engineering**	Proff. Harman, Brambilla, Fraser, Canfora, Antoniol, De Lucia	16/06/2016	3

Seminars

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

Table 2 - List of attended seminars

Title	Presenter	Date	Credits
Memory Technologies for Android Based systems	Luca Porzio	10/11/15	0,4
Test and Diagnosis of Integrated Circuits	Alberto Bosio	17/11/15	0,6
Test and Diagnosis of Integrated Circuits	Alberto Bosio	18/11/15	0,6
Hardware Security and Trust	Giorgio Di Natale	19/11/15	0,6
Hardware Security and Trust	Giorgio Di Natale	20/11/15	0,6

Model Based Testing	Prof. Ana Paiva	23/11/15	0,4
Pattern Based GUI Testing	Prof. Ana Paiva	25/11/15	0,4
Adversarial Testing of Protocol Implementations	Prof. Cristina Nita Rotaru	23/02/16	0,4
Programmable Network Conjugations	Dr. Roberto Bifulco	26/02/16	0,4
Challenging real-time measurement systems for immersive life-size augmented environment	Dr. Giovanni Caturano	29/04/16	0,4
Internet: la dimensione Immateriale dell'esistenza	Stefano Quintarelli	19/05/16	0,4
Methodologies for Embedded Software Validation	Tornese, D'Avino, Isernia, Oppido	28/04/15	0,2
La Protezione Brevettuale: Opportunità, Procedure, Casi di Studio	B. Bjola, M. Maremonti	30/09/16	0.5

Research Activity

Supporting the management and improvements of industrial software processes

The main objective of the proposed research activity, continuing the one carried out last year related to software processes, is to define methods, techniques and tools aimed at supporting the management and the improvements of software processes.

Nowadays, software has acquired a great relevance in our society. Different domains relay in same 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 supporting software processes of great interest for the software engineering community.

An effective management of software processes requires methods, techniques and tools enabling an easy definition of its static and dynamic aspects: the roles of software engineers involved in the process, the artifacts that should be 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 all the different development activities.

In this second year, I focused my attention on proposing approaches and techniques as well as tools to support the management and the improvement of software processes that can be also applied in real industrial contexts.

The quality of software products is affected by the processes followed to produce them. For this reason, *Standards* 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. 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, a new approach aimed at supporting companies in effectively understanding what are the gaps with respect to the requirements imposed by a Standard they are willing to introduce and to plan the needed improvements has been defined. The approach defines both the process to be followed and the needed tool support in order to carry out this kind of task. The approach we defined exploits the concepts defined by the Model Driven Engineering paradigm [3]. The approach 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 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.

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 this year is 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 [4]. 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 [5]. According to Clements and Northrop [6], "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 this year, we 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.

Moreover, during this year I investigated the problem of the improvement of the quality attributes related to enacted software processes. To this aim, we defined and evaluated a new approach aimed at improving the visibility, the acceptability and controllability of software processes. To reach improvements of these quality attributes we 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, call connectors and data connectors was defined. Our proposed approach to handle integration of different tasks in the context of software

development processes exploits the features offered by Application Lifecycle Management (ALM) [7] 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 technique and a tool aimed at automatically producing documentation for Software process management tool (as an example ALM projects) supporting the enactment of software development process was defined. The automatically produced documentation was defined in order 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 my second year, I was able to apply some of the methodologies and techniques proposed 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 is aimed at proposing new methodologies to support the development of safe vehicles. 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 [8] 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.

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.

We are planning to execute an empirical evaluation of our 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 are being planned to evaluate the feasibility and effectiveness of the proposed automatic documentation approach.

I also supported the ReVERSE Group in activities related to software robustness testing. More in detail, I supported the definition of an infrastructure for executing robustness test of mobile applications (for the Android ecosystem) exploiting input perturbation techniques. The aim of the infrastructure is to test the robustness of Android applications

During this second 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.

Products

Accepted Papers

P1. Domenico Amalfitano, *Vincenzo De Simone*, Anna Rita Fasolino and Vincenzo Riccio: "**Comparing model coverage and code coverage in Model Driven Testing: an exploratory study"**. Proceedings of the 6th International Workshop on Testing Techniques for Event Based Software (TESTBEDS) co-located with 10th IEEE/ACM International Conference on Automated Software Engineering (ASE 2015)

P2. Domenico Amalfitano, *Vincenzo De Simone*, Anna Rita Fasolino, Porfirio Tramontana, 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, Volume 178 - chapter 9, Springer

P3. 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, Wiley

P4. Domenico Amalfitano, *Vincenzo De Simone*, Anna Rita Fasolino and Vincenzo Riccio: "**Introducing Software Product Lines in Model-Based Design Processes: An Industrial Experience"**. Proceedings of the 13th Working IEEE/IFIP Conference on Software Architecture, WICSA 2016

Submitted Papers

-

In Preparation Papers

P5. Domenico Amalfitano, *Vincenzo De Simone*, Anna Rita Fasolino: **"A questionnaire-based and ALM-supported approach for the gap analysis of quality standards"**
P6. D. Amalfitano, V. De Simone, A. R. Fasolino and V. Riccio "**aRTETECA - Robustness TEsting TEChnique for Android: an Exploratory Study**"
P7. aRTETECA: Tool Demonstration (ICST 2017)

Conferences and Seminars

I attended the **13th Working IEEE/IFIP Conference on Software Architecture** (8 April, Venice, Italy) where I presented the paper: "**Introducing Software Product Lines in Model-Based Design Processes: An Industrial Experience**".

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	2	3	4	5	6			1	2	3	4	5	6			1	2	3	4	5	6		
	Estimated	bimonth	bimonth	bimonth	bimonth	bimonth	bimonth	Summary	Estimated	bimonth	bimonth	bimonth	bimonth	bimonth	bimonth	Summary	Estimated	bimonth	bimonth	bimonth	bimonth	bimonth	bimonth	Summary	Total
Modules	20		5	3	3	9		**20**	10	0	3	3	8			**14**	3							**0**	**34**
Seminars	10	1,5	1,2	2,5	4,8			**10**	5	3,6	0,8	0,4	2,2		0,5	**7,5**	2							**0**	**18**
Research	30	8,5	3,8	4,5	2,2	1	10	**30**	45	6,4	6,2	6,6	1,8	8	9,5	**39**	55							**0**	**72**
	60	10	10	10	10	10	10	**60**	60	10	10	10	12	8	10	**60**	60	0	0	0	0	0	0	**0**	**123**

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

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

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

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

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

[9] 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.

[10] 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].

[11] 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.

[12] 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.