ARCHITECTURAL EROSION DETERMINATION MODEL USING APPROACH-BASED METRICS AND QUALITY ATTRIBUTES WITH STRUCTURAL EQUATION MODELLING

By

AHMED OMAR SALEM BABBAD

Thesis Submitted to the School of Graduate Studies, Universiti Putra Malaysia, in Fulfilment of the Requirements for the Degree of Doctor of Philosophy

May 2024

FSKTM 2024 1

DEDICATION

To my dear family, your endless love, support, and encouragement have been my guiding light throughout this academic journey. To my parents, your wisdom and sacrifices paved the way for my success. To my loving wife, your understanding and motivation pushed me beyond my limits. To my dear children and all the wonderful people who enriched my life and inspired me to achieve more than I ever imagined. This thesis is dedicated to each one of you with sincere gratitude.

Abstract of thesis presented to the Senate of Universiti Putra Malaysia in fulfilment of the requirement for the degree of Doctor of Philosophy

ARCHITECTURAL EROSION DETERMINATION MODEL USING APPROACH-BASED METRICS AND QUALITY ATTRIBUTES WITH STRUCTURAL EQUATION MODELLING

By

AHMED OMAR SALEM BAABAD

May 2024

Chairman : Hazura binti Zulzalil, PhD
Faculty : Computer Science and Information Technology

The success or failure of software development and design heavily depends on software architecture, which plays a vital role in various aspects of the development process. It is acknowledged that having the right architecture is crucial for system design and development. However, as systems evolve, software architecture tends to degrade, leading to architectural erosion. Although several studies have explored different approaches to tackle architectural erosion, with metrics being a common solution, there is a significant gap in knowledge on the metrics used to identify architectural erosion and classify adopted approaches.

Therefore, this research aims a model that combines approach-based metrics and architectural attributes quality to address the following main aspects: 1) identification of adopted approaches and commonly used metrics practices associated with each approach to identify architectural erosion, 2) identification

of key quality attributes related to architectural erosion for each adopted approach, and 3) provision of empirical evidence on the identification of each adopted approach in the context of architectural erosion.

Initially, 92 metrics practices and 10 quality attributes relevant to architectural erosion were identified. These metrics practices were systematically assigned to adopted approaches such as architectural change, historical data revision, architectural dependency coupling, architectural bad smells, architectural cohesion, software architecture size, architectural technical debt, architectural complexity, and architecture modularization. The model was subjected to content validation by experts and was confirmed to be reliable. A subsequent reliability study involving 30 software engineering professionals further validated the constructs within the model. The model was further investigated using Structural Equation Modeling (SEM) based on data collected from a survey of 130 software engineering professionals. SEM analysis provided significant insights into the relationship between different aspects of the approaches and architectural erosion. The findings of the study revealed that most approaches significantly impact architectural erosion, with the exception of architectural complexity and technical debt, which showed weaker relationships. The findings justify the integration of approach-based metrics with architectural quality attributes, demonstrating the model's reliability by obtaining positive feedback from experts regarding its usefulness. In conclusion, this research offers a comprehensive perspective on the approaches used to identify architectural erosion, encompassing types of approaches, quality attributes, and common metrics practices. In addition, it provides valuable insights and practical guidance for researchers and practitioners in the field.

Keywords: Architectural Erosion, Metrics, Quality Attributes, Software Architecture, Structural Equation Modeling (SEM).

SDG: GOAL 8: Decent Work and Economic Growth, GOAL 9: Industry, Innovation, and Infrastructure, GOAL 12: Responsible Consumption and Production

Abstrak tesis yang dikemukakan kepada Senat Universiti Putra Malaysia sebagai memenuhi keperluan untuk ijazah Doktor Falsafah

MODEL PENENTUAN EROSI SENIBINA MENGGUNAKAN METRIK BERASASKAN PENDEKATAN DAN ATRIBUT KUALITI DENGAN PEMODELAN PERSAMAAN STRUKTUR

Oleh

AHMED OMAR SALEM BAABAD

Mei 2024

Pengerusi : Hazura binti Zulzalil, PhD
Fakulti : Sains Komputer dan Teknologi Maklumat

