Curriculum Vitae et Studiorum

Sandro Morasca

November 2, 2014

1 Education

1979 Graduation (Italian Diploma di Maturità Classica) from the "Liceo Classico Statale "A. Volta"" in Como, Italy, with a 60/60 mark.

1985 Graduation (Italian Laurea) *cum laude* in Electronic Engineering from the Politecnico di Milano, in Milan, Italy.

1991 PhD in Electrical Engineering of Information and Systems from the Politecnico di Milano, in Milan, Italy.

2 Academic Positions

Oct. 1991-Oct. 1993 Faculty Research Assistant at the Institute for Advanced Computer Studies of the University of Maryland at College Park, USA.

Oct. 1993 - Oct. 1998 Assistant Professor of Computer Science at the school of Engineering at Como, Italy and the Dipartimento di Elettronica e Informazione of Politecnico di Milano in Milano, Italy.

Nov. 1998 - Oct. 2000 Associate Professor of Computer Science at the school of Engineering at Como, Italy and the Dipartimento di Elettronica e Informazione of Politecnico di Milano in Milano, Italy.

Oct. 1999 - Feb. 2000 Visiting Scientist at the Department of Computer Science of the University of Maryland at College Park, USA.

Nov. 2000 - Apr. 2010 Professor of Computer Science at the school of Mathematical, Physical, and Natural Sciences and the Dipartimento di Scienze Chimiche, Fisiche e Matematiche and the Dipartimento di Scienze della Cultura, Politiche e dell'Informazione of the Università degli Studi dell'Insubria in Como, Italy.

May 2010 - present Professor of Computer Science at the school of Mathematical, Physical, and Natural Sciences and the Dipartimento di Scienze Biomediche, Informatiche e della Comunicazione of the Università degli Studi dell'Insubria in Varese, Italy.

3 Teaching

3.1 Teaching Assistant

Sandro Morasca was a teaching assistant for the following courses

- "Macchine per l'elaborazione dell'informazione" ("Machines for Information Processing"), held by prof. Renato Stefanelli at the Politecnico di Milano (1986/87)

- "Programmazione dei calcolatori elettronici" ("Computer Programming"), held by prof. Pierluigi Della Vigna at the Politecnico di Milano (1987/88, 1988/89, and 1989/90)
- "Introduzione agli algoritmi ed alla programmazione" ("Introduction to Algorithms and Programming"), held by prof. Pierluigi Della Vigna at the "Scuola a Fini Speciali del Politecnico di Milano Diretta in Informatica (Gestionale)" of the Politecnico di Milano at Como (1987/88, 1988/89, and 1989/90)
- "Ingegneria del Software" ("Software Engineering"), held by prof. Carlo Ghezzi at the Politecnico di Milano (1989/90)
- "Elementi di Informatica" ("Elements of Computer Science"), held by prof. Nello Scarabottolo at the Politecnico di Milano at Como (1990/91).

3.2 Assistant, Associate, and Full Professor

Sandro Morasca cooperated in the teaching of the following courses

- "Fondamenti di Informatica" ("Fundamentals of Computer Science") held by prof. Pierluigi Della Vigna at the Politecnico di Milano at Como (1993/94, 1994/95, 1995/96)
- "Fondamenti di Informatica 2" ("Fundamentals of Computer Science 2") held by prof. Giuseppe Pozzi at the Politecnico di Milano at Como (1997/98).

Sandro Morasca held the following courses

- "Fondamenti di Informatica" ("Fundamentals of Computer Science") at the Politecnico di Milano at Como (1996/97 - 2000/01)
- "Informatica A" ("Computer Science") at the Politecnico di Milano at Como (2000/01)
- "Cultura Tecnologica del Progetto" ("Technological Culture of Design") at the Politecnico di Milano at Como (2000/01)
- "Laboratorio di Programmazione I" ("Laboratory of Computer Programming") at the Università degli Studi dell'Insubria at Como (2000/01)
- "Programmazione e Laboratorio di Programmazione" ("Computer Programming and Laboratory") at the Università degli Studi dell'Insubria at Como (2001/02)
- "Progettazione di Software" ("Software Design") at the Università degli Studi dell'Insubria at Como (2001/02)
- "Progettazione del Software I con Laboratorio" ("Software Design I and Laboratory") at the Università degli Studi dell'Insubria at Como (2002/03 - 2010/2011)
- "Progettazione del Software II con Laboratorio" ("Software Design II and Laboratory") at the Università degli Studi dell'Insubria at Como (2002/03 - 2010/2011)
- "Ingegneria del Software I" ("Software Engineering I") at the Università degli Studi dell'Insubria at Como (2002/03 - 2009/2010)
- "Ingegneria del Software II" ("Software Engineering II") at the Università degli Studi dell'Insubria at Como (2002/03 - 2006/07 and 2009/10)
- "Computer Science" at the Libera Università Carlo Cattaneo in Castellanza (2002/03 - 2004/05)
- "Software Testing" at the Free University of Bozen, in Bozen, Italy (2008/09 - 2009/10)
- "Valutazione della Qualità del Software" ("Software Quality Evaluation") at the Università degli Studi dell'Insubria at Varese (2010/11)

- "Software Engineering Project" at the Free University of Bozen, in Bozen, Italy (2009/10)
- "Software Reliability and Testing" at the Free University of Bozen, in Bozen, Italy (2010/11 - present)
- "Progettazione del Software" ("Software Design") at the Università degli Studi dell'Insubria at Varese (2011/12 - present)
- "Architetture Orientate ai Servizi" ("Service Oriented Architectures") at the Università degli Studi dell'Insubria at Varese (2011/12)

Sandro Morasca is a co-author of book "Ingegneria del software - Progettazione, sviluppo e verifica" ("Software Engineering - Design, Development, and Verification," in Italian), published by Mondadori [32].

Sandro Morasca is a co-author of book "Fondamenti di Informatica - Dal problema al programma" ("Fundamentals of Computer Science - From the Problem to the Program," in Italian), published by ETAS libri [33].

He taught at continuing education courses at the Dipartimento di Elettronica of the Politecnico di Milano.

Sandro Morasca has held courses at important Italian companies, such as ENIDATA (Milan), Alenia (Rome), SIG (Palermo), SSGRR (L'Aquila), Telettra (Milan), Engineering (Rome), Elsag Bailey (Genua), Systech (Milan), ASL (Como) and international schools and research institutions, such as Università di Catania, Italy, University of Buenos Aires, Argentina, University of San Luis Potosì, Mexico, the Scuola Universitaria Professionale della Svizzera Italiana (SUPSI) in Manno, Switzerland, the Universidad de Castilla-La Mancha, the Norwegian University for Science and Technology in Trondheim, the University of Oslo, the Catholic University of Leuven, the Technische Universität München, and the Universidad Politécnica de Madrid.

4 Research

4.1 Empirical Studies in Software Engineering

An overview and an analysis of the state of the art in Software Engineering measurement are in [34].

4.1.1 Case Studies Based on Existing Measures

This activity began with Dr. Morasca's Laurea dissertation [118], which contained a critical evaluation and an empirical validation of counting strategies on Pascal programs [38]. In the framework of the PROTAGORA project, an empirical study was carried out at the Italian Ministry of Treasury to identify factors related to software productivity from a functional (i.e., in terms of Function Points) and an internal (i.e., in terms of code size measures) viewpoints [61]. By using the GQM paradigm, a few factors have been identified so as to help software managers make informed decisions. The relationship between IFPUG Function Points and COSMIC Function Points is investigated in [25]. The main result of this study is that, by using linear or piecewise linear models, it is possible to convert one kind of Function Points into the other kind with reasonable accuracy. An additional study [109] addresses the estimation of productivity defined in terms of IFPUG Function Points. The data analysis shows that estimation based on the closest analogues provides better results for most models, but very bad estimates in a few cases. To mitigate this behavior, the correction of regression toward the mean proved effective. A further study [27] shows that Basic Functional Components can be used to build models of effort that are equivalent, in terms of accuracy, to those based on Function Points. Also, simplified Function Points measures can be used as software development effort predictors in models that also use other requirements measures. So, the definition and measurement processes of Function Points can be dramatically simplified by taking into account a subset of the Base Functional Components used in the original definition of the measure, thus allowing for substantial savings in measurement effort, without sacrificing the accuracy of software development effort estimates.

Both IFPUG Function Points and COSMIC Function Points were applied to a number of situations that occur quite often in real-time and embedded [113, 31]. The results seem to indicate that, overall, COSMIC Function Points are better suited than traditional Function Points for measuring characteristic features of real-time and embedded systems. The results also provide practitioners with useful indications about the pros and cons of functional size measurement methods when confronted with specific features of real-time and embedded software.

Functional Size Measurement methods are widely used but have two major shortcomings: they require a complete and detailed knowledge of user requirements, and they involve relatively expensive and lengthy processes. UML is routinely used in the software industry to effectively describe software requirements in an incremental way, so UML models grow in detail and completeness through the requirements analysis phase. In this context, we have defined the characteristics of increasingly more refined UML requirements models that support increasingly more sophisticated hence presumably more accurate size estimation processes [111, 30]. Specifically, we consider the COSMIC method and three alternative processes (two of which are proposed in this line of research) to estimate COSMIC size measures that can be applied to UML diagrams at progressive stages of the requirements definition phase. Then, we checked the accuracy of the estimates by comparing the results obtained on a set of projects to the functional size values obtained with the standard COSMIC method. The analysis shows that it is possible to write increasingly more detailed and complete UML models of user requirements that provide the data required by COSMIC size estimation methods, which in turn yield increasingly more accurate size measure estimates of the modeled software. Initial estimates are based on simple models and are obtained quickly and with little effort. The estimates increase their accuracy as models grow in completeness and detail, i.e., as the requirements definition phase progresses.

We investigated whether IFPUG Function Point Analysis and the COSMIC method take directly into consideration the amount of data processing involved in a functional process and the extent to which the amount of data processing has an impact on the effort required for software development [117]. To this end, we considered a few applications that provide similar functionality, but require different amounts of data processing: these applications are then measured both via functional size measurement methods and via traditional size measures (such as LOC). A comparison of the obtained measures shows that differences among the applications are best captured by differences in LOC. It is likely that the actual size of an application that requires substantial amounts of data processing is not fully represented by functional size measures. Thus, not taking into account data processing dramatically limits the expressiveness of the size measures. Practitioner that use size measures for effort estimation should complement functional size measures with measures that quantify data processing, to get precise effort estimates.

