PhD in Information Technology and Electrical Engineering

Università degli Studi di Napoli Federico II

PhD Student: Antonio Guerriero

XXXIV Cycle

Training and Research Activities Report – First Year

Tutor: Stefano Russo – co-Tutor: Roberto Pietrantuono

1. Information

PhD candidate:	Antonio Guerriero (mat. DR993614)
Date of birth:	06/08/1992
Master Science title:	Master's degree in Computer Engineering (cum laude) on 25/05/2018, Università di Napoli Federico II
Doctoral Cycle:	XXXIV
Fellowship type:	UNINA
Tutor:	Prof. Stefano Russo
Co-tutor:	Prof. Roberto Pietrantuono
Year:	First
Publication:	R. Pietrantuono, S.Russo, A. Guerriero, Run-time Reliability Estimation of Microservice Architectures, Proc. of the 2018 IEEE International Symposium on Software Reliability Engineering (ISSRE), Memphis, TN, USA, Oct. 15-18, IEEE, 2018 Winner of "Best Research Paper Award"

My master thesis, titled "*Reliability Assessment of Microservice Architectures*", focused on proposing a technique called Microservice Adaptive Reliability Testing (MART), that exploits the information coming from monitoring and an advanced sampling algorithm to have an updated, accurate and efficient estimate of reliability of the Microservices Architecture under test.

2. Study and Training activities

In my first year I attended several Modules, Seminars, and two PhD schools.

Lecture/Activity	Type	Hours	Credits	Dates	Organizer	Certificate
An introduction to Blockchains	Seminar	2	0.4	03/12/2018	Università di Napoli Federico II	Yes
Multimedia Big data Management and Processing at UNINA	Seminar	2	0.4	06/12/2018	Università di Napoli Federico II	Yes
Types and levels of Computational Explanation in AI: A Dual Process Proposal	Seminar	2	0.4	14/11/2018	Università di Napoli Federico II	Yes
CiberConflitti: sicurezza informatica, difesa, stabilità internazionale e diritto umanitario	Ad hoc module	4	0.8	28/11/2018	Università di Napoli Federico II	Yes
How to publish a scientific paper	Ad hoc module	2	0.4	26/11/2018	Università di Napoli Federico II	Yes
L'Accademia delle Startup _ Le Startup dell'Accademia	Ad hoc module	5	0.8	20/12/2018	Università di Napoli Federico II	Yes*
Bitcoin e Blockchain oltre l'hype	Seminar	3	0.6	18/01/2019	Università di Napoli Federico II	Yes
Mathematics for Machine Learning and Optimization and classification	Seminar	2	0.4	05/02/2019	Università di Napoli Federico II	Yes
Big Data	Ad hoc module		3	26-27/02/2019 and 4-5-6/03/2019	Università di Napoli Federico II	Yes

Data Mining	Master Science Course (ING-INF/05)		6	1th Semester 2018/2019	Università di Napoli Federico II	Yes
Advanced techniques for software robustness and security testing	Ad hoc module		3	16-30/01/2019 and 6-13-20/02/2019 and 2/04/2019	Università di Napoli Federico II	Yes
IEEExplore Training and Authorship workshop	Ad hoc module	2	0.5	04/04/2019	Università di Napoli Federico II	Yes
Recurrent Networks	Seminar	2	0.4	17/05/2019	Università di Napoli Federico II	Yes
Neural Networks	Seminar	2	0.4	13/05/2019	Università di Napoli Federico II	Yes
Deep Learning in MATLAB	Seminar	3	0.6	20/05/2019	Università di Napoli Federico II	Yes
Programmazione II	Master Science Course (ING-INF/05)		6	2nd Semester 2018/2019	Università di Napoli Federico II	Yes
Applying semi-supervised learning to app store analysis	Seminar	1	0.2	04/07/2019	Università di Napoli Federico II	Yes
Comparison of Techniques for Dealing with Imbalance in software defect prediction	Seminar	1	0.2	12/07/2019	Università di Napoli Federico II	Yes
International Summer School on Software engineering	Doctoral School		3	From 17 to 21/06/2019	Università degli Studi di Salerno	Yes
Laser Sumer School on Software Engineering	Doctoral School	43	8.6*	From 1 to 9/06/2019	LASER foundation	Yes
Facilitating Programming for Data Science vis DSLs and Machine Learning	Seminar	1	0.2	19/08/2019	**Chinese University of Hong Kong**	Yes
Methods for Explainable Machine Learning	Ad hoc module		2.4	21-21-24-27-29-31/05/2019	Università di Napoli Federico II	Yes

* The certification provided by the LASER foundation reports the amount of hours spent during the doctoral school. The credits are therefore calculated as *0.2*hours*.

