



PhD in Information Technology and Electrical Engineering

Università degli Studi di Napoli Federico II

PhD Student: Ugo Giordano

XXIX Cycle

Training and Research Activities Report – First Year

Tutor: Stefano Russo



1. Information

PhD Student: Ugo Giordano

MS title: Computer Engineering – University of Naples Federico II

PhD cycle: XXIX – ITEE – University of Naples Federico II

Fellowship: PON “Display” 2007

Tutor: prof. Stefano Russo

2. Study and Training activities

a. Courses

Lecture/Activity	Type	Date	CFU	Certification
Euro Progettazione	Ad hoc module	14/11/14	3	Yes
Sistemi Real Time	MS Module	11/07/14	6	Yes
Sistemi Distribuiti	MS Module	12/02/15	6	Yes
Impianti di Elaborazione	MS Module	30/01/05	9	Yes
Total			24	

b. Seminars and other

Lecture/Activity	Type	Date	CFU	Certification
Reliability and Availability Modeling in Practice	Seminar	05/11/14	0.4	Yes
Capacity Planning for Infrastructure-as-a-Service Cloud	Seminar	07/11/14	0.4	Yes
Methods and tools for smart device integration and simulation	Seminar	20/11/14	0.4	Yes
UML Profiles for the specification of non functional properties of software systems	Seminar	26/11/14	0.5	Yes
Verification of Mobile Agents in Partially Known Environments	Seminar	27/11/14	0.4	Yes
Site Reliability Engineering at Google	Seminar	27/11/14	0.6	Yes
Applications for software for development: types, interactions and continuous integration	Seminar	16/01/15	0.4	Yes
Joint location and design optimization for resource in software defined virtual networks	Seminar	21/01/15	0.8	Yes
Efficient service distribution in next generation cloud networks	Seminar	10/02/15	0.8	Yes
Three core issues for the Internet: things, security and economics	Seminar	19-20/02/15	2	Yes
11th International School on Software Engineering	Doctoral School	30/06 to 3/07/14	3	Yes
Convention PON Recas	Convention	16/12/14	1.6	
Total			11.3	

3. Research activity

Titolo sintetico: Software Defects Analysis Techniques for Critical Systems

Descrizione attività di ricerca:

Nei sistemi critici, cioè in quei sistemi il cui fallimento può portare a perdite elevate in termini di vite umane e/o denaro, è di fondamentale importanza il rispetto dei requisiti di affidabilità e sicurezza su cui si fonda. Questo si riflette sulla progettazione di tali sistemi che risulta essere alquanto onerosa anche a causa della presenza di standard imposti dai domini di applicazione e dal processo di sviluppo.

Lavori scientifici precedenti hanno evidenziato come l'analisi del codice [1, 6] e dei difetti software [10], applicata a questi sistemi, può risultare di fondamentale importanza nella verifica della qualità del codice prodotto, nonché per apportare netti miglioramenti al processo di sviluppo software. Dai risultati dell'analisi statica è possibile evincere quali sono i componenti più critici del sistema, in termini di numero di regole di programmazione violate, e quali sono i tipi di regole maggiormente violate dai programmatori.

Tuttavia, un problema comune riscontrato con gli approcci di analisi statica è quello dovuto alla straordinaria quantità d'informazione fornite, che complica l'individuazione delle regole infrante più importanti e dei componenti più critici del sistema. Questo risulta essere particolarmente importante nei sistemi *mission critical* di grandi dimensioni, dove la corretta ripartizione dell'effort [7, 8, 9] da dedicare ai processi di *detection* e *fixing* difetti software, può controllare in buona parte il trade-off tra costo e qualità dei sistemi. È perciò necessario identificare delle tecniche intelligenti, che tengano conto delle informazioni fornite dall'analisi del codice, e che analizzino il livello di criticità e l'influenza che un componente e/o una regola di programmazione possiede durante il processo di *detection* e *fixing*.