An empirical investigation on the building of fault-proneness models for software applications based on static code measures is in [62, 20]. The models are built based on real-world data by means of Logistic Regression and cross-validation. Two empirical studies were carried out at the Digital Engineering Italia center in the framework of ESSI project CEMP. The final goal was to study the costs and benefits related to the introduction of a measurement program in an industrial context. To this end, a project under development [13] and a project under maintenance [9] were studied. The study resulted in a number of lessons learned that are largely reusable by other industrial organizations willing to introduce a GQM-based measurement program in their development environments.

The influencing factors of the reusability of a class in an object-oriented system were studied in [29]. Specifically, 29 measures for class size, coupling, and cohesion were used in a correlational study that involved 1710 classes taken from real-life systems. The main results show that the size and coupling attributes of a class have positive impacts on its reuse-proneness via inheritance and instantiation. The cohesion of a class has a negative impact on its inheritance reuse-proneness and a positive impact on its instantiation reuse-proneness.

References [9, 13, 20, 25, 29, 27, 30, 31, 38, 61, 62, 109, 111, 113, 117, 118]

4.1.2 Definition of Measures for Software Code and Early Artifacts in the Life Cycle

New software measures have been defined for concurrent programming. More specifically, a measure has been defined for concurrent languages that allows the evaluation of control flow complexity. This measure, based on Petri nets, is able to capture both sequential and concurrent characteristics of a program [45]. This work was later revised and extended in [53, 60], where measures are defined for a number of qualities of specifications written by means of Petri Nets.

New measures have been defined and experimentally validated for the evaluation of software high level design. These measures address several characteristics of high level design, such as cohesion and coupling. The measures, initially defined in [50], have been further refined, extended, theoretically validated according to the axioms in [6] and experimentally validated on industrial projects written in Ada at the Goddard Space Flight Center of NASA [14]. Measurement is further extended to the specification phase in [56], which contains the definition of measures for formal specifications written in TRIO+ and their validation on an industrial case study to predict the effort needed to develop a specification and the number of changes a specification undergoes. The study has provided predictive models of fairly high accuracy.

A model for assessing the effectiveness of fault-detection processes was introduced based on reliability modeling techniques. The model is able to evaluate both fault-detection effectiveness and the number of faults in artifacts [8].

Studies carried out in industrial and academic settings do not appear to completely support the common belief that hierarchical structuring of UML models increases their understandability [81, 22].

An application of simplified Function Point Analysis to UML models of software has been proposed [106] via the establishment of a precise mapping between UML elements and the so-called Basic Functional Components, upon which Function Point Analysis measurement is based. It is possible to decrease the cost of modeling, and consequently the cost of measurement and estimation. The relatively low cost of the estimation models also allows developers to build different alternative models, to perform what-if analyses and choose the most economically sensible option.

While there is a vast literature about static, structural software measures, little is known about dynamic measures for a number of software quality attributes, i.e., measures that quantify software attributes such as size, complexity, and coupling based on actual executions of software code, instead of a static analysis. An initial study [108], based on data obtained on a few Open Source Software projects, shows that dynamic software measures may be as useful as or more useful than static, structural measures in building models for software fault-proneness.

References [8, 6, 14, 22, 45, 50, 53, 56, 60, 81, 106, 108]

4.1.3 Theoretical Validation of Software Measures

The research addressed the theoretical characterization of internal measures for a number of internal quality attributes (such as size, complexity, cohesion, and coupling) of software artifacts [6]. Each of these attributes was characterized via a set of axioms that a measure should satisfy to be deemed suitable for quantifying the attribute it purports to measure. This approach makes it easier to identify the similarities and differences among internal quality attributes, which all too often are classified as either contributing to size or to complexity. This characterization can be applied to several kinds of software artifacts, namely, specifications, designs, code, and test cases.

An extension of the approach was proposed to be used with interval, ordinal, and nominal measures, in addition to ratio measures for which it was originally proposed [55]. This characterization is one of the fundamental steps in the approach for the definition of software measures proposed in [19].

Further studies [85] have shown that the axiom sets that were originally proposed can be reduced and some of the initial axioms are implied by the reduced axiom sets. This has shown that there is a close relationship between the software size attribute and the concept of "measure" in Measure Theory in Mathematics. Furthermore, the reduced axiom sets have made it possible to highlight the relationships among internal software attributes.

The axiomatic approach has also been used and extended to provide a theoretical validation for the dynamic measures defined in [108].

A careful critical analysis addressed the way in which Measurement Theory has been used in Software Engineering and the way in which it should be used, according to some authors. This analysis showed that Measurement Theory has sometimes been used incorrectly and that Measurement Theory is a precious tool which, however, should be used in a nondogmatic way and always with common sense [7, 10, 11, 12].

An extension to Measurement Theory has been proposed in [65], to take into account all those cases in which a total ordering among the entities cannot be obtained. This has led to the definition of Weak Measurement Theory, where all basic concepts of Measurement Theory are extended, specifically those of nominal, ordinal, interval, and ratio scale. New measurement levels have been introduced in [72], where it was shown that, in some cases, it is meaningful to compute the average value of an ordinal measure or even of a measure defined on a set of entities that are not even totally ordered with respect to the attribute of interest.

The rigorous modeling of external attributes according to Measurement Theory has been proposed and it has been shown that external attributes should be modeled via a probabilistic approach that leads to estimation models [91]. The advantages of estimation models over weighted sums, and the intrinsic disadvantages of weighted sums have been discussed in [92].

The ISO9126 external software attributes have been investigated in [83]. The underlying idea was to check whether it is true that different external software attributes may be conflicting, i.e., that increasing one external quality may decrease some other external quality. The study involved several software practitioners and found that, with the exception of Portability, external software qualities are actually not believed to be conflicting.

A new measure definition process has been defined in [49, 19], based on previous applied experiences. This process is based on a set of explicit goals for which measurement activities are carried out and a deep knowledge of the software process under study. This will allow for the definition of measures that are significant and useful with respect to their application environment.

An analysis of fundamental issues and aspects in software measurement can be found in [37].

References [6, 7, 10, 11, 12, 19, 49, 55, 65, 72, 83, 85, 91, 92, 37, 108]

4.1.4 Web-related Measurement

An empirical study was carried out with students to evaluate the accuracy of the estimates of the effort needed to complete the design of a Web application [64].

The design phase of a Web application may require a significant part of the total development effort and can be subdivided in several sub-phases. The empirical study showed that there is a tendency to underestimate the effort needed for the single phases. However, effort estimates made by the subjects can be used as a part of an effort prediction model. In addition, the empirical study showed that there is a correlation between design qualities (such as size and complexity) and actual effort [68, 36]. The study was replicated with Computer Engineering and Communication Sciences students, to evaluate the impact of the students' backgrounds on the effort needed to complete the activities related to designing Web software [21]. This has also led to identifying some aspects that are common across the subjects' background.

An initial framework has been defined [114] for the measurement of a number of attributes of Web services along the lines theoretically defined in [91].

References [21, 36, 64, 68, 114]

4.1.5 Quantitative Evaluation of Free/Libre Open Source Software

Quantitative quality evaluation is a central aspect in the engineering and the widespread adoption of Free/Libre Open Source Software (FLOSS), like for any other product.

OpenBQR (Open Business Quality Rating) was initially proposed as a model for the evaluation of Open Source Software [80]. OpenBQR draws some elements from previously existing Open Source Software evaluation models, with the aim of providing a simpler and more complete evaluation method that can be practically used.

The evaluation of the trustworthiness of FLOSS products has been studied in the context of the QualiPSo project, in which Dr. Morasca was the leader of the activity devoted to the evaluation of FLOSS quality. The underlying idea is that FLOSS, like any other product, needs to be trustworthy to be adopted by end users and developers, who need to make informed decisions on the FLOSS products and components that best suit their needs. In the QualiPSo project, we used an organized, step-by-step approach to explore some of the most important issues related to FLOSS trustworthiness and provide viable solutions that can be used by FLOSS stakeholders–that is, end users and developers. Our approach is deeply rooted in knowledge that we elicited from many FLOSS stakeholders and through the analysis of several FLOSS projects.

These are the main results of this activity for end users and developers:

MOSST (Model of Open Source Software Trustworthiness): A customizable quality model for the trustworthiness of FLOSS products [95, 24, 104]

OP2A (Open Source Product Portal Assessment): A model for evaluating the quality of web portals that store FLOSS products [90, 103].

OSS-TMM (Open Source Software Testing Maturity Model): A maturity model specifically targeted to FLOSS [86, 26].

T-DOC (Test-DOCumentation): A framework that supports a team of FLOSS developers in creating test documentation that will enhance FLOSS trustworthiness [96].

Tools An integrated set of tools to collect data on FLOSS products and processes, and report results useful to their users.

Evaluations Additional evaluations of FLOSS products and of techniques and methodologies used during FLOSS development.

To this end, a number of tasks have been performed.

Identification of goals and trustworthiness factors. Different stakeholders may have different goals and factors in mind when choosing FLOSS products or components. We interviewed a number of diverse FLOSS stakeholders and we asked them to rank the factors that contribute to FLOSS trustworthiness. The result is an aggregate ranking of these factors that shows what kind of information FLOSS stakeholders would really like to have and the relative importance of the factors. This aggregate ranking was obtained via solid statistical techniques. We also profiled the stakeholders, so we can identify the needs of different kinds of stakeholders and find out if their needs actually differ [24, 84, 94].

Evaluation of the quality of portals and repositories. We investigated if portals and repositories hosting FLOSS products and components provide FLOSS stakeholders with enough information to make an informed decision on whether to choose some FLOSS products and components. We carefully examined the repositories and websites hosting a number of well-known FLOSS projects to find out what kind of information is actually available and match it against the ranking of factors that we found by interviewing FLOSS stakeholders. As a result, we can provide recommendations to FLOSS developers on what they need to add to FLOSS repositories and websites to 1) help FLOSS stakeholders make informed decisions, and 2) better promote their FLOSS product. We introduced OP2A (Open Source Product Portal Assessment) as a model for evaluating the quality of web portals that store FLOSS products, which provides guidelines for FLOSS developers to improve their own portals and repositories [90]. We also investigated the quality of the code in FLOSS repositories and the speed with which software problems are fixed by communities [99].

Identification of relevant measures. We used a goal-oriented method (the Goal/Question/Metric approach) to define candidate measures based on our problem, instead of trying to make sense out of a large number of measures, which may be costly to collect or even misleading if used.

So, there is a motivation behind each measure and the values collected with each measure can be interpreted in the context of our measurement goal, that is, FLOSS trustworthiness evaluation. This makes our approach even more "open," because everyone can see which measures have been used to quantify which factors and why [89, 95].