	Year 1								Year 2	Year 3		
		1	2	3	4	5	6					
	Estimated	bimonth	bimonth	bimonth	bimonth	bimonth	bimonth	Summary	Estimated	Estimated	Total	Check
Modules	20	1.2	0.8	12.5	9.0		2.4	25.9	5	0	30.9	30-70
Seminars	10	1.2	1.0		10.0	0.4	0.2	12.8	0	0	12.8	10-30
Research	35	3.0	6.0	6.0	3.0	9.0	10.0	37.0	45	60	142	80-140
	65	5.4	7.8	18.5	22.0	9.4	12.6	75.7	50	60	185.7	180

3. Research activity

During the first year my research activities was focused on *software testing* domain. In particular, two main aspects were considered: software testing for quality assessment (*reliability*, *performance*, …) and software testing for *fault detection*.

3.1 Reliability Assessment of Microservices Architectures

Microservice architecture (MSA) is a software architectural style which is gaining popularity in many companies [1]. Netflix, eBay, Amazon, Twitter, PayPal and many other web-based services have evolved to this paradigm recently. MSA shifts traditional service-oriented architectures (SOA) from a share-as-much-as-you-can philosophy, focused on reuse, to a share-nothing philosophy, emphasizing strong service decoupling. MSA applications are built by architecting a set of services, each providing a well-defined and self-contained business capability and high independence from others. Combined with technologies such as RESTful protocols and containers and agile development practices such as DevOps [2], MSA features lightweight communication and independent and rapid service deployment. These characteristics promote scalability, flexibility, maintainability, prompt reaction to changes and failures, and frequent software releases.

In this research activity, I addressed the problem of assessing quantitatively the reliability of an MSA during the operational phase. This is a great concern for companies migrating towards MSA. While MSA is expected to favor seamless management of microservices' failures via fault tolerance means, what finally matters is the reliability of the overall MSA actually observed during operations (operational reliability). Operational reliability refers to the probability for a system to perform correctly on user demands: it is a user-perceived quality that depends not only on how much reliable a single microservice is, but also on how much it is actually used. A less reliable microservice may have small impact on the user perception if it is rarely stimulated during operations. Conversely, a highly reliable yet frequently invoked microservice may determine perceivable MSA unreliability, as the likelihood to observe a failure increases with usage. Decision makers – MSA stakeholders, as well as managers of development, testing and operations - need to be aware of how reliable is the MSA in the operating environment. This would drive strategic decisions, e.g. about effort allocation to maintenance or re-engineering activities.

Along this line, I then worked on developing a method for assessing the reliability of an MSA in its operational context called Enhanced Microservice Adaptive Reliability Testing (EMART). EMART extends the MART strategy introduced in my previous work, where the idea of using an adaptive testing algorithm to improve the reliability estimation of MSA applications was first presented [3]. The idea is to generalize MART by removing the upper bound on the number of tests to run. This generalization enables the proposed method to be used also in the scenario when the tester can afford more tests to the aim of achieving a high level of confidence in the estimate.

This research activity resulted in a submission to *Software Testing, Verification and Reliability* journal, currently under review.

3.2 Quality Assessment of Web Services

After about twenty years since their appearance, Web Services (WS) are probably still the most popular, standardized and technologically supported model for building distributed service-oriented applications. Two key Quality of Service (QoS) attributes for WS are reliability and performance. The engineering task of quantitatively evaluating the reliability and performance of a web service in its operating conditions represents an important problem to address in this research activity.

This engineering problem is relevant in several situations: examples are the selection of a WS, the search and QoS-based ranking of functionally equivalent WS, or the composition of atomic services into a new composite WS [4].