Nel primo anno il dottorando ha condotto un'analisi dettagliata delle seguenti tematiche:

- Approcci di analisi statica del codice [1, 2, 3, 4, 5, 6]
- Approcci che fanno uso di defect prediction [11, 12, 13, 14, 15]
- Approcci che combinano l'analisi statica e la defect prediction [16, 17, 18, 19]

ed è stato successivamente proposto un primo possibile approccio per la risoluzione della problematica precedentemente descritta. In particolare il dottorando ha proposto un metodo che, in funzione dell'obiettivo di qualità che si vuole raggiungere, fornisce un'indicazione diretta sulla priorità con la quale un componente e/o una regola di programmazione dovrebbe essere gestito per realizzare un'efficiente *code sanitization*, riducendo al tempo stesso i costi del processo di sviluppo.

I risultati preliminari di tale lavoro, hanno condotto alla stesura di un paper dal titolo "*Prioritizing correction of static analysis infringements for cost-effective code sanitization*" e saranno presentati in occasione della 37th "*International Conference on Software Engineering*", che si terrà a Firenze nel Maggio prossimo.

Inoltre, nel contesto di una collaborazione nata tra l'università e SELEX ES¹, il dottorando ha proposto uno strumento che, a partire dall'analisi delle informazioni strutturate contenute nei defect-report redatti dai team di testing, fornisca dei feedback al processo decisionale aziendale con l'obiettivo di migliorare uno tra i fattori di qualità, costo e tempo o il trade-off tra di essi.

Per quanto riguarda gli sviluppi futuri, si prevede l'estensione dell'approccio proposto mediante l'introduzione di una combinazione tra le tecniche di analisi statica del codice e quelle di predizione dei difetti software.

In aggiunta, nella parte restante del primo anno il dottorando si è occupato di una differente tematica, che nell'ambito di una collaborazione tra l'università e Huawei², l'ha portato ad analizzare lo stato dell'arte relativo alla gestione ottimale delle risorse nelle infrastrutture di cloud computing

¹ Selex ES, una società Finmeccanica, è leader internazionale nella realizzazione di sistemi elettronici e di information technology per la difesa, l'aerospazio, le infrastrutture, la protezione e sicurezza del territorio, nonché nella creazione di soluzioni smart sostenibili.

² Huawei (ufficialmente Huawei Technologies Co. Ltd.) è una società cinese leader globale nello sviluppo, produzione e commercializzazione di prodotti, sistemi e soluzioni di rete e telecomunicazioni.

[20, 21, 22, 23], con l'obiettivo di individuare le diverse tecniche adottate nella gestione dell'overload [24, 25] di tali infrastrutture fisiche.

Nell'ambito del secondo anno invece, il dottorando si focalizzerà sulla tematica relativa alla virtualizzazione delle funzionalità (NFV) e delle infrastrutture (SDN) di rete nelle architetture cloud, con l'obiettivo di proporre delle possibili soluzioni al gravoso problema di overload che le attuali reti di interconnessione stanno vivendo, anche a causa dell'aumento esponenziale di device che esse devono supportare.

4. Products

a. Publications

Conference Paper

Title: Prioritizing correction of static analysis infringements for cost-effective code sanitization.

Conference: 37th “*International Conference on Software Engineering*” (ICSE 2015 – IEEE), Workshops on “*Software Engineering Research and Industrial Practice*” (SER&IP 2015), Florence, May 16th to 24th.

ABSTRACT

Static analysis is a widely adopted technique in the industrial development of software systems. It allows to automatically check for code compliance with respect to predefined programming rules. When applied to large software systems, sanitizing the code in an efficient way requires a careful guidance, as a high number of (more or less relevant) rule infringements can result from the analysis. We report the results of a static analysis study conducted on several industrial software systems developed by SELEX ES, a large manufacturer of software-intensive mission-critical systems. We analyzed results on a set of 156 software components developed in SELEX ES; based on them, we developed and experimented an approach to prioritize components and violated rules to correct for a cost-effective code sanitization. Results highlight the benefits that can be achieved in terms of quality targets and incurred cost.

5. Conference and Seminars

a. Conference

25th “*International Symposium on software Reliability Engineering*”, Naples, November 3th to 6th, 2014. Number of papers: 113.

11th “*International School on software Engineering*”, Salerno, from June 30th to July 3th, 2014.

b. Seminars and convention

“Reliability and Availability Modeling in Practice”, Naples, November 5th, 2014.