Definition of FLOSS verification and validation techniques. Software quality in general and trustworthiness in particular are greatly influenced by verification and validation. Techniques and methodologies are available for software products built in a "conventional" way, but it is unclear if they still apply to FLOSS production as well or they need to be modified. We thoroughly studied the various techniques available for "conventional" software to see how they can be adapted to FLOSS. We have found out that there is a lot of room for improvement and that FLOSS verification and validation are neither mature enough nor well-documented. So, we developed: 1) OSS-TMM (Open Source Software Testing Maturity Model) [86], a maturity model for FLOSS testing, to show the progression level of a FLOSS testing process; 2) T-DOC (Test-DOCumentation) [96], a Javadoc-like notation that helps testers record information about testing.

Automation of the use of our approach. We developed new tools and integrated and extended existing data collection, measurement, and reporting tools to capture and report the information that would be useful according to the FLOSS stakeholders we interviewed. The user simply needs to specify the necessary information (name, version, ...) about the project that needs to be analyzed and receives the analysis results via a unified interface, provided by Spago4Q.

Estimation of FLOSS trustworthiness. Before QualiPSo, there were no trustworthiness models available for FLOSS, but only quality models typically based on weighted sums of measures. However, FLOSS users may be left on their own when establishing sensible weights that best fit their own needs, because real evidence is hardly ever available about the actual usefulness and impact of the single measures on the overall quality. In addition, quality models may include a large number of measures. One may need to measure all of them, even though their contribution to the assessment may be quite low, while the cost of measuring them may be quite high.

We have defined MOSST (Model of Open Source Software Trustworthiness), a customizable estimation model for the trustworthiness of FLOSS products [95]. Based on the values of measures that can be collected automatically, MOSST provides an estimation of the probability that a FLOSS product is rated as trustworthy, that is, the percentage of stakeholders that would rate the product as trustworthy. So, for developers of a FLOSS product, MOSST provide an assessment of how well the product will be received; for users, MOSST will estimate how other people may estimate the product's trustworthiness.

MOSST has been built by applying solid statistical techniques to data coming from the field. We collected trustworthiness data by surveying a diverse set of FLOSS stakeholders by means of a questionnaire on how they would evaluate several Java- and C/C++-based FLOSS projects. We surveyed hundreds of FLOSS stakeholders and have so far collected thousands of evaluations of FLOSS products. We have automatically collected data about those projects and we have found that correlations exist between these data and the trustworthiness as reported by the surveyed stakeholders. The correlation analysis provides FLOSS stakeholders with an idea of the importance of each factor, of the weight that should be given to each factor, and of the measures that contribute only little to the estimation. These measures may be dropped, thus reducing the cost of data collection and making the interpretation and use of the models simpler.

MOSST can be used by stakeholders for other qualities too, like, for instance, perceived Reliability, Usability, Portability, Interoperability, Security, Documentation Quality, ..., because, when we surveyed FLOSS stakeholders, we also asked them about all of these qualities for the projects we chose. So, even for those qualities, MOSST can provide an estimation of the probability that FLOSS stakeholders positively or negatively evaluate a project.

Thanks to the profiling of the respondents in our surveys, specific models can be customized for specific classes of stakeholders. Also, the models allow FLOSS stakeholders to identify ranges ("good," "fair," and "poor") for the various qualities. FLOSS stakeholders can check how well a specific product fares in each of those qualities and make a decision that is based on evidence with the support of our tools.

In some cases, even the application of simple coding rules can be used as a predictor for the trustworthiness and quality of FLOSS products.

During the development of one of the tools, we also introduced SCRUM in a modified way and we assessed its advantages and disadvantages when compared to the development approach we had previously used [97].

As a result of the QualiPSo project, a network of Competence Centers will be created. We discussed the issues and advantages of the Italian Competence Center in [98].

A preliminary investigation has led to insights into the strategies for FLOSS marketing and communication, by interviewing FLOSS professionals [107].

References [24, 80, 84, 86, 89, 90, 95, 94, 95, 96, 97, 98, 99, 101, 103, 24, 104, 107]

4.1.6 Innovative Data Analysis Techniques

In [18, 54], two data analysis techniques—Rough Sets and Logistic Regression—are compared, from both the theoretical and the experimental points of view. A hybrid approach is introduced that combines the strengths of the two data analysis techniques and reduces Type I and Type II errors. The approach was validated on data from the DATATRIEVE project at Digital Engineering Italia.

A new approach to deal with continuous attributes in decision trees is proposed in [59, 63], to extend current decision trees, which may be used with essentially discrete attributes, so they require that continuous data be discretized first, which implies loss of information and prediction power of the measures. In addition, the discretization process is largely arbitrary. The proposed method can solve these problems in a way that is perfectly consistent with discrete decision trees. An application showed that the method may be more accurate that the discrete decision trees in predicting software module fault-proneness.

An axiomatic approach was introduced for the definition of concentration indicators for nominal data distributions. This approach allows the definition of concentration indicators that can be selected based on specific application needs. Quite surprisingly, these concentration indicators can be built by using Riemann's Z-transforms. It was also shown that it is not true that is is not meaningful to take the average of a set of ordinal data. A theorem shows the necessary and sufficient condition needed to make the averaging operation meaningful [72].

Least Median of Squares Linear Regression (LMS), a specific kind of Robust Regression, was used [21] on data sets collected in empirical studies. Using LMS allows researchers to avoid assumptions on the data at hand that may not hold, like the assumptions that are used in Ordinary Least Squares Linear Regression (OLS). Later, a novel statistical significance test was introduced for univariate models built with LMS [87]. This test has been used [100, 25] to check the existence of a statistically significant relationship between Function Points and COSMIC Full Function Points on a real-life data. Least Quantile of Squares Linear Regression (LQS), an extension of LMS, was investigated, used for effort estimation, and compared to LMS and OLS [105]. It seems that LQS may be a valid alternative to LMS or OLS when LMS and OLS cannot be used.

Paper [110] introduces the concept of data analysis anti-patterns, i.e., data analysis procedures that may lead to invalid results that may mislead decision makers. Two examples of anti-patterns are presented and discussed.

The classification of modules as faulty or non-faulty usually involves setting a fault-proneness threshold: software modules whose estimated fault-proneness is above that threshold are classified as estimated faulty and the others as estimated non-faulty. The selection of the threshold value is to some extent subjective and arbitrary, and different threshold values may lead to very different results in terms of classification accuracy. We proposed a technique, based on a property of Binary Logistic Regression fault-proneness estimation models, that does not require that a threshold be fixed [115].

Thus, we first demonstrate a property according to which the number of actually faulty software modules in the training set used to build a model is equal to the number of modules estimated faulty in that set, i.e., estimation of the number of faulty modules is perfect on the training set. Then, we use the model on a test set, and estimate the number of faulty modules. We also estimate the number of faulty modules in the test set by using a more conventional approach with five different fault-proneness thresholds, and we finally compare the estimates with the estimates obtained via our approach. We carried out the empirical validation on a data set from NASA hosted on the PROMISE repository, by using a technique similar to the one used in K-fold cross validation. In the empirical validation we carried out, the approach we propose is able to estimate the number of faulty modules in the test sets better than the threshold-based ones, in a statistically significant way.

References [18, 54, 59, 63, 72, 87, 100, 25, 105, 110, 115]

4.1.7 Software Effort Estimation

We carried out an empirical study on the estimation of software development effort broken down by phase, so that it can be used along the software development lifecycle [116]. The research has two goals. At any given point in the software development lifecycle, we estimate the effort needed for the next phase. Also, we estimate the effort for the remaining part of the software development process. Our empirical study is based on historical data from the ISBSG database. The results show a set of statistically significant correlations between: (1) the effort spent in one phase and the effort spent in the following one; (2) the cumulative effort spent up to a phase and the effort spent in the next phase; (3) the effort spent in a phase and the remaining effort; (4) the cumulative effort up to the current phase and the remaining effort. However, the results also show that these estimation models come with different degrees of goodness-of-fit. Finally, including further information, such as the functional size, does not significantly improve estimation quality.

References [116]

4.1.8 Software Engineering Education and Empirical Methods

A number of experiences on using students in software courses as subjects in empirical studies have been summarized in [35, 69]. The scientific literature usually only addresses research-related issues, such as the external validity of the results obtained via experiments that use students as subjects. It is therefore necessary that, in addition to the researcher's viewpoint, a number of other viewpoints be taken into account, namely: the student's, the instructor's, and the software industry's. Each of these viewpoints should obtain benefits when empirical students are carried out during software classes. A few aspects of the relationships between industry and teaching are further investigated in [75, 76, 77]. Teaching students specific empirical software engineering topics has been investigated in [74, 78]. The sometimes conflicting goals and benefits of the different viewpoints and the stakeholders' various relationships make it necessary to carefully plan the execution of empirical studies with students during software engineering courses. To this end, a checklist for integrating student empirical studies with research and teaching goals has been defined in [23].

Empirical methods were also used in [112] to investigate how participants experience game prototyping activities and their use in the context of a software course. Statistical analysis indicates that participants' satisfaction and activity's usefulness are the most influential factors for participants' intention to attend similar activities in the future. Qualitative analysis suggests improvements on how to prepare the participants, introduce the software used in the courses, and enrich the variety of the materials used.

References [23, 35, 69, 74, 75, 76, 77, 78, 112]

4.2 Specification of Concurrent and Real-time Systems

This area was the major theme of the research activity carried out during the doctorate [119]. An extended model of Petri nets, Environment/Relationship nets (ER nets) was formally defined. This high-level Petri net model unifies high-level Petri nets (where only functional features are considered

and time is not modeled) and timed Petri nets (where the functional behavior is not taken into account, and only time is considered). Functional and timing aspects are modeled in a homogeneous way, so ER nets allow modelers to represent the interaction of functional and timing aspects of concurrent and real-time systems in a natural way [2, 39, 40], unlike previous Petri nets-based formalisms. The way time has been introduced in ER nets can also be used to introduce timing aspects in all other high-level Petri nets, as shown in [3]. MTER nets, an extension of ER nets, provide a way to model systems with several time references [57].

ER nets have been used as the kernel formalism for an environment for specification and verification of concurrent time dependent systems [39, 41]. In addition, ER nets have been used in the modeling of hardware systems [44, 46], and a stochastic extension of ER nets has been used to model software processes [48].

A timed extension of UML has been defined to adequately capture timing aspects in UML [66, 70]. Time-related features are introduced in a rigorous way in UML's Statecharts and OCL.