During the first year my supervisors and I collaborated about this topic with prof. Raffaela Mirandola. The collaboration resulted in a paper "A Hybrid Framework for Web Services Reliability and Performance Assessment" to be presented at the 1st "*Governing Adaptive and Unplanned Systems of Systems* (GAUSS) workshop" at *International Symposium on Software Reliability Engineering* (ISSRE) 2019. In this paper, we introduce WS-REPAS as a hybrid engineering framework combining a modelling approach, based on Discrete Time Markov Chains (DTMC), with monitoring and in vivo testing of the service under assessment. The combination of off-line analysis techniques with run-time mechanisms for continuous verification is advocated in [5]. In WS-REPAS, field data are gathered through passive observations of the service in operation, and used to continuously update parameters of the service DTMC model. When changes occur in the way the service is provisioned, or in the way it is used, an active strategy is triggered, which executes proper testing sessions. WS-REPAS is meant for practitioners, such as site operation engineers, who can thus leverage monitoring data, usually available from WS infrastructure facilities, to feed automated analysis and testing activities, to finally produce accurate estimates of the reliability and/or performance of a Web service.

3.3 Machine Learning applied to Regression Testing in Continuous Integration

Another important research collaboration was with Antonia Bertolino. The focus of this research activity was on applying Machine Learning strategies (which I studied during the above-mentioend course *Data Mining* and the LASER Doctoral School) to the problem of regression testing in a Continuous Integration (CI) software development environment.

CI is widely practiced in the software industry [6, 7] for its benefits in terms of release time and productivity [8]. Due to the frequent commits to the shared codebase, the cost of continuously performing regression testing escalates [9]. Regression testing has been investigated for decades [10]. However, most techniques for reducing its cost in traditional development cannot be applied at the large scale of modern CI practices [11]. Scalability issues are due not only to the size of codebases and test suites, but also to the dynamicity of such environments [9]. Hence, we assist today to renewed interest into lightweight test selection and prioritization (TS&P) techniques that can suit CI processes. An ideal TS&P technique for CI should be able to quickly identify a relevant subset of test cases that can safely and timely detect any potential regression introduced by the latest committed changes.

Prioritization is basically a ranking problem and naturally lends itself to be formulated as a learning task. We consider two very different learning strategies: Learning-to-rank (LTR) encompasses mainly supervised algorithms, proved useful in information retrieval and natural language processing [12]; an alternative strategy, suited in dynamic contexts, is reinforcement learning (RL). RL foresees an artificial agent that learns from the environment by observing its state and selects a proper action, either from a learned policy or by random exploration of possible actions.

The original contributions of this research activity are:

- Different ways to formulate TS&P as a ML problem;
- The first experimental study comparing performance of LTR and RTL test prioritization algorithms in CI.

This collaboration resulted in a submission to the 42nd *International Conference on Software Engineering* (ICSE 2020).

3.4 Continuous Reliability Testing in DevOps

A further activity, resulting again from the collaboration with Antonia Bertolino, focused on the application of continuous reliability assessment through testing in the context of DevOps.

DevOps [13] is an agile software development practice that stresses the seamless connection between development and operation stages. Despite its growing spread, there is no commonly agreed definition for *DevOps* [14, 15]. Some authors describe it as a cultural shift that IT organizations should undergo to remove technical or managerial barriers between the Dev and Ops teams, and let them collaborate under shared responsibilities [16]. Others focus on the technological capabilities that are necessary to enable such culture [15]; for instance, Bass et al. define DevOps as "a set of practices intended to reduce the time between committing a change to a system and the change being placed into normal production, while ensuring high quality" [13]. DevOps can be characterized as the intersection among the (formerly separated) scopes of software Development (Dev), Operation (Ops) and Quality Assurance (QA).

The idea at the basis of this research activity is that if development and operation are tightly connected, QA can leverage data coming from operation to guide reliability testing, in a virtuous DevOps cycling.

We proposed the DevOpRET approach for continuous reliability testing in DevOps. The approach leverages usage and failure information in operation to continuously refine operational-profile based testing. The proposed Ops-driven reliability testing is conceived as part of the acceptance testing stage before each next release to production.

The result of this collaboration is described in a submission of a paper to the *Journal of Software: Evolution and Process*, in September 2019, which is currently under review.

3.5 Testing of Artificial Intelligence

Testing is a pillar of Software Reliability Engineering (SRE). The new AI software development paradigm, makes it difficult to exploit common software testing techniques. By focusing on data rather than on the source code, this development paradigm leads to new challenges for software practitioners and, specifically, for software testers. The impossibility, in some application domains where AI is adopted, to establish whether a "choice/decision" made by a classifier is actually correct or not is one of the main challenges in applying testing to AI.