“Capacity Planning for Infrastructure-as-a-Service Cloud”, Naples, November 7th, 2014.

“Methods and tools for smart device integration and simulation”, Naples, November 20th, 2014.

“UML Profiles for the specification of non functional properties of software systems”, Naples, November 26th, 2014.

“Verification of Mobile Agents in Partially Known Environments”, Naples, November 27th, 2014.

“Site Reliability Engineering at Google”, Naples, November 27th, 2014.

“Applications for software for development: types, interactions and continuous integration”, Naples, January 16th, 2015.

“Joint location and design optimization for resource in software defined virtual networks”, Naples, January 21th, 2015.

“Efficient service distribution in next generation cloud networks”, Naples, February 10th, 2015.

“Three core issues for the Internet: things, security and economics”, Naples, February 19th to 20th, 2015.

Convention *“PON Recas”*, Naples, 16th December, 2014.

6. CS summary

	Credits year 1							Summary	Credits year 2		Credits year 3		
	1	2	3	4	5	6	Estimated		Check	Estimated	Check		
Modules	24	0	0	6	0	3	15	24	20-40	13	10-20	5	0-10
Seminars	6	0	0	3	0	2.7	2	7.7	5-10	8	5-10	5	0-10
Research	35	8	8	2	9	8	2	37	10-35	45	30-45	60	40-60
	65	8	8	11	9	13.7	19	68.7		66		70	

7. Bibliography

- [1] J. Zheng, L. Williams, N. Nagappan, W. Snipes, J. Hudepohl, and M. Vouk, "On the value of static analysis for fault detection in software," *Software Engineering, IEEE Transactions on*, vol. 32, no. 4, pp. 240–253, April 2006.
- [2] F. Wedyan, D. Alrmuny, and J. Bieman, "The effectiveness of automated static analysis tools for fault detection and refactoring prediction," in *Software Testing Verification and Validation, 2009. ICST '09. International Conference on*, April 2009, pp. 141–150.
- [3] N. Nagappan and T. Ball, "Static analysis tools as early indicators of pre-release defect density," in *Software Engineering, 2005. ICSE 2005. Proceedings. 27th International Conference on*, May 2005, pp. 580–586.
- [4] C. Boogerd and L. Moonen, "Assessing the value of coding standards: An empirical study," in *Software Maintenance, 2008. ICSM 2008. IEEE International Conference on*, Sept 2008, pp. 277–286.
- [5] N. Nagappan, L. Williams, J. Hudepohl, W. Snipes, and M. Vouk, "Preliminary results on using static analysis tools for software inspection," in *Software Reliability Engineering, 2004. ISSRE 2004. 15th International Symposium on*, Nov 2004, pp. 429–439.
- [6] R. Plosch, H. Gruber, A. Hentschel, G. Pomberger, and S. Schiffer, "On the relation between external software quality and static code analysis," in *Software Engineering Workshop, 2008. SEW '08. 32nd Annual IEEE*, Oct 2008, pp. 169–174.
- [7] M. Lyu, S. Rangarajan, and A. van Moorsel, "Optimal allocation of test resources for software reliability growth modeling in software development," *IEEE Trans. on Reliability*, vol. 51, no. 2, pp. 336–347, 2002.
- [8] R. Pietrantuono, S. Russo, and K. Trivedi, "Software reliability and testing time allocation: An architecture-based approach," *Software Engineering, IEEE Transactions on*, vol. 36, no. 3, pp. 323–337, May 2010.
- [9] G. Carrozza, M. Faella, F. Fucci, R. Pietrantuono, and S. Russo, "Engineering air traffic control systems with a model-driven approach," *IEEE Software*, vol. 30, no. 3, pp. 42–48, 2013.
- [10] G. Carrozza, R. Pietrantuono, and S. Russo, "Defect analysis in mission – critical software systems: a detailed investigation," *J. Softw. Evol. and Proc.*, vol. 27, no. 1, pp. 22–49, 2014.
- [11] Hall, T.; Beecham, S.; Bowes, D.; Gray, D.; Counsell, S., "A Systematic Literature Review on Fault Prediction Performance in Software Engineering," *Software Engineering, IEEE Transactions on*, vol.38, no.6, pp.1276,1304, Nov.-Dec. 2012
- [12] Huihua Lu; Kocaguneli, E.; Cukic, B., "Defect Prediction between Software Versions with Active Learning and Dimensionality Reduction," *Software Reliability Engineering (ISSRE), 2014 IEEE 25th International Symposium on*, vol., no., pp.312,322, 3-6 Nov. 2014.
- [13] Chris Lewis, Zhongpeng Lin, Caitlin Sadowski, Xiaoyan Zhu, Rong Ou, and E. James Whitehead Jr.. 2013. "Does bug prediction support human developers? findings from a google case study," in *Proceedings of the 2013 International Conference on Software Engineering (ICSE '13)*. IEEE Press, Piscataway, NJ, USA, 372-381.
- [14] Zimmermann, T.; Premraj, R.; Zeller, A., "Predicting Defects for Eclipse," *Predictor Models in Software Engineering, 2007. PROMISE'07: ICSE Workshops 2007. International Workshop on*, vol., no., pp.9,9, 20-26 May 2007.
- [15] D'Ambrosio, M.; Lanza, M.; Robbes, R., "An extensive comparison of bug prediction approaches," *Mining Software Repositories (MSR), 2010 7th IEEE Working Conference on*, vol., no., pp.31,41, 2-3 May 2010.
- [16] Yuming Zhou; Leung, H., "Empirical Analysis of Object-Oriented Design Metrics for Predicting High and Low Severity Faults," *Software Engineering, IEEE Transactions on*, vol.32, no.10, pp.771,789, Oct. 2006.
- [17] Gray, D.; Bowes, D.; Davey, Neil; Sun, Yi; Christianson, Bruce, "Software defect prediction using static code metrics underestimates defect-proneness," *Neural Networks (IJCNN), The 2010 International Joint Conference*, pp.1,7, 18-23 July 2010.