References [2, 3, 39, 40, 41, 44, 46, 48, 57, 66, 70, 119]

4.3 Software Verification

4.3.1 Operational Models

ER Nets were taken as the reference operational model for carrying out symbolic execution of specifications and implementations of concurrent systems. An algorithm was first proposed for a class of ER Nets that can model concurrent and discrete-time systems. Then, a more specific algorithm was proposed for a subclass of ER Nets, which can be used to model most systems of interest. This algorithm uses a significantly lower amount of information than the first one [1, 42]. The research has also focused on the definition of a set of test case selection criteria, based on the topological coverage of the underlying net [43]. In addition, the timing properties of TB Nets were studied. TB Nets are a subclass of ER Nets in which only temporal information is modeled and used. This line of research led to the definition of methods for evaluating the reachability of specific states within a specified time interval [4].

References [1, 4, 42, 43]

4.3.2 Logic-based Models

Sandro Morasca has carried out research on the generation of functional test cases for real-time systems specified via logic languages. Based on the temporal logic language TRIO, an algorithm for the (semi)automatic derivation of test cases was devised [47]. To curb the intrinsic complexity of the problem, a number of coverage criteria were defined, as is usually done in structural software testing [5]. This also led to the design and prototypal implementation of a test case generation tool.

Paper [51] shows the proposal of several criteria and techniques for the generation of test cases "in the large" for real-time systems specified with TRIO+, which is an object-oriented extension of TRIO. In addition, a technique was proposed [16] for generating functional test cases for modular, hierarchical, real-time software systems. This technique can be applied to a large class of specification languages. The use of this technique was also practically supported with the implementation of an automated tool [17, 58].

References [5, 16, 17, 47, 51, 58]

4.3.3 Mutation Analysis

Mutation analysis, which is usually applied to sequential programs, was extended to concurrent programming. Ada was taken as the reference language and a number of specific mutant operators were defined for the language's concurrent constructs. The application of the technique defined in [52] allows the evaluation of how thoroughly a test set exercise the concurrent features of an Ada program.

References [52]

4.3.4 Stopping Rules for Testing

One of the most important problems in software verification lies in finding stopping rules for testing. The literature contains several proposals that are usually based on criteria that look objective. However, these proposals may not be adequate, especially for systems with high reliability requirements. An approach based on Bayesian statistics was introduced, based on the idea that testing actually stops when testers have reached a sufficient degree of confidence in the correctness of a program. The use of these techniques in this context was first subject to a systematic theoretical analysis. Then, a model was proposed to describe how this degree of confidence varies during the test process. An experiment provided evidence that supported the initial hypotheses [67].

An approach was proposed for assessing the mean failure frequency of a program, based on the statistical test of hypotheses [79]. The approach can be used to establish stopping rules and evaluate the quality of a program based on its mean failure frequency during the late testing phases. The proposal shows how to set and satisfy conservative bounds for the minimum number of test executions that are needed to achieve a target mean failure frequency with a specified level of statistical significance, based on the quality goal of testing and the specific test execution profile chosen. The approach also relaxes a few assumptions of the literature, so it can be used in a larger set of real-life cases.

References [67, 79]

4.3.5 Analytical Studies

The assessment of different testing can be carried out by evaluating their effectiveness in causing 1) a high number of software failures, or 2) at least one failure. The theoretical study of [71] first identifies the least restrictive constraints for rationally allocating test data across the input subdomains of a software program. Specifically, a theorem (a necessary and sufficient condition) was first proven that allows the comparison of the effectiveness of different testing techniques based on the expected number of failures they cause. This theorem is based on the only knowledge of the ordering of the failure rates of the subdomains. So, the knowledge of the exact values of the failure rates is not required. The theorem was also extended to the case in which a partial ordering of the subdomain failure rates is known. Also, along the same line of reasoning, a similar theorem was proven to compare different testing techniques in terms of the probabilities of causing at least one failure.

References [71]

4.3.6 Model-driven Verification of Software for Web Systems

The generation of test data for a Web application generator has been investigated. Specifically, the test data are defined as Web application models coded in WebML. Different application models exercise different productions of the WebML grammar, so coverage measures have been defined for these productions. A preliminary empirical analysis carried out in an industrial environment has allowed the study of how well the test data exercise the entire grammar, so as to devise possible improvements in the quality of the test set by adding more test data [73].

References [73]

4.4 Analysis of On-line Social Networks

The unauthorized propagation of information is an important problem in the Internet, especially because of the increasing popularity of On-line Social Networks. To address this issue, many access control mechanisms have been proposed so far, but there is still a lack of techniques to evaluate the risk of unauthorized flow of information within social networks. We have introduced a probability-based approach to modeling the likelihood that information propagates from one social network user to users who are not authorized to access it. The approach is demonstrated via an example, to show how it can be applied in practical cases [102].

References [102]

4.5 Semantic Web Services

Semantic web services are gaining more attention as an important element of the emerging semantic web. Therefore, testing semantic web services is becoming a key concern as an essential quality assurance measure. The objective of the systematic literature review carried out [28] is to summarize the current state of the art of functional testing of semantic web services by providing answers to a set of research questions. The review follows a predefined procedure that involves automatically searching 5 well-known digital libraries. After applying the selection criteria to the results, a total of 34 studies were identified as relevant. Required information was extracted from the studies and summarized. Our systematic literature review identified some approaches available for deriving test cases from the specifications of semantic web services. However, many of the approaches are either not validated or the validation done lacks credibility. We believe that a substantial amount of work remains to be done to improve the current state of research in the area of testing semantic web services.

References [28]

5 Fellowships

Sandro Morasca earned the following fellowships.

- One fellowship on "Testing funzionale di sistemi software" from Selenia S. p. A., in 1987, 1988 e 1989.
- One eighteen month fellowship of the European Consortium ERCIM for research to be carried out at European universities and scientific institutions.
- Two twelve month fellowships of CNR (the Italian National Research Council) for research to be carried out at Italian universities and scientific institutions.
- Two twelve month fellowships of CNR (the Italian National Research Council) for research to be carried out at foreign universities and scientific institutions.

6 Prizes

Sandro Morasca has received the following prizes:

- Prize "Francesco Somaini" (1979) as best student in his high school to enroll in a scientific university
- "Electrical Engineering Student Award" (1988) from the IEEE North Italy section.
- "Seymour Cray" 1991.
- Final prize for a fellowship from CNR (the Italian National Research Council) for research to be carried out at foreign universities and scientific institutions.

7 Projects

Sandro Morasca's research activities have been partially carried out in the following projects.

National projects List:

- Progetto MPI 40% "Reti di Petri: modelli, applicazioni, strumenti"
- Progetto MPI 60% "Strumenti per la specifica e la verifica di sistemi software in tempo reale"

- Progetto CINI "PROTAGORA"
- Progetto cofinanziato MURST "MOSAICO"
- Progetto cofinanziato MIUR "QUACK"

International projects List:

- EU-ESPRIT project "IPTES"
- EU-ESSI project "CEMP"
- EU-ESSI project "PROMOTE"
- EU-IST Project "ESERNET" (Thematic Network)
- EU-IST Project "QualiPSo" (Integrated Project)

8 Talks

Tutorials Sandro Morasca's activity includes the following tutorials:

- "Formal Methods in Software Measurement and Software Measurement in Formal Methods," presented at the international conferences METRICS 2003 and Formal Methods Europe 2003
- "An Introduction to Web Quality," presented (with Luciano Baresi) at the international conference ICWE (International Conference on Web Engineering) 2004
- "Fundamental Aspects of Empirical Software Engineering," presented at ESELAW 2007
- "Empirical Software Engineering," presented at the 6th International Summer School on Software Engineering, Salerno, Italy, September 16 - 18, 2009
- "QualiPSo Overview," presented (with Etiel Petrinja) at OSS2010, South Bend, IN, USA, June 2, 2010

Keynotes Sandro Morasca has been invited to deliver a keynote speech at the following international conferences

- OSS 2007
- ESELAW 2007
- SAST 2008
- MetriSec 2009
- OSS 2013

Invited Talks He has been invited to deliver talks at several institutions, including

- University of Stuttgart, Germany
- Universidad de Castilla-La Mancha at Ciudad Real, Spain
- Norwegian University of Science and Technology in Trondheim, Norway
- Free University of Bolzano-Bozen, Italy
- Universidad Politécnica de Valencia, Spain
- Fraunhofer Center Maryland at College Park, USA
- North Carolina State University at Raleigh, USA

9 Reviewing Activities

9.1 International Journals

Editorial Board Member of the Editorial Board of "Empirical Software Engineering: An International Journal," published by Springer-Verlag (5-year impact factor for 2011: 2.039)

Reviewer Reviewer of articles for international journals including: IEEE Transactions on Software Engineering, ACM Transactions on Software Engineering Methodology, ACM Computing Surveys, IEEE Transactions on Reliability, IEEE Transactions on Services Computing, Information and Software Technology, ACM Computing Surveys, Advances in Engineering Software, Advances in Software Engineering, Fundamenta Informaticae, IEE Proceedings, International Journal of Open Source Software and Processes, International Journal of Software Engineering and Knowledge Engineering, Information Sciences, Journal of Systems Architecture, Journal of Software Maintenance and Evolution: Research and Practice, Journal of Systems and Software, Software and System Modeling.

Guest Editor He was guest editor of a Special Issue on Knowledge Discovery from Software Engineering Data of the International Journal of Software Engineering and Knowledge Engineering [15].

9.2 International Conferences

9.2.1 Chair and Co-chair

Sandro Morasca has served as Chair or Co-chair for the following conferences.