I'm currently targeting, in the context of AI-based systems, a known problem in software testing, namely the *test oracle* problem. The definition of an Oracle (Test Oracle) is a necessary task for testing. Testing in AI can be performed only when the correct decision for an arbitrary scenario to test (i.e., for arbitrary input data) is known a priori: however, in many systems it is hard to create the system's specification manually against which the system's behavior can be tested, because it is hard to specify the correct behavior for all the possible input data (e.g., for all possible images in image recognition).

In [17] the authors note that "Conventional software engineering processes and tools do not neatly apply: in particular, it is challenging to detect subtle errors, faults, defects or anomalies (henceforth "bugs") in the ML applications of interest because there is no reliable "test oracle" to indicate what the correct output should be for arbitrary input".

My aim is to propose a way to support the automatic construction of an oracle so as to safely evaluate the outcome of a test when a correct decision is not known. For the purpose, I defined a prototype based on three level, each one representing a different methodology: deductive method, inductive method applied to training data, and inductive method applied to system-generated data.

This research activity is conducted in collaboration with prof. Michael R. Lyu (chairman of Computer Science and Engineering Department at Chinese University of Hong Kong), and it is the main topic that I am addressing at the end of the first year of my PhD and that I am planning to study deeper during the second year.

4. Products

During the first year I co-authored different submissions along with my research group as well as with external researchers.

[P1]	Antonio Guerriero, Raffaella Mirandola, Roberto Pietrantuono, Stefano Russo. "*A Hybrid Framework for Web Services Reliability and Performance Assessment*". **1st International Workshop of Governing Adaptive and Unplanned Systems of Systems (GAUSS 2019)**, In: Proc. of the 2019 IEEE International Symposium on Software Reliability Engineering Workshops (ISSREW), pp. 185-192, IEEE. DOI: DOI 10.1109/ISSREW.2019.00070
[P2]	Stefano Russo, Roberto Pietrantuono, Antonio Guerriero, "*Testing Microservice Architectures for Operational Reliability*". Accepted for publication into the journal: **Software Testing, Verification and Reliability** (Wiley-Blackwell, ISSN:0960-0833).
[P3]	Stefano Russo, Roberto Pietrantuono, Antonio Guerriero, Antonia Bertolino, Breno Miranda, "*Learning-to-Rank vs Ranking-to-Learn: Strategies for Regression Testing in Continuous Integration*". Submitted to the **42nd International Conference on Software Engineering** (ICSE 2020), Seoul, Korea, May 2020. Status: under review.
[P4]	Stefano Russo, Roberto Pietrantuono, Antonio Guerriero, Antonia Bertolino, Breno Miranda, Guglielmo De Angelis, "*DevOpRET: Continuous Reliability Testing in DevOps*". Submitted to **Journal of Software: Evolution and Process** (Wiley-Blackwell, ISSN:2047-7481). Status: under review.

5. Conferences and Seminars

I participated to the following conferences and workshops:

Conference name	Place	Dates	Number of papers
The 30th IEEE International Symposium on Software Reliability Engineering (ISSRE)	Berlin, Germany	October 29-31, 2019	42
1st International Workshop of Governing Adaptive and Unplanned Systems of Systems (GAUSS)	Berlin, Germany	October 28, 2019	11

As the author, I **presented** the following paper:

Antonio Guerriero, Raffaella Mirandola, Roberto Pietrantuono, Stefano Russo. "A Hybrid Framework for Web Services Reliability and Performance Assessment". 1st International Workshop of Governing Adaptive and Unplanned Systems of Systems (GAUSS 2019), Berlin, Germany, October 28, 2019, pp. 185-192, IEEE (see [P1])

6. Activity abroad

Starting from the 2nd of September 2019, I am currently in Hong Kong at Chinese University of Hong Kong as visiting scholar, as member of prof. Michael R. Lyu's research group. During this stay, which is planned to finish in March 2020, I will have to opportunity to address the above-mentioned research topics, especially with reference to "testing of artificial intelligence", with one of the most highly recognized expert in the field of software reliability engineering.