- [18] T. Menzies, Z. Milton, B. Turhan, B. Cukic, Y. Jiang, and A. Bener, "Defect prediction from static code features: current results, limitations, new approaches," *Automated Software Engineering*, vol. 17, no. 4, pp. 375–407, 2010.
- [19] Moser, R.; Pedrycz, W.; Succi, G., "A comparative analysis of the efficiency of change metrics and static code attributes for defect prediction," *Software Engineering, 2008. ICSE '08. ACM/IEEE 30th International Conference on*, vol., no., pp.181,190, 10-18 May 2008.
- [20] N. Bobroff, A. Kochut, and K. Beaty. Dynamic placement of virtual machines for managing sla violations. In *Integrated Network Management, 2007. IM '07. 10th IFIP/IEEE International Symposium on*, pages 119–128, May 2007.
- [21] A. Quiroz, Hyunjoo Kim, M. Parashar, N. Gnanasambandam, and N. Sharma. Towards autonomic workload provisioning for enterprise grids and clouds. In *Grid Computing, 2009 10th IEEE/ACM International Conference on*, pages 50–57, Oct 2009.
- [22] Xiaoqiao Meng, Canturk Isci, Jeffrey Kephart, Li Zhang, Eric Bouillet, and Dimitrios Pendarakis. Efficient resource provisioning in compute clouds via vm multiplexing. In *Proceedings of the 7th International Conference on Autonomic Computing, ICAC '10*, pages 11–20, New York, NY, USA, 2010. ACM.
- [23] A. Beloglazov and R. Buyya. Managing overloaded hosts for dynamic consolidation of virtual machines in cloud data centers under quality of service constraints. *Parallel and Distributed Systems, IEEE Transactions on*, 24(7):1366–1379, July 2013.
- [24] Long Wang, R.A. Hosn, and Chunqiang Tang. Remediating overload in over-subscribed computing environments. In *Cloud Computing (CLOUD), 2012 IEEE 5th International Conference on*, pages 860–867, June 2012.
- [25] L. Tomas and J. Tordsson. An autonomic approach to risk-aware data center overbooking. *Cloud Computing, IEEE Transactions on*, 2(3):292–305, July 2014.