Proposing and empirically validating change impact analysis metrics

Arvanitou, Elvira Maria

university of groningen

Proposing and Empirically Validating Change Impact Analysis Metrics

PhD Thesis

to obtain the degree of PhD at the
University of Groningen
on the authority of the
Rector Magnificus Prof. E. Sterken
and in accordance with
the decision by the College of Deans.

This thesis will be defended in public on

Friday 13 July 2018 at 09.00 hours

by

Elvira Maria Arvanitou

born on 16 September 1988
in Thessaloniki, Griekenland

Supervisors
Prof. P. Avgeriou
Prof. A.N. Chatzigeorgiou

Co-supervisor
Prof. A. Ampatzoglou PhD

Assessment Committee
Prof. A.C. Telea
Prof. F. Arcelli Fontana
Prof. T. Mens

ISBN: 978-94-034-0752-4

to my baby….

Samenvatting

Bij het onderhoud van de software is de software onderhevig aan veranderingen als gevolg van het verhelpen van fouten, veranderende eisen, aanvullende eisen, enz. Het belang van het laag houden van de onderhoudskosten is benadrukt in de literatuur met empirisch bewijs, waaruit blijkt dat de kosten van deze fase ongeveer 50%-75% uitmaken van de totale kosten van softwareontwikkeling. Deze kosten kunnen nog verder oplopen als de software wordt gekenmerkt door (a) ***veranderingsgevoeligheid***, d.w.z. de waarschijnlijkheid dat een softwareartefact om interne redenen verandert (d.w.z. het oplossen van bugs in dat artefact of het wijzigen van eisen met betrekking tot het artefact); en (b) ***instabiliteit***, d.w.z. de waarschijnlijkheid dat een softwareartefact verandert als gevolg van veranderingen in andere artefacten van het systeem.

Het proces dat veranderingsgevoeligheid en instabiliteit onderzoekt wordt ***change impact analysis*** genoemd; dit proces is niet alleen belangrijk tijdens het onderhoud, maar ook tijdens de andere ontwikkelingsfasen, zoals requirements engineering, ontwerp, implementatie en testen. Change impact analysis is gebaseerd op de kwantificering van veranderingsgevoeligheid en instabiliteit, om beslissingen te kunnen nemen over welke veranderingen moeten worden uitgevoerd, en hoe en wanneer. In de literatuur zijn echter de volgende beperkingen vastgesteld met betrekking tot de praktijken van effectbeoordelingen van veranderingen:

- In de fasen van het ontwerp van het architecturale ontwerp en de behoeftetechniek ontbreken meetwaarden die de instabiliteit van het artefact en de gevoeligheid van de verandering bepalen. Daarom kan de impactanalyse in die fasen niet kwantitatief zijn.
- De implementatie- en detailontwerpfasen worden ondersteund door meetwaarden voor instabiliteit en veranderingsgevoeligheid, maar dergelijke meetwaarden zijn niet nauwkeurig omdat ze de bovengenoemde parameters niet combineren.

- Er is een gebrek aan hulpmiddelen die het proces van het berekenen van veranderingsgevoeligheid en instabiliteitsmetriek voor implementatie en gedetailleerde ontwerpfasen kunnen automatiseren.

Om de bovenvermelde beperkingen aan te pakken, hebben ***we*** als ***onderdeel van dit doctoraat een reeks methoden en instrumenten voorgesteld die zowel instabiliteit als veranderingsgevoeligheid met verhoogde nauwkeurigheid kunnen kwantificeren in drie belangrijke ontwikkelingsfasen*** (d.w.z. requirements engineering, ontwerp en implementatie).

Als eerste stap om dit doel te bereiken, hebben we eerst de literatuur bestudeerd om ontwerp-tijd kwaliteitsattributen en meetwaarden te onderzoeken, die kunnen worden gebruikt om ze te kwantificeren. Specifiek hebben we een corpus van meer dan 150 primaire studies beoordeeld. De belangrijkste bevindingen benadrukken het belang van veranderingsgevoeligheid en instabiliteit in de huidige onderzoeksliteratuur. Met name stabiliteit en veranderingsgevoeligheid zijn de meest bestudeerde kwaliteitsattributen, na onderhoudbaarheid (respectievelijk 15 en 18 studies). De studie bevestigde echter het gebrek aan meetwaarden in verschillende ontwikkelingsfasen (vooral in eisen en architectuur); de meeste beschikbare meetwaarden bevinden zich op het broncodeniveau.

Gebaseerd op de belangrijkste bevinding van de vorige studie, hebben we als volgende stap de mogelijkheid van codemeetwaarden onderzocht om toepasbaar te zijn op artefacten in een andere ontwikkelingsfase, namelijk architectuur. Daarom onderzochten we meerdere codemeetwaarden, met betrekking tot hun vermogen om fijn- en grofkorrelige veranderingen in softwareonderhoud vast te leggen. Intuïtief passen meetwaarden die in staat zijn om grofkorrelige veranderingen vast te leggen beter bij het architectuurniveau; integendeel, fijnkorrelige veranderingen moeten worden verwaarloosd op het architectuurniveau. De empirische evaluatie suggereerde dat sommige meetwaarden gevoeliger zijn voor veranderingen, en dus meer geschikt zijn voor evaluaties op methodiek- en klasseniveau, terwijl andere stabieler zijn, d.w.z. dat ze grootschaliger veranderingen vereisen om hun

waarden te beïnvloeden, en dus meer geschikt zijn voor het architectuurniveau (bv. pakketten, componenten, enz.). Op basis van de resultaten is de enige meetwaarde die gerelateerd is aan instabiliteit en in staat is om grofkorrelige veranderingen vast te leggen de Response voor a Class (RFC)-meetwaarde, door klasse-niveaumeetwaarden samen te voegen tot het architectuurniveau met behulp van de gemiddelde (AVG)-functie. RFC combineert koppeling (d.w.z. gebruik van de openbare interface van andere klassen) en grootte (d.w.z., aantal lokale methoden) eigenschappen en daarom kan het niet worden beschouwd als een zuivere veranderbaarheid of instabiliteitsmeetwaarde. Daarom wordt het voorstel van nieuwe gevoeligheids- en instabiliteit meetwaarden voor veranderingen in de architectuur noodzakelijk geacht.

Voortbouwend op de resultaten van de hierboven genoemde studies, zijn we overgegaan tot de belangrijkste bijdragen van deze thesis, namelijk *het voorstel van methoden (voor berekening van de meetwaarden) en de ontwikkeling van overeenkomstige tools voor het kwantificeren van veranderingsgevoeligheid en instabiliteit in de behoefte-, architectuur- en implementatiefase.* In het bijzonder hebben we methoden gedefinieerd die rekening houden met zowel instabiliteit als veranderings vatbaarheidsstatistieken, het verstrekken van een uniforme meetwaarde die kwantitatief de veranderingsimpact analyse kan leiden. Aanvankelijk richtten we ons op het broncodeniveau, waarvoor we twee maatstaven hebben voorgesteld: de ***Ripple Effect Measure*** (REM) voor het vastleggen van de instabiliteit van klassenafhankelijkheden, en de ***Change Proneness Measure*** (CPM) voor het kwantificeren van de vatbaarheid voor klassenwisselingen. Beide meeteenheden zijn empirisch gevalideerd op open-sourcesoftwareprojecten (OSS) door hun beoordelingsvermogen te vergelijken met bestaande meetwaarden. De validatie is uitgevoerd op basis van de 1061-1998 IEEE Standard for Software Quality criteria. De resultaten van de studie suggereerden dat de voorgestelde meetwaarden betere voorspellers van veranderingsgevoeligheid en instabiliteit zijn dan de bestaande. In het bijzonder heeft REM een 38% sterkere correlatie vergeleken met de beste voorspeller van instabiliteit in de literatuur (dat wil zeggen, koppeling tussen objecten), terwijl CPM een 48% sterkere correlatie vertoont in vergelijking met de beste voorspeller van veranderbaarheid in de literatuur (dat wil zeggen, berichtdoorgangskoppeling). Daarnaast bieden we bewijs dat een meetwaarde die zowel veranderbaarheid en instabiliteit combineert een hogere

nauwkeurigheid biedt in vergelijking met het gebruik van de twee factoren afzonderlijk.