7. Tutorship

During this first year I made some tutorship activity for two master students Alessandro Chillemi, for his master thesis in Software Engineering about "Una tecnica per la stima dell'affidabilità in DevOps", and Romolo De Vito, for his master thesis in Software Engineering about "Reinforced RT: una tecnica di Regression Testing in Continuous Integration".

In this first year, I have been teaching assistant for the course of Distributed Systems, a.a. 2018/2019, and Software Engineering a.a. 2018/2019.

8. References

[1] P. Di Francesco, I. Malavolta, and P. Lago, "Research on Architecting Microservices: Trends, Focus, and Potential for Industrial Adoption," in Int. Conf. on Software Architecture (ICSA). IEEE, 2017, pp. 21–30.

[2] H. Kang, M. Le, and S. Tao, "Container and Microservice Driven Design for Cloud Infrastructure DevOps," in Int. Conf. on Cloud Engineering (IC2E). IEEE, 2016, pp. 202–211.

[3] R. Pietrantuono, S. Russo, and A. Guerriero, "Run-time Reliability Estimation of Microservice Architectures," in 29th Int. Symp. on Software Reliability Engineering (ISSRE). IEEE, 2018, pp. 25–35.

[4] Y.Zhang and M.Lyu, QoS Prediction in Cloud and Service Computing: Approaches and Applications, ser. SpringerBriefs in Computer Science. Springer, 2017.

[5] A. Filieri, C. Ghezzi, R. Mirandola, and G. Tamburrelli, "Conquering Complexity via Seamless Integration of Design-Time and Run-Time Verification," in Conquering Complexity, M. Hinchey and L. Coyle, Eds. Springer, 2012, pp. 253–275.

[6] Mike McGarr, Dianne Marsh, and the Developer Productivity team. 2017. Towards true continuous integration: distributed repositories and dependencies. https://medium.com/netflix-techblog/towards-true-continuous-integration-distributed-repositories-and-dependencies-2a2e3108c051

[7] Ade Miller. 2008. A Hundred Days of Continuous Integration. In Agile 2008 Conference. IEEE, 289–293. https://doi.org/10.1109/Agile.2008.8

[8] Daniel Ståhl and Jan Bosch. 2013. Experienced benefits of continuous integration in industry software product development: A case study. In 12th IASTED International Conference on Software Engineering. ACTA Press, Calgary, 736–743.

[9] Atif Memon, Zebao Gao, Bao Nguyen, Sanjeev Dhanda, Eric Nickell, Rob Siem- borski, and John Micco. 2017. Taming Google-Scale Continuous Testing. In IEEE/ACM 39th International Conference on Software Engineering: Software Engi- neering in Practice Track (ICSE-SEIP). IEEE, 233–242.

[10] Shin Yoo and Mark Harman. 2012. Regression testing minimization, selection and prioritization: a survey. Software Testing, Verification and Reliability 22, 2 (2012), 67–120. https://doi.org/10.1002/stvr.430

[11] Sebastian Elbaum, Gregg Rothermel, and John Penix. 2014. Techniques for Improv- ing Regression Testing in Continuous Integration Development Environments. In 22nd ACM SIGSOFT International Symposium on Foundations of Software Engineering. ACM, New York, NY, 235–245. https://doi.org/10.1145/2635868.2635910

[12] Hang Li. 2011. Learning to Rank for Information Retrieval and Natural Language Processing. Morgan & Claypool, San Rafael, CA.

[13] Bass LJ, Weber IM, Zhu L., DevOps A Software Architect's Perspective. SEI series in software engineering. Addison-Wesley. 2015.

[14] Dyck A, Penners R, Lichter H., Towards definitions for release engineering and DevOps. In: IEEE/ACM 3rd International Workshop on Release Engineering (RELENG), IEEE; 2015: 3–3.

[15] Smeds J, Nybom K, Porres I., DevOps: A Definition and Perceived Adoption Impediments. In: Lassenius C, Dingsøyr T, Paasivaara M., eds. Agile Processes in Software Engineering and Extreme Programming, Springer International Publishing; 2015; Cham: 166–177.

[16] Walls M., Building a DevOps culture. "O'Reilly Media, Inc.". 2013.

[17] Christian Murphy, Gail Kaiser, and Marta Arias. 2008. An Approach to Software Testing of Machine Learning Applications.