- Program Chair of a workshop on measuring Object-Oriented Software at ECOOP'98.
- Program co-Chair of the "First International Workshop on Web Quality" held jointly with ICWE (International Conference on Web Engineering) 2004.
- Program co-Chair of a workshop on "Measurement and Metrics" held jointly with WWW 2005.
- Program co-Chair of the "1st International Workshop on Trust in Open Source Software" (TOSS) held jointly with OSS (Open Source Software) 2007 [82].
- Program co-Chair of ICSEA 2007.
- Chair of the "Second International Doctoral Symposium on Empirical Software Engineering" held as a part of the International Empirical Software Engineering Week 2007.
- Program co-Chair of the "3rd International Workshop on Designing Empirical Studies: Assessing the Effectiveness of Agile Methods" (IWDES 2009) held jointly with XP (Extreme Programming) 2009 [88]
- Program co-Chair of the Short papers & posters track of PROFES 2012
- Program co-Chair of IWSM-MENSURA 2012

9.2.2 Steering Committees

He has served in the Steering Committees of the following international conferences:

- METRICS
- ESEM

9.2.3 Program Committees

Sandro Morasca has served on the PC of a number of international software engineering conferences, including

- EUROMICRO
- Workshop on Industrial-strength Formal Methods (WIFT)
- "International Conference on Software Engineering and Software Engineering" (SEKE)
- "International Conference on Software Maintenance" (ICSM)
- "International Symposium on Software Metrics" (METRICS)
- "European Conference on Software Quality" (ECSQ)
- "International Conference on Web Engineering" (ICWE)
- "International Symposium on Software Reliability Engineering" (ISSRE)
- "International Symposium on Empirical Software Engineering" (ISESE)
- "Fundamental Approaches in Software Engineering" (FASE)
- "International Conference on Software Engineering Advances" (ICSEA)
- "Automated Software Engineering" (ASE)
- "Brazilian Symposium on Software Engineering" (CBSOFT-SBES)
- "International Conference on Software and Data Technologies" (ICSOFT)
- "International Symposium on Empirical Software Engineering and Measurement" (ESEM)
- "International Conference on Resource Intensive Applications and Services" (INTENSIVE)
- "Italian Workshop on Eclipse Technologies" (Eclipse-IT)
- "Web Quality and Testing Workshop" at ICWE
- "International Workshop on Security Measurements and Metrics" (MetriSec)
- "Conference on Open Source Software" (OSS)
- "Predictive Models in Software Engineering" (PROMISE)
- "Quantitative Approaches on Object-Oriented Software Engineering and Related Paradigms" (QAOOSE)
- "International Conference on the Quality of Information and Communications Technology" (QUATIC), track on Quality in Web Engineering
- "International Working Conference on Source Code Analysis and Manipulation" (SCAM)
- "Software Measurement European Forum" (SMEF)
- "Workshop on Emerging Trends in Software Engineering" (WETSoM)
- "Workshop on Public Data about Software Development" (WoPDaSD)
- "International Conference On Signal-Image Technology and InternetBased Systems" (SITIS), track on Open Source Software Development and Solutions (OSSDS)
- "Latin American Web Congress" (LA-WEB)

- "Estimation and Prediction in Software & Systems Engineering" (EsPreSSE)
- "International Conference on Quality Software" (QSIC)
- "Value Creation Process in Agile Projects" (VALOIR)
- "New Ideas and Emergent Results" (NIER)
- "PROFES 2011- Doctoral Symposium"(PROFES)
- "Evaluation & Assessment in Software Engineering"(EASE)

In addition, he has been a reviewer of articles for other international conferences, including ICSE and ESEC.

9.2.4 Conference Organization

Sandro Morasca has participated in the organization of the following conferences as

- Finance Chair of the IEEE and ACM "Sixth International Workshop on Software Specification and Design"
- Organizing Committee Member of Congresso AICA 2001
- General Chair of METRICS 2005.

10 Public Administration Committees

Ministry Level Sandro Morasca was a member of the Committee on Open Source Software of the Italian "Ministero per le Riforme e le Innovazioni nella Pubblica Amministrazione."

Local Level Sandro Morasca served on a Committee on software strategies of the Provincia di Como.

11 Publications

11.1 International Journals

[1] Carlo Ghezzi, Dino Mandrioli, Sandro Morasca, and Mauro Pezzè. Symbolic execution of concurrent systems using Petri nets. *Computer Languages*, 14(4):263–281, December 1989.

[2] Carlo Ghezzi, Dino Mandrioli, Sandro Morasca, and Mauro Pezzè. A unified high-level Petri net formalism for time-critical systems. *IEEE Transactions on Software Engineering*, 17:160–172, February 1991.

[3] Sandro Morasca, Mauro Pezzè, and Marco Trubian. Timed high-level nets. *Real-Time Systems*, 3:165–189, July 1991.

[4] Carlo Ghezzi, Sandro Morasca, and Mauro Pezzè. Validating timing requirements for time basic net specifications. *Journal of Systems and Software*, 27:97–117, November 1994.

[5] Dino Mandrioli, Sandro Morasca, and Angelo Morzenti. Generating test cases for real-time systems from logic specifications. *ACM Transactions on Computer Systems*, 13:365–398, November 1995.

[6] Lionel C. Briand, Sandro Morasca, and Victor R. Basili. Property-based software engineering measurement. *IEEE Transactions on Software Engineering*, 22:68–86, January 1996.

[7] Lionel Briand, Khaled El Emam, and Sandro Morasca. On the application of measurement theory in software engineering. *Empirical Software Engineering*, 1:61–88, 1996.

[8] Sandro Morasca. Assessment of fault detection processes: An approach based on reliability techniques. *IEEE Transactions on Reliability*, 45(4):632–637, December 1996.

[9] Sandro Morasca. Applying QIP/GQM in a maintenance project. *Empirical Software Engineering*, 2:163–166, February 1997.

[10] Sandro Morasca, Lionel C. Briand, Victor R. Basili, Elaine J. Weyuker, and Marvin V. Zelkowitz. Comments on "towards a framework for software measurement validation" (correspondence paper). *IEEE Transactions on Software Engineering*, 23:187–188, March 1997.

[11] Lionel C. Briand, Sandro Morasca, and Victor R. Basili. Response to: Comments on "Property-based software engineering measurement: Refining the additivity properties" (correspondence paper). *IEEE Transactions on Software Engineering*, 23:196–197, March 1997.

[12] Lionel C. Briand, Khaled El Emam, and Sandro Morasca. Reply to "Comments to the paper: Briand, El Emam, Morasca: On the Application of measurement theory in software engineering" (correspondence paper). *Empirical Software Engineering*, 2:317–322, March 1997.

[13] Alfonso Fuggetta, Luigi Lavazza, Sandro Morasca, Stefano Cinti, Giandomenico Oldano, and Elena Orazi. Applying GQM in an industrial software factory. *ACM Transactions on Software Engineering and Methodology*, 7:411–448, October 1998.

[14] Lionel C. Briand, Sandro Morasca, and Victor R. Basili. Defining and validating measures for object-based high-level design. *IEEE Transactions on Software Engineering*, 25:722–743, September 1999.

[15] Sandro Morasca and Günther Ruhe. Guest editors' introduction of the special issue on knowledge discovery from software engineering data. *International Journal of Software Engineering and Knowledge Engineering*, 9(5):496–498, October 1999.

[16] Pierluigi San Pietro, Angelo Morzenti, and Sandro Morasca. Generation of execution sequences for modular time critical systems. *IEEE Transactions on Software Engineering*, 26:128–149, February 2000.

[17] Sandro Morasca, Angelo Morzenti, and Pierluigi San Pietro. A case study on applying a tool for automated system analysis based on modular specifications written in TRIO. *Automated Software Engineering*, 7:125–155, May 2000.

[18] Sandro Morasca and Günther Ruhe. A hybrid approach to analyze empirical software engineering data and its application to predict module fault-proneness in maintenance. *Journal of Systems and Software*, 53:225–237, September 2000.

[19] Lionel C. Briand, Sandro Morasca, and Victor R. Basili. An operational process for goal-driven definition of measures. *IEEE Transactions on Software Engineering*, 28:1106–1125, December 2002.

[20] Giovanni Denaro, Mauro Pezzè, and Sandro Morasca. Towards industrially relevant fault-proneness models. *International Journal of Software Engineering and Knowledge Engineering*, 13(4):395–417, August 2003.

[21] Luciano Baresi and Sandro Morasca. Three empirical studies on estimating the design effort of Web applications. *ACM Transactions on Software Engineering and Methodology*, 16, September 2007.

[22] José A. Cruz-Lemus, Marcela Genero, M. Esperanza Manso, Sandro Morasca, and Mario Piattini. Assessing the understandability of UML statechart diagrams with composite states–A family of empirical studies. *Empirical Software Engineering*, 14:685–719, December 2009.

[23] Jeffrey C. Carver, Letizia Jaccheri, Sandro Morasca, and Forrest Shull. A checklist for integrating student empirical studies with research and teaching goals. *Empirical Software Engineering*, 15:35–59, February 2010.

[24] Vieri del Bianco, Luigi Lavazza, Sandro Morasca, and Davide Taibi. A survey on open source software trustworthiness. *IEEE Software*, 28(5):67–75, 2011.

[25] Luigi Lavazza and Sandro Morasca. Convertibility of function points into cosmic function points: a study using piecewise linear regression. *Information & Software Technology*, 53(8):874–884, 2011.

[26] Sandro Morasca, Davide Taibi, and Davide Tosi. OSS-TMM: Guidelines for improving the testing process of open source software. *International Journal of Open Source Software and Processes*, 3(2):1–22, 2011.

[27] Luigi Lavazza, Sandro Morasca, and Gabriela Robiolo. Towards a simplified definition of function points. *Information & Software Technology*, 55(10):1796–1809, 2013.

[28] Abbas Tahir, Davide Tosi, and Sandro Morasca. A systematic review on the functional testing of semantic web services. *Journal of Systems and Software*, 86:2877–2889, November 2013.

[29] Jehad Al Dallal and Sandro Morasca. Predicting object-oriented class reuse-proneness using internal quality attributes. *Empirical Software Engineering*, 19(4):775–821, 2014.

[30] Vieri Del Bianco, Luigi Lavazza, Geng Liu, Sandro Morasca, and Abedallah Zaid Abualkishik. Model-based early and rapid estimation of cosmic functional size an experimental evaluation. *Accepted for publication in Information & Software Technology.*

[31] Luigi Lavazza, Sandro Morasca, and Davide Tosi. Using function point analysis and cosmic for measuring the functional size of real-time and embedded software: a comparison. *International Journal On Advances in Software*, 7(1&2), 2014. URL=http://www.iariajournals.org/software/.

11.2 Books

[32] Carlo Ghezzi, Alfonso Fuggetta, Sandro Morasca, Angelo Morzenti, and Mauro Pezzè. *Ingegneria del software - Progettazione, sviluppo e verifica (in Italian).* Mondadori Informatica, 1991.

[33] Augusto Celentano, Pierluigi Della Vigna, Piero Fraternali, and Sandro Morasca. *Fondamenti di Informatica - Dal problema al programma (in Italian).* ETAS Libri, 1999.

11.3 Refereed Book Chapters

[34] Sandro Morasca. Software measurement. In *Handbook of Software Engineering and Knowledge Engineering - Volume 1: Fundamentals (S.K. Chang ed.)*, pages 239–276. World Scientific, 2001.

[35] Jeffrey Carver, Maria Letizia Jaccheri, Sandro Morasca, and Forrest Shull. Using empirical studies during software courses. In *Empirical Methods and Studies in Software Engineering, Experiences from ESERNET*, volume 2765 of *Lecture Notes in Computer Science*, pages 81–103, Berlin, Heidelberg, 2003. Springer-Verlag.

[36] Luciano Baresi, Sebastiano Colazzo, Luca Mainetti, and Sandro Morasca. W2000: A modeling notation for complex Web applications. In *Web Engineering (Emilia Mendes and Nile Mosley eds.)*, pages 335–364, Berlin, Heidelberg, 2006. Springer-Verlag.

[37] Sandro Morasca. Fundamental aspects of software measurement. In *International Summer School on Software Engineering–Tutorial Lectures*, volume 7171 of *Lecture Notes in Computer Science*, pages 1–45, Berlin, Heidelberg, 2013. Springer-Verlag.

11.4 Conference Proceedings

[38] Roberto Lecciso, Stefano Mainetti, and Sandro Morasca. Software metrics: a critical evaluation and an application to Pascal. *Microprocessing and Microprogramming. Proceedings of EUROMICRO '86, Venice, Italy, September 16-18*, 18(1-5):605–616, 1986.

[39] Sandro Morasca and Mauro Pezzè. Reti di Petri estese e sviluppo di sistemi in tempo reale (in italian). In *Atti del Congresso Annuale AICA, Cagliari, Italy, September 28 - 30, 1988*, pages 267–281, 1988.

[40] Carlo Ghezzi, Dino Mandrioli, Sandro Morasca, and Mauro Pezzè. A general way to put time in Petri nets. In *IWSSD '89: Proceedings of the 5th International Workshop on Software Specification and Design, Pittsburgh, PA, USA, May 19-20, 1989*, pages 60–67, New York, NY, USA, 1989. ACM.

[41] Sandro Morasca and Mauro Pezzè. The rationale of an environment for real-time software. In *EUROMICRO Workshop on Real Time, Como, Italy, June 14 - 16, 1989*, pages 37–42. IEEE CS Press, 1989.

[42] Sandro Morasca and Mauro Pezzè. Validation of concurrent Ada programs using symbolic execution. In *Proceedings of the 2nd European Software Engineering Conference*, ESEC'89, Coventry, UK, September 11-15, 1989. Lecture Notes in Computer Science, volume 387 (Carlo Ghezzi and John McDermid eds.), pages 469–485, Berlin, Heidelberg, 1989. Springer-Verlag.

[43] Sandro Morasca and Mauro Pezzè. Using high-level Petri nets for testing concurrent and real-time systems. In *Workshop on Real-time Systems - Theory and Applications*, York, UK, September 28-29, 1989, pages 119–131. North-Holland, 1989.

[44] Giacomo Buonanno, Sandro Morasca, Mauro Pezzè, Kim Portman, and Donatella Sciuto. Using high-level Petri nets for timing specification of hardware. In *Proceedings of the ACM International Workshop on Timing Issues in the Specification and Synthesis of Digital Systems*, TAU '90, Vancouver, BC, Canada, August 15 - 17, 1990, 1990.

[45] Flavio De Paoli and Sandro Morasca. Extending software complexity metrics to concurrent programs. In *Proceedings of the 14th Annual International Computer Software and Applications Conference*, COMPSAC '90, Chicago, IL, USA, October 29 - November 2, 1990, pages 414–419, Washington, DC, USA, 1990. IEEE Computer Society.

[46] Giacomo Buonanno, Sandro Morasca, Mauro Pezzè, Kim Portman, and Donatella Sciuto. A new timed Petri net model for hardware representation. In *Proceedings of the Computer Hardware Description Languages and their Application*, CHDL '91, Marseille, France, April 22-24, 1991. Dependable Computing and Fault-Tolerant Systems, pages 261–280. North-Holland, 1991.

[47] Dino Mandrioli, Sandro Morasca, and Angelo Morzenti. Functional test case generation for real-time systems. In *3rd IFIP Working conference on Dependable Computing for Critical Applications*, DCCA '92, Palermo, Italy, September 14-16, 1992. Dependable Computing and Fault-Tolerant Systems, pages 13–26, Berlin, Heidelberg, 1992. Springer-Verlag.

[48] Gianluigi Caldiera and Sandro Morasca. Performance evaluation of software process evolution. In *International Workshop on Evolution of Software Processes*, McGill University, Montreal, QC, Canada, January 26-28, 1993, 1993.

[49] Lionel C. Briand and Sandro Morasca. Useful metrics must depend on a process and a goal. In *Proceedings of the International Workshop on Metrics in Software Maintenance*, IWMSM '93, Montreal, QC, Canada, September 26, 1993, 1993.

[50] Lionel C. Briand, Sandro Morasca, and Victor R. Basili. Measuring and assessing maintainability at the end of high level design. In *Proceedings of the Conference on Software Maintenance*, ICSM '93, Montreal, QC, Canada, September 27-30, 1993, pages 88–97, Washington, DC, USA, 1993. IEEE Computer Society.

[51] Sandro Morasca, Angelo Morzenti, and Pierluigi San Pietro. Generating functional test cases in-the-large for time-critical systems from logic-based specifications. *SIGSOFT Software Engineering Notes*, 21 (3):39–52, Proceedings of the ACM/SIGSOFT International Symposium on Software Testing and Analysis, ISSTA 1996, San Diego, CA, USA, January 8–10, 1996, May 1996.

[52] Sandro Morasca, Mauro Pezzè, and Sergio Silva-Barradas. Mutation analysis for concurrent ada programs. In *Quality Week*, QW '96, San Francisco, CA, USA, May 21-24, 1996, 1996.

[53] Sandro Morasca. Defining measures for Petri net-based specifications of concurrent software. In *Annual Oregon Workshop on Software Metrics*, AOWSM '97, Coeur d'Alene, OR, USA, May 12-13, 1997, 1997.

[54] Sandro Morasca and Günther Ruhe. Knowledge discovery from software engineering measurement data: A comparative study of two analysis techniques. In *Proceedings of the 9th international conference on Software engineering and knowledge engineering*, SEKE '97, Madrid, Spain, June 18-20, 1997, pages 450 – 458, Skokie, IL, USA, 1997. Knowledge Systems Institute.

[55] Sandro Morasca and Lionel C. Briand. Towards a theoretical framework for measuring software attributes. In *Proceedings of the 4th International Symposium on Software Metrics*, IEEE METRICS '97, Albuquerque, NM, USA, November 5-7, 1997, pages 119–126, Washington, DC, USA, 1997. IEEE Computer Society.

[56] Lionel C. Briand and Sandro Morasca. Software measurement and formal methods: A case study centered on TRIO+ specifications. In *Proceedings of the 1st International Conference on Formal Engineering Methods*, ICFEM '97, Hiroshima, Japan, November 12-14, 1997, pages 315–325, Washington, DC, USA, 1997. IEEE Computer Society.

[57] Sandro Morasca. Using MTER nets for modeling multiple time references in distributed systems. In *Proceedings of Distributed computer control systems 1998*, DCCS 98, Como, Italy, September 9 - 11, 1998, pages 113–118, Oxford, UK, 1998. Pergamon.

[58] Angelo Morzenti, Pierluigi San Pietro, and Sandro Morasca. A tool for automated system analysis based on modular specifications. In *Proceedings of the 13th IEEE international conference on Automated software engineering*, ASE '98, Honolulu, HI, USA, October 13 - 16, 1998, pages 2–11, Washington, DC, USA, 1998. IEEE Computer Society.

[59] Sandro Morasca. Towards a way to deal with continuous attributes in decision trees (poster paper). In *Proceedings of the 11th international conference on Software engineering and knowledge engineering*, SEKE '99, Kaiserslautern, Germany, June 16-19, 1999, pages 262 – 266, Skokie, IL, USA, 1999. Knowledge Systems Institute.

[60] Sandro Morasca. Measuring attributes of concurrent software specifications in Petri nets. In *Proceedings of the 6th International Symposium on Software Metrics*, IEEE METRICS '99, Boca Raton, FL, USA, November 4-6, 1999, pages 100–110, Washington, DC, USA, 1999. IEEE Computer Society.

[61] Sandro Morasca and Giuliano Russo. An empirical study of software productivity. In *Proceedings of the 25th International Computer Software and Applications Conference on Invigorating Software Development*, COMPSAC '01, Chicago, IL, USA, October 8-12, 2001, pages 317–322, Washington, DC, USA, 2001. IEEE Computer Society.

[62] Giovanni Denaro, Sandro Morasca, and Mauro Pezzè. Deriving models of software fault-proneness. In *Proceedings of the 14th international conference on Software engineering and knowledge engineering*, SEKE '02, Ischia, Italy, July 15-19, 2002, pages 361–368, New York, NY, USA, 2002. ACM.

[63] Sandro Morasca. A proposal for using continuous attributes in classification trees. In *Proceedings of the 14th international conference on Software engineering and knowledge engineering*, SEKE '02, Ischia, Italy, July 15-19, 2002, pages 417–424, New York, NY, USA, 2002. ACM.

[64] Luciano Baresi, Sandro Morasca, and Paolo Paolini. An empirical study on the design effort of Web applications. In *Proceedings of the 3rd International Conference on Web Information Systems Engineering*, WISE '02, Singapore, Singapore, December 11-14, 2002, pages 345–354, Washington, DC, USA, 2002. IEEE Computer Society.

[65] Sandro Morasca. Foundations of a weak measurement-theoretic approach to software measurement. In *Proceedings of the 6th international conference on Fundamental approaches to software engineering*, FASE'03, Held as Part of the Joint European Conferences on Theory and Practice of Software, ETAPS 2003, Warsaw, Poland, April 7-11, 2003. Lecture Notes in Computer Science, volume 2621 (Mauro Pezzè ed.), pages 200–215, Berlin, Heidelberg, 2003. Springer-Verlag.

[66] Luigi Lavazza, Sandro Morasca, and Angelo Morzenti. A dual language approach extension to UML for the development of time-critical component-based systems. *Electron. Notes Theor. Comput. Sci.*, 82, Proceedings of TACoS 2003, Warsaw, Poland, April 13, 2003:121–132, 2003.

[67] Sandro Morasca. A Bayesian approach to software testing evaluation. In *Proceedings of the 14th international conference on Software engineering and knowledge engineering*, SEKE '03, San Francisco Bay, CA, USA, July 1-3, 2003, pages 706–713, Skokie, IL, USA, 2003. Knowledge Systems Institute.

[68] Luciano Baresi, Sandro Morasca, and Paolo Paolini. Estimating the design effort of Web applications. In *Proceedings of the 9th International Symposium on Software Metrics*, METRICS 2003, September 3-5, 2003, Sydney, Australia, pages 62–72, Washington, DC, USA, 2003. IEEE Computer Society.

[69] Jeffrey Carver, Maria Letizia Jaccheri, Sandro Morasca, and Forrest Shull. Issues in using students in empirical studies in software engineering education. In *9th IEEE International Software Metrics Symposium*, METRICS 2003, September 3-5, 2003, Sydney, Australia, pages 239–249, Washington, DC, USA, 2003. IEEE Computer Society.

[70] Luigi Lavazza, Sandro Morasca, and Angelo Morzenti. A dual language approach to the development of time-critical systems. *Electron. Notes Theor. Comput. Sci.*, 116, Proceedings of TACOS 2004, Barcelona, Spain, March 27 - 28, 2004:227–239, January 2005.

[71] Sandro Morasca and Stefano Serra-Capizzano. On the analytical comparison of testing techniques. *SIGSOFT Software Engineering Notes*, 29 (4):154–164, Proceedings of the ACM/SIGSOFT International Symposium on Software Testing and Analysis, ISSTA 2004, Boston, MA, USA, July 11–14, 2004, July 2004.

[72] Sandro Morasca. On the definition and use of aggregate indices for nominal, ordinal, and other scales. In *10th IEEE International Software Metrics Symposium*, METRICS 2004, September 11-17, 2004, Chicago, IL, USA, pages 46–57, Washington, DC, USA, 2004. IEEE Computer Society.

[73] Luciano Baresi, Piero Fraternali, Massimo Tisi, and Sandro Morasca. Towards model-driven testing of a Web application generator. In *Proceedings of the 5th International Conference on Web Engineering*, ICWE 2005, Sydney, Australia, July 27-29, 2005. Lecture Notes in Computer Science, volume 3579 (David Lowe and Martin Gaedke eds.), pages 75–86, Berlin, Heidelberg, 2005. Springer-Verlag.

[74] Letizia Jaccheri and Sandro Morasca. Understanding the role of software metrics in an empirical software engineering course for PhD students. In *Workshop on "Methods for Learning Metrics," colocated with METRICS 2005*, September 19, 2005, Como, Italy.

[75] Letizia Jaccheri and Sandro Morasca. On the importance of dialogue with industry about software engineering education. In *Proceedings of the 2006 international workshop on Summit on software engineering education*, SSEE '06, May 20, 2006, Shanghai, China, pages 5–8, New York, NY, USA, 2006. ACM.

[76] Sandro Morasca. Courses for software professionals as two-way communication channels between academia and industry. In *Proceedings of the 2006 international workshop on Summit on software engineering education*, SSEE '06, May 20, 2006, Shanghai, China, pages 25–28, New York, NY, USA, 2006. ACM.

[77] Letizia Jaccheri and Sandro Morasca. Involving industry professionals in empirical studies with students. In *Proceedings of the 2006 international conference on Empirical software engineering issues: critical assessment and future directions*, Dagstuhl Castle, Germany, June 26-30, 2006. Lecture Notes in Computer Science, volume 4336 (Victor R. Basili, H. Dieter Rombach, Kurt Schneider, Barbara A. Kitchenham, Dietmar Pfahl, and Richard W. Selby eds.), page 152, Berlin, Heidelberg, 2007. Springer-Verlag.

[78] Austen Rainer, Marcus Ciolkowski, Dietmar Pfahl, Barbara Kitchenham, Sandro Morasca, Matthia M. Müller, Guilherme H. Travassos, and Sira Vegas. Teaching empirical methods to undergraduate students working group results. In *Proceedings of the 2006 international conference on Empirical software engineering issues: critical assessment and future directions*, Dagstuhl Castle, Germany, June 26-30, 2006. Lecture Notes in Computer Science, volume 4336 (Victor R. Basili, H. Dieter Rombach, Kurt Schneider, Barbara A. Kitchenham, Dietmar Pfahl, and Richard W. Selby eds.), pages 158 –162, Berlin, Heidelberg, 2007. Springer-Verlag.

[79] Sandro Morasca. On the assessment of the mean failure frequency of software in late testing. In *Proceedings of the International Conference on Software Engineering Advances*, ICSEA 2006, Papeete, Tahiti, French Polynesia, October 28 - November 2, 2006, pages 15–, Washington, DC, USA, 2006. IEEE Computer Society.

[80] Davide Taibi, Luigi Lavazza, and Sandro Morasca. OpenBQR: a framework for the assessment of OSS. In *Open Source Development, Adoption and Innovation, IFIP Working Group 2.13 on Open Source Software Systems*, OSS 2007, Limerick, Ireland, June 11-14, 2007. IFIP Advances in Information and Communication Technology, volume 234 (Joseph Feller, Brian Fitzgerald, Walt Scacchi, and Alberto Sillitti eds.), pages 173–186. Springer, 2007.

[81] José A. Cruz-Lemus, Marcela Genero, Sandro Morasca, and Mario Piattini. Using practitioners for assessing the understandability of uml statechart diagrams with composite states. In *Advances in Conceptual Modeling - Foundations and Applications, ER 2007 Workshops CMLSA, FP-UML, ONISW, QoIS, RIGiM,SeCoGIS*, QoIS, Auckland, New Zealand, November 5-9, 2007. Lecture Notes in Computer Science, volume 4802 (Jean-Luc Hainaut, Elke A. Rundensteiner, Markus Kirchberg, Michela Bertolotto, Mathias Brochhausen, Yi-Ping Phoebe Chen, Samira Si-Said Cherfi, Martin Doerr, Hyoil Han, Sven Hartmann, Jeffrey Parsons, Geert Poels, Colette Rolland, Juan Trujillo, Eric S. K. Yu, and Esteban Zimányi eds.), pages 213–222. Springer, 2007.

[82] Sandro Morasca and Alberto Sillitti. 1st international workshop on Trust in Open Source Software (TOSS). In *Open Source Development, Adoption and Innovation, IFIP Working Group 2.13 on Open Source Software Systems*, OSS 2007, Limerick, Ireland, June 11-14, 2007. IFIP Advances in Information and Communication Technology, volume 234 (Joseph Feller, Brian Fitzgerald, Walt Scacchi, and Alberto Sillitti eds.), pages 371–373. Springer, 2007.

[83] Sandro Morasca. Subjective assessment of the mutual influence of ISO 9126 software qualities: an empirical study. In *Proceedings of the 20th international conference on Software engineering and knowledge engineering*, SEKE '08, San Francisco, CA, USA, July 1-3, 2008, pages 297–302, Skokie, IL, USA, 2008. Knowledge Systems Institute.

[84] Davide Taibi, Vieri Del Bianco, Davide Dalle Carbonare, Luigi Lavazza, and Sandro Morasca. Towards the evaluation of OSS trustworthiness: Lessons learned from the observation of relevant OSS projects. In *Open Source Development, Communities and Quality, IFIP 20th World Computer Congress, Working Group 2.3 on Open Source Software Systems*, OSS 2008, Milan, Italy, September 7-10, 2008. IFIP Advances in Information and Communication Technology, volume 275 (Barbara Russo, Ernesto Damiani, Scott A. Hissam, Björn Lundell, and Giancarlo Succi eds.), pages 389–395. Springer, 2008.

[85] Sandro Morasca. Refining the axiomatic definition of internal software attributes. In *Proceedings of the Second ACM-IEEE international symposium on Empirical software engineering and measurement*, ESEM '08, October 9-10, 2008, Kaiserslautern, Germany, pages 188–197, New York, NY, USA, 2008. ACM.

[86] Sandro Morasca, Davide Taibi, and Davide Tosi. Towards certifying the testing process of open-source software: New challenges or old methodologies? In *Proceedings of the 2009 ICSE Workshop on Emerging Trends in Free/Libre/Open Source Software Research and Development*, FLOSS '09, Vancouver, BC, Canada, May 16, 2009, pages 25–30, Washington, DC, USA, 2009. IEEE Computer Society.

[87] Sandro Morasca. Building statistically significant robust regression models in empirical software engineering. In *Proceedings of the 5th International Conference on Predictor Models in Software Engineering*, PROMISE '09, Vancouver, BC, Canada, May 18 - 19, 2009, pages 19:1–19:10, New York, NY, USA, 2009. ACM.

[88] Massimiliano Di Penta, Sandro Morasca, and Alberto Sillitti. 3rd international workshop on designing empirical studies: Assessing the effectiveness of agile methods (IWDES 2009). In *Agile Processes in Software Engineering and Extreme Programming, 10th International Conference*, XP '09, Pula, Italy, May 25-29, 2009. Lecture Notes in Business Information Processing, volume 31, pages 234–235. Springer, 2009.

[89] Vieri del Bianco, Luigi Lavazza, Sandro Morasca, and Davide Taibi. Quality of open source software: The QualiPSo trustworthiness model. In *Open Source Ecosystems: Diverse Communities Interacting, 5th IFIP WG 2.13 International Conference on Open Source Systems*, OSS 2009, Skövde, Sweden, June 3-6, 2009. IFIP Advances in Information and Communication Technology, volume 299 (Cornelia Boldyreff, Kevin Crowston, Björn Lundell, and Anthony I. Wasserman eds.), pages 199–212. Springer, 2009.

[90] Luigi Lavazza, Sandro Morasca, Davide Taibi, and Davide Tosi. Verso un processo di certificazione dei portali web di progetti software Open Source (in italian). In *Proceedings of the III Conferenza Italiana sul Software Libero*, CONFSL '09, June 12-13, 2009, Bologna, Italy, 2009.

[91] Sandro Morasca. A probability-based approach for measuring external attributes of software artifacts. In *Proceedings of the 2009 3rd International Symposium on Empirical Software Engineering and Measurement*, ESEM '09, Lake Buena Vista, FL, USA, October 15-16, 2009, pages 44–55, Washington, DC, USA, 2009. IEEE Computer Society.

[92] Sandro Morasca. On the use of weighted sums in the definition of measures. In *Proceedings of the 2010 ICSE Workshop on Emerging Trends in Software Metrics*, WETSoM '10, Cape Town, South Africa, May 4, 2010, pages 8–15, New York, NY, USA, 2010. ACM.

[93] Vieri del Bianco, Luigi Lavazza, Sandro Morasca, Davide Taibi, and Davide Tosi. The QualiSPo approach to OSS product quality evaluation. In *Proceedings of the 3rd International Workshop on Emerging Trends in Free/Libre/Open Source Software Research and Development*, FLOSS '10, Cape Town, South Africa, May 8, 2010, pages 23–28, New York, NY, USA, 2010. ACM.

[94] Vieri Del Bianco, Luigi Lavazza, Sandro Morasca, Davide Taibi, and Davide Tosi. A survey on the importance of some economic factors in the adoption of open source software. In *Software Engineering Research, Management and Applications 2010: selected papers from the 8th ACIS International Conference on Software Engineering Research, Management and Applications*, SERA 2010, Montreal, QC, Canada, May 24-26, 2010. Studies in Computational Intelligence, volume 296 (Roger Y. Lee, Olga Ormandjieva, Alain Abran, and Constantinos Constantinides eds.), pages 151–162. Springer, 2010.

[95] Vieri del Bianco, Luigi Lavazza, Sandro Morasca, Davide Taibi, and Davide Tosi. An investigation of the users' perception of OSS quality. In *Open Source Software: New Horizons 6th International IFIP WG 2.13 Conference on Open Source Systems*, OSS 2010, Notre Dame, IN, USA, May 30 - June 2, 2010. Proceedings. IFIP Advances in Information and Communication Technology, volume 319 (Pär Ågerfalk, Cornelia Boldyreff, Jesús M. González-Barahona, Gregory R. Madey, and John Noll eds.), pages 15–28. Springer, 2010.

[96] Sandro Morasca, Davide Taibi, and Davide Tosi. T-DOC: a tool for the automatic generation of testing documentation for OSS products. In *Open Source Software: New Horizons 6th International IFIP WG 2.13 Conference on Open Source Systems*, OSS 2010, Notre Dame, IN, USA, May 30 - June 2, 2010. Proceedings. IFIP Advances in Information and Communication Technology, volume 319 (Pär Ågerfalk, Cornelia Boldyreff, Jesús M. González-Barahona, Gregory R. Madey, and John Noll eds.), pages 200–213. Springer, 2010.

[97] Luigi Lavazza, Sandro Morasca, Davide Taibi, and Davide Tosi. Applying SCRUM in an OSS development process: An empirical evaluation. In *Proceedings of Agile Processes in Software Engineering and Extreme Programming, 11th International Conference*, XP '10, Trondheim, Norway, June 1-4, 2010. Lecture Notes in Business Information Processing, volume 48 (Alberto Sillitti, Angela Martin, Xiaofeng Wang, and Elizabeth Whitworth eds.), pages 147–159. Springer, 2010.

[98] Matteo Melideo, Gabriele Ruffatti, Sergio Oltolina, Davide Dalle Carbonare, Alberto Sillitti, Etiel Petrinja, Giancarlo Succi, Sandro Morasca, Davide Taibi, Davide Tosi, Luigi Lavazza, Gerardo Canfora, and Eugenio Zimeo. Il centro di competenza italiano per l'open source e la nuova rete globale di centri di competenza FLOSS (in italian). In *Proceedings of the IV Conferenza Italiana sul Software Libero*, CONFSL '10, June 11-12, 2010, Cagliari, Italy, 2010.

[99] Sandro Morasca, Michele Proto, and Davide Taibi. The evolution of defects and patches in sourceforge projects: A quantitative analysis. In *Proceedings of the IV Conferenza Italiana sul Software Libero*, CONFSL '10, June 11-12, 2010, Cagliari, Italy, 2010.

[100] Luigi Lavazza and Sandro Morasca. A study of non-linearity in the statistical convertibility of function points into cosmic function points. In *Proceedings of the Workshop on Advances in Functional Size Measurement and Effort Estimation*, ECOOP '10, June 21, 2010, Maribor, Slovenia, 2010.

[101] Luigi Lavazza, Sandro Morasca, Davide Taibi, and Davide Tosi. Predicting OSS trustworthiness on the basis of elementary code assessment (short paper). In *Proceedings of the 2010 ACM-IEEE International Symposium on Empirical Software Engineering and Measurement*, ESEM '10, September 16-17, 2010, Bolzano/Bozen, Italy, pages 36:1–36:4, New York, NY, USA, 2010. ACM.

[102] Barbara Carminati, Elena Ferrari, Sandro Morasca, and Davide Taibi. A probability-based approach to modeling the risk of unauthorized propagation of information in on-line social networks. In *Proceedings of the First ACM Conference on Data and Application Security and Privacy*, CODASPY '11, February 21-23, 2011, San Antonio, TX, USA, pages 51–62, 2011.

[103] Gabriele Basilico, Luigi Lavazza, Sandro Morasca, Davide Taibi, and Davide Tosi. OP2A: Assessing the quality of the portal of open source software products. In *Proceedings of the Seventh International Conference on Web Information Systems and Technologies*, WEBIST '11, May 6-9, 2011, Noordwijkerhout, The Netherlands, pages 184–193. SciTePress, 2011.

[104] Luigi Lavazza, Sandro Morasca, Davide Taibi, and Davide Tosi. An empirical investigation of perceived reliability of open source java programs. In *Proceedings of the 27th Symposium On Applied Computing*, SAC '12, March 26-30, 2012, Riva del Garda, Italy, pages 1109–1114, New York, NY, USA, 2012. ACM.

[105] Luigi Lavazza and Sandro Morasca. Software effort estimation with a generalized robust linear regression technique. In *Proceedings of the 16th International Conference on Evaluation and Assessment in Software Engineering*, EASE 2012, May 14-15, 2012, Ciudad Real, Spain, pages 206–215, 2012.

[106] Vieri Del Bianco, Luigi Lavazza, and Sandro Morasca. A proposal for simplified model-based cost estimation models. In *Proceedings of the 13th Product-focused Software Process Improvement*, PROFES 2012, Madrid, Spain, June 14-15, 2012. Lecture Notes in Computer Science, volume 7343 (Oscar Dieste, Andreas Jedlitschka, and Natalia Juristo eds.), pages 59–73, London, UK, 2012. Springer-Verlag.

[107] Vieri Del Bianco, Luigi Lavazza, Valentina Lenarduzzi, Sandro Morasca, Davide Taibi, and Davide Tosi. A study on OSS marketing and communication strategies. In *Proceedings of Open Source Software: Long-Term Sustainability - 8th International IFIP WG 2.13 Conference on Open Source Systems*, OSS 2012, Hammamet, Tunisia, September 10-13, 2012, pages 338–343. Springer, 2012.

[108] Luigi Lavazza, Sandro Morasca, Davide Taibi, and Davide Tosi. On the definition of dynamic software measures. In *Accepted for Publication in Proceedings of the 2012 ACM-IEEE International Symposium on Empirical Software Engineering and Measurement*, ESEM '12, September 19-20, 2012, Lund, Sweden, 2012.

[109] Luigi Lavazza, Sandro Morasca, and Gabriela Robiolo. An empirical evaluation of effort prediction models based on functional size measures. In *Accepted for Publication in Proceedings of the International Conference on Software Engineering Advances*, ICSEA 2012, November 18-23, 2012, Lisbon, Portugal, 2012.

[110] Sandro Morasca. Data analysis anti-patterns in empirical software engineering. In *Proceedings of the 2013 ICSE Workshop on Data Analysis Patterns in Software Engineering*, DAPSE '13, San Francisco, USA, May 21, 2013, pages 9–10, 2013.

[111] Vieri Del Bianco, Luigi Lavazza, Geng Liu, Sandro Morasca, and Abedallah Zaid Abualkishik. Model-based simplified functional size measurement - an experimental evaluation with cosmic function points. In *EESSMOD@MoDELS*, pages 13–22, 2013.

[112] Michail Giannakos, Letizia Jaccheri, and Sandro Morasca. An empirical examination of behavioral factors in creative development of game prototype. In *To Appear in the Proceedings of the 12th International Conference on Entertainment Computing*, ICEC 2013, Sao Paulo, Brazil, October 16-18, 2013, 2013.

[113] Luigi Lavazza and Sandro Morasca. Measuring the functional size of real-time and embedded software: a comparison of function point analysis and cosmic. In *Proceedings of the International Conference on Software Engineering Advances*, ICSEA 2013, October 27 - November 1, 2013, Venice, Italy, 2013.

[114] Abbas Tahir, Sandro Morasca, and Davide Tosi. Towards probabilistic models to predict availability, accessibility and successability of web services. In *Proceedings of the International Conference on Software Engineering Advances*, ICSEA 2013, October 27 - November 1, 2013, Venice, Italy, 2013.

[115] Sandro Morasca. Using logistic regression to estimate the number of faulty software modules. In *Proceedings of the 18th International Conference on Evaluation and Assessment in Software Engineering*, EASE '14, May 13-14, 2014, pages 26:1–26:9, New York, NY, USA, 2014. ACM.

[116] Valentina Lenarduzzi, Sandro Morasca, and Taibi Davide. Estimating software development effort based on phases. In *Accepted for publication in EUROMICRO-SEAA 2014*, 40th Euromicro Conference on Software Engineering and Advanced Applications, SEAA 2014, Verona, Italy, August 27-29, 2014, 2014.

[117] Luigi Lavazza, Sandro Morasca, and Davide Tosi. On the ability of functional size measurement methods to size complex software applications. In *Accepted for publication in Proceedings of the International Conference on Software Engineering Advances*, ICSEA 2014, October 12 - 16, 2014, Nice, France, 2014.

11.5 Master's and PhD Theses

[118] Roberto Lecciso, Stefano Mainetti, and Sandro Morasca. *Metriche del software: un'esposizione critico-sperimentale e un'applicazione al Pascal (in Italian).* Master's Thesis (Italian "tesi di laurea") in Electronic Engineering (Italian "Ingegneria Elettronica," defended on July 19, 1985. Politecnico di Milano, 1985.

[119] Sandro Morasca. *Un modello e un ambiente per la specifica e la verifica di sistemi concorrenti e in tempo reale (in Italian).* PhD Dissertation (Italian "Tesi di Dottorato") in Electrical Engineering for Computer Science and Systems (Italian "Ingegneria Elettronica dell'Informazione e dei Sistemi," defended on September 23, 1991. Politecnico di Milano, Dipartimento di Elettronica, 1991.

12 Contact

Sandro Morasca
Università degli Studi dell'Insubria
Dipartimento di Scienze Teoriche e Applicate
Via Oriani 5
I-22100 Como, Italy
tel.: 031/2386228
e-mail: sandro.morasca@uninsubria.it