Ten slotte hebben we de bovengenoemde meetwaarden als uitgangspunt genomen en deze aangepast aan het architecturale ontwerp- en behoeftepeil. De voorgestelde meetwaarde: ***Module Change Proneness Measure*** (MCPM) en ***Requirement Ripple Effect Metric*** (R2EM) zijn empirisch gevalideerd in respectievelijk een OSS- en een industriële context. De evaluatie van beide meetwaarden is net als voorheen zeer positief. Enerzijds is MCPM gemiddeld 23% een nauwkeurigere voorspeller in vergelijking met bestaande pakketstatistieken (bijv. Efferente en Afferente Koppeling). Aan de andere kant heeft R2EM bewezen sterk gecorreleerd te zijn (ca. 60%) met het deskundig oordeel van software engineers.

Voor alle vier bovengenoemde meetwaarden hebben we tools ontwikkeld die hun berekening kunnen automatiseren op basis van bestaande software artefacten, waardoor we de schaal van onze empirische evaluaties kunnen vergroten (waardoor we meer vertrouwen krijgen in de resultaten) en de mogelijke toepasbaarheid van de voorgestelde methoden in de praktijk kunnen vergroten.

Abstract

Along software maintenance the software is subject to changes due to bug fixing, changing requirements, additional requirements, etc. The importance of keeping the cost of maintenance low has been highlighted in the literature with empirical evidence, suggesting that the cost of this phase is approximately 50%-75% of the total cost of software development. This cost can be further increased if the software is characterized by (a) ***change proneness***, i.e., the probability of a software artifact to change due to the internal reasons (e.g., fixing bugs in that artifact or changing requirements related to the artifact); and (b) ***instability***, i.e., the probability of a software artifact to change due to changes in other artifacts of the system.

The process that investigates change-proneness and instability is called ***change impact analysis***; this process is important not just during maintenance but also during the other development phases, e.g. requirements engineering, design, implementation, and testing. Change impact analysis is based on the quantification of change proneness and instability, in order to make decisions on which changes to perform, how and when. However, the literature has identified the following limitations on change impact analysis practices:

- The architectural design and requirements engineering phases are completely lacking metrics that capture artifact instability and change proneness. Therefore, change impact analysis at those phases cannot be quantitative.
- The implementation and detailed-design phases are supported by metrics for instability and change proneness, but such metrics lack accuracy, since they do not combine the aforementioned parameters.
- There is a lack of tools that can automate the process of calculating change proneness and instability metrics for implementation and detailed-design phases.

To tackle the aforementioned limitations, as part of this PhD, ***we*** have ***proposed a set of methods and tools that can quantify both instability and change proneness with increased accuracy at three main development phases*** (i.e. requirements engineering, design, and implementation).

As a first step to achieve this goal, we have first reviewed the literature to investigate design-time quality attributes and metrics that can be used to quantify them. Specifically we have reviewed a corpus of more than 150 primary studies. The main findings highlight the importance of change proneness and instability in the current research literature. In particular, stability and change proneness are the most frequently studied quality attributes, after maintainability (15 and 18 studies, respectively). However, the study confirmed the lack of metrics in different development phases (especially in requirements and architecture); the majority of available metrics are at the source-code level.

Based on the main finding of the previous study, as a next step we explored the ability of code metrics to be applicable at artifacts in a different development phase, namely architecture. Thus, we investigated multiple code metrics, with respect to their ability to capture fine- and coarse-grained changes along software maintenance. Intuitively, metrics that are able to capture coarse-grained changes are more fitting to the architecture level; on the contrary fine-grained changes should be neglected at the architecture level. The empirical evaluation suggested that some metrics are more sensitive to changes, and are thus more fitting for method and class level assessments, whereas others are more stable, i.e., they require larger-scale changes for their values to be affected, and are thus more fitting for the architecture level (e.g., packages, components, etc.). Based on the results, the only metric that is related to instability and is able to capture coarse-grained changes is the Response for a Class (RFC) metric, by aggregating class-level metrics to the architecture-level with the use of the average (AVG) function. However, RFC combines coupling (i.e., use of other classes' public interface) and size (i.e., number of local methods) properties, and therefore it cannot be considered as a pure change proneness or instability metric. Thus, the proposal of novel change proneness and instability metrics for architecture is considered necessary.

Building on the results of the aforementioned studies, we proceeded to the main contributions of this thesis, i.e., the *proposal of methods (for metrics calculation) and the development of corresponding tools for quantifying change proneness and instability at the requirements, architecture, and implementation phase*. In particular, we defined methods that consider both instability and

change proneness metrics, providing a unified metric that can quantitatively guide change impact analysis. Initially, we focused on the source-code level, for which we proposed two metrics: the ***Ripple Effect Measure*** (REM) for capturing class dependencies instability, and the ***Change Proneness Measure*** (CPM) for quantifying class change proneness. Both metrics have been empirically validated on open source software (OSS) projects, by comparing their assessment power to existing metrics. The validation has been performed based on the 1061-1998 IEEE Standard for Software Quality Metrics. The results of the study suggested that the proposed metrics are better predictors of change proneness and instability compared to existing ones. In particular REM has 38% stronger correlation compared to the best predictor of instability in the literature (i.e., Coupling Between Objects), whereas CPM shows 48% stronger correlation compared to the best predictor of change proneness in the literature (i.e., Message Passing Coupling). In addition to that, we provide evidence that a metric combining both change proneness and instability offer higher accuracy, compared to using the two factors in isolation.

Finally, using the aforementioned metrics as a starting point, we tailored them to fit the architectural design and requirements level. The proposed metrics: ***Module Change Proneness Measure*** (MCPM) and ***Requirement Ripple Effect Metric*** (R2EM) have been empirically validated in an OSS and an industrial context, respectively. Similarly as before, the evaluation of both metrics has been very positive. On the one hand, MCPM is on average 23% a more accurate predictor compared to existing package metrics (e.g., Efferent and Afferent Coupling). On the other hand, R2EM has proven to be strongly correlated (approx. 60%) to the expert opinion of software engineers.

For all four aforementioned metrics we have developed tools that can automate their calculation from existing software artifacts, enabling us to expand the scale of our empirical evaluations (increasing our confidence of the results), and increasing the possible applicability of the proposed methods in practice.

Table of Contents

Acknowledgements

Conducting a PhD research is a challenging, but at the same time a highly constructive process. Now that this process is almost completed, I need to acknowledge that this thesis is a collective outcome of the forces and support of many people that were involved in this research per se, but also in the underlying process. To these people I would like to express my sincere gratitude.

First of all, I would like to thank my supervisors' team for the research and ethical support that they have provided me throughout this endeavor. In particular, I would like to thank my supervisor, Prof. Paris Avgeriou for giving me the chance to start this project and investigate this very interesting topic. His wide knowledge was precious while guiding my research and finalizing the thesis itself. Additionally, I would like to thank Prof. Alexander Chatzigeorgiou for his trust throughout our time-lasting collaboration. His active participation in the course of this project and his research guidance were really helpful, especially in the first years when he helped me understand aspects of the research and the domain that at that point were unclear to me.

Last, but not least, I could not omit from this section Apostolis Ampatzoglou, who is the person that, among others, 'persuaded' me to start this PhD project. Apostolis guided my research all these years, devoting to me large portion of his valuable time. A big thanks for the endless hours that he 'spent' with me either through Skype calls or face-to-face meetings. Along with Apostoli I have started getting to know what research is, in my BSc thesis, continued to research with him in my MSc thesis, and of course during my PhD. All these years I have learned how to perform research properly, but most importantly through our collaboration, I was continuously motivated, inspired and eventually ended-up to love researching. I would also like to thank him for the help that he has provided me all these years in all aspects of the PhD process: when he was always willing and available to discuss with me anything that might be a problem for me, for his understanding all the difficult periods of the project when I was underperforming, but most importantly for the times when he was believing in me even more than myself.

At the personal level, I big thanks goes to my husband Traianos Plougarli, who gave me the option to pursue my PhD degree and encouraged me to continue my studies all these years. Through his priceless support and selfless attitude,

especially in the difficult times of this PhD journey, he helped me to successfully complete on more step in my academic route. In the course of my PhD I became pregnant and gave birth to a beautiful boy. I could not omit thanking him for the tranquility, strength, and courage that he brought to my life. Every day, he makes me want to become a better person and of course a better researcher. Finally, I would like to thank my family, my father Christo, my mother Dimitra, and my brother Niko for their love and continuous support in all the years of my studies. Their support was of the ultimate importance in both good and bad times, but especially in cases when frustration was my main 'PhD feeling'.

This PhD thesis is devoted to all the people that believed in me all these years and helped my bringing this project to a successful ending.

Thank you all!