Kejayaan atau kegagalan pembangunan dan reka bentuk perisian sangat bergantung pada seni bina perisian, yang memainkan peranan penting dalam pelbagai aspek proses pembangunan, termasuk analisis, penggunaan semula, pemahaman, evolusi, pembinaan dan pengurusan. Adalah diakui bahawa mempunyai seni bina yang betul adalah penting untuk reka bentuk dan pembangunan sistem. Walau bagaimanapun, apabila sistem berkembang, seni bina perisian cenderung merosot, membawa kepada fenomena yang dikenali sebagai hakisan seni bina. Beberapa kajian telah meneroka pendekatan yang berbeza untuk menangani hakisan seni bina, dengan metrik muncul sebagai penyelesaian yang paling banyak digunakan. Walau bagaimanapun, terdapat kekurangan penyelidikan mengenai metrik untuk mengenal pasti hakisan seni bina dan mengkategorikan pendekatan yang diterima pakai. Mengenal pasti amalan metrik yang digunakan dalam konteks ini adalah penting untuk pemahaman yang menyeluruh. Walaupun

menyedari kepentingannya, masih terdapat kekurangan pengetahuan tentang menyesuaikan pendekatan dan metrik dalam menyiasat fenomena ini.

Oleh itu, penyelidikan ini bertujuan untuk mencadangkan model yang menggabungkan metrik berasaskan pendekatan dan kualiti atribut seni bina untuk menangani aspek utama berikut: 1) pengenalpastian pendekatan yang diterima pakai dan amalan metrik yang biasa digunakan yang dikaitkan dengan setiap pendekatan untuk mengesan hakisan seni bina, 2) pengenalpastian atribut kualiti utama yang berkaitan dengan hakisan seni bina bagi setiap pendekatan yang diterima pakai, dan 3) penyediaan bukti empirikal mengenai pengenalpastian setiap pendekatan yang diterima pakai dalam konteks hakisan seni bina.

Model yang dicadangkan pada mulanya melibatkan 92 amalan metrik dan 10 atribut kualiti khusus untuk mengenal pasti hakisan seni bina. Setiap amalan metrik yang diperiksa secara sistematik diberikan kepada salah satu pendekatan yang diterima pakai, termasuk perubahan seni bina, semakan data sejarah, gandingan kebergantungan seni bina, bau busuk seni bina, perpaduan seni bina, saiz seni bina perisian, hutang teknikal seni bina, kerumitan seni bina dan modularisasi seni bina. Model ini tertakluk kepada proses pengesahan kandungan selepas memasukkan cadangan daripada pakar kandungan dan didapati boleh dipercayai. Selepas itu, kajian kebolehpercayaan yang melibatkan 30 profesional kejuruteraan perisian dalam bidang kejuruteraan perisian yang secara khusus menangani hakisan seni bina dan langkah-langkahnya mengesahkan kebolehpercayaan semua

binaan dalam model. Model ini disiasat selanjutnya menggunakan Pemodelan Persamaan Struktur (SEM) berdasarkan data yang dikumpul daripada tinjauan terhadap 130 profesional kejuruteraan perisian. Analisis SEM memberikan pandangan yang ketara ke dalam hubungan antara pelbagai aspek pendekatan dan hakisan seni bina. Penemuan kajian mendedahkan bahawa kebanyakan pendekatan memberi kesan ketara kepada hakisan seni bina, kecuali kerumitan seni bina dan hutang teknikal, yang menunjukkan hubungan yang lebih lemah. Penemuan ini membenarkan penyepaduan metrik berasaskan pendekatan dengan atribut kualiti seni bina, menunjukkan kebolehpercayaan model dengan mendapatkan maklum balas positif daripada pakar mengenai kegunaannya. Kesimpulannya, penyelidikan ini menawarkan perspektif menyeluruh tentang pendekatan yang digunakan untuk mengenal pasti hakisan seni bina, merangkumi jenis pendekatan, atribut kualiti dan amalan metrik biasa. Di samping itu, ia memberikan pandangan yang berharga dan bimbingan praktikal untuk penyelidik dan pengamal dalam bidang tersebut.

Kata Kunci: Atribut Kualiti, Hakisan Seni Bina, Metrik, Pemodelan Persamaan Struktur (SEM), Seni Bina Perisian.

SDG: MATLAMAT 8: Pekerjaan yang Layak dan Pertumbuhan Ekonomi, MATLAMAT 9: Industri, Inovasi, dan Infrastruktur, MATLAMAT 12: Penggunaan dan Pengeluaran yang Bertanggungjawab

ACKNOWLEDGEMENTS

I wish to express my deep gratitude to the supervisory committee, headed by Assoc. Prof. Dr. Hazura Zulzalil, along with Assoc. Prof. Dr. Sa'adah Hassan, and Assoc. Prof. Dr. Salmi Baharom. Your invaluable time, unwavering support, and wise counsel have been instrumental in my journey. I am particularly indebted to Assoc. Prof. Dr. Hazura Zulzalil, who, as the committee's chairman, provided essential guidance, profound insights, expert knowledge, and constant encouragement, leading to the successful completion of this thesis.

I am profoundly grateful to my parents, wife, friends, and all members of my family for their unwavering and unconditional support throughout my Ph.D. research journey. Their encouragement has been a constant source of motivation. Additionally, I extend my sincere acknowledgment and deep gratitude to Hadramout establishment and Hadramout University for their generous financial and moral support, which played a crucial role in enabling me to successfully complete my studies.

This thesis was submitted to the Senate of Universiti Putra Malaysia and has been accepted as fulfilment of the requirement for the degree of Doctor of Philosophy. The members of the Supervisory Committee were as follows:

Hazura binti Zulzalil, PhD
Associate Professor
Faculty of Computer Science and Information Technology
Universiti Putra Malaysia
(Chairman)

Sa'adah binti Hassan, PhD
Associate Professor
Faculty of Computer Science and Information Technology
Universiti Putra Malaysia
(Member)

Salmi binti Baharom, PhD
Associate Professor
Faculty of Computer Science and Information Technology
Universiti Putra Malaysia
(Member)

ZALILAH MOHD SHARIFF, PhD
Professor and Dean
School of Graduate Studies
Universiti Putra Malaysia

Date: 12 December 2024

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF APPENDICES

LIST OF ABBREVIATIONS

ABS	Architectural bad smells
ACH	Architectural change
ACO	Architectural cohesion
ACP	Architectural complexity
AD	Architectural decay / Architectural degradation
AEr	Architecture Erosion / Architectural Erosion
AIn	Architectural inconsistencies
AM	Architecture modularization
AQAs	Architecture quality attributesبهق
ASA	Architectural significant attributes
ATD	Architectural technical debt
AVE	Average Variance Extracted
C.R	Critical value
CA	Conceptual architecture
CCCU	Composite Construct Consistency and Unidimensionality
CMB	Common Method Bias
CR	Composite Reliability
CVR	Content Validity Ratio
DA	Descriptive architecture
DL	Decoupling Level
DSL	Domain-Specific Language
FL	Factor Loading
FLOSS	Free/Libre and Open-Source Software
FSM	Functional Sizing Measurement

GoF	Goodness of Fit
HDR	Historical data revision
HTMT	Heterotrait-Monotrait Ratio of Correlations
ISA	Implemented software architecture
ISO	International Organisation for Standardization
ISs	Information Systems
MMA	Measurement Model Assessment
OSS	Open-source software
PA	Prescriptive architecture
PCI	Project Call Instability
PDI	Project Design Instability
PLS-SEM	Partial Least Squares Structural Equation Modelling
PSA	Planned software architecture
SA	Software Architecture
SACC	Software architecture conformance checking
SAD	Software architecture degradation
SAZ	Software architecture size
SD	Standard Deviation
SDLC	Software development lifecycle
SE	Software Engineering
SEM	Structural Equation Modelling
SEPs	Software Engineering Professionals
SLR	systematic literature review
SMA	Structural Model Assessment
SMs	Software Metrics
SMS	Systematic mapping study
SPSs	Selected primary studies

SQ	Software Quality
VIF	Variance Inflation Factor

CHAPTER 1

INTRODUCTION

1.1 Background

Since late 1989, software architecture (SA) has appeared as the initial conception of the large-widely structures of software systems. SA is getting increasing attention within the software engineering domain over the past decade (Bosch & Molin, 1999). In both academia and industry, SA has become a widely accepted concept (Bass, Weber, & Zhu, 2015; Clements et al., 2003). As a result, the essence of software system design and development is the key notion for SA (Taylor, Medvidovic, & Dashofy, 2009). It plays a pivotal role in many aspects of software development, including analysis, reuse, comprehension, evolution, construction, and management. SA interplays and overlaps with the study of domain-specific design, component-based reuse, software families, specific classes of components, program analysis, and software design (Shaw & Clements, 2006). Accordingly, the influence of SA change on software-intensive systems has a significant impact on their long-term viability, whether positively or negatively. As part of the software evolution life cycle, SA frequently provides components for the fulfilment of non-functional and functional requirements (Bachmann, Bass, Klein, & Shelton, 2005; Bass et al., 2015).

In essence, the SA can be defined as the structure of the system, which includes software components, observable attributes of components, and their interactions (Bass, Clements, & Kazman, 2021). As can be seen from this definition, the internal characteristics of each entity's system structure have a substantial impact on SA. It is concerned with the high-level structure and properties of the system (Perry & Wolf, 1992; Shaw & Garlan, 1996). It is important to highlight that SA refers to the highest level of abstraction and fundamental reasoning applied in software development, regardless of whether it is open-source software (OSS) or closed-source (proprietary source).

Moreover, the architecture of a system refers to the collection of major design decisions made throughout the system's development and any subsequent evolution. As a subject, architecture is the proper major emphasis of software engineering, because the production of high-quality, successful products is dependent on the decisions made at the beginning of the development process. The decisions and principles that will guide the development of the system are considered to be part of the SA (Stringfellow, Amory, Potnuri, Andrews, & Georg, 2006). The goal of concentrating on SA is to identify and analyse crucial early design decisions (Klein et al., 1999; Pfeifer et al., 2022). As a consequence, system designers and developers have realised that getting the architecture right is a vital success factor in the process of developing a system (Garlan, 2000).

Interestingly, as systems evolve, their SA usually erodes over time (Lehman, 1979, 1996). Inevitably, this problem does not happen overnight, but it is long-standing in software engineering (Herold, Knieke, Schindler, & Rausch, 2020). The eroding architecture drives the system to greater complexity, difficulty, and frequency of change than previously (Stringfellow et al., 2006). As a result, the system's lifetime is shortened, or it has a significant impact on software quality or the maintenance and development systems life cycle (Garcia, Ivkovic, & Medvidovic, 2013), necessitating an entire redesign of the system's architecture from scratch (Lenhard, Blom, & Herold, 2019; Zude & Jun, 2011). This phenomenon is referred to as Architecture Erosion (AEr) (Bosch, 2004; Hochstein & Lindvall, 2005; Medvidovic, Egyed, & Grünbacher, 2003; Parnas, 1994), Software Architecture Degradation (SAD) (Amalfitano, Luca, & Fasolino, 2023; Lenhard et al., 2019; Perry & Wolf, 1992; Taylor et al., 2009), Architectural Decay (AD) (Jin et al., 2023; Riaz, Sulayman, & Naqvi, 2009). In accordance with the studies conducted by researchers in this field, Architecture Erosion (AEr) or Architectural Erosion (AEr) emerges as the most used term, and this will be the term adopted in this research.

Architecture Erosion (AEr) can be defined as the persistent divergence between prescriptive and descriptive software architecture as intended and implemented (Perry & Wolf, 1992; Taylor et al., 2009). It takes place when the implemented software architecture (ISA), which represents the actual functions of the system, deviates from the planned software architecture (PSA), which represents the system's original design. The occurrence of AEr represents by predefined mapping and accomplishing of systems based on inconsistent properly regarding the architecture design and the source code. This phenomenon is caused by factors that may lead to AEr such as undocumented, unforeseen, unplanned, random, scattered, and confused architectural design decisions. Furthermore, architectural change (ACH) of a system over time (Jin et al., 2023; Lindvall, Tesoriero, & Costa, 2002), disregard for fundamental architectural rules of a system due to the modification, developer mistakes, and bad practices (Bandara & Perera, 2018) have a strong impact for continuing AEr, which may shorten the system lifetime or require reengineering from scratch (Bandara & Perera, 2018; Zude & Jun, 2011).

Numerous experimental studies have been conducted to address the issue of AEr, aiming to identify, avoid, minimize, or repair it. To achieve this, various tools, models, and measures have been suggested to detect deviations from the intended architecture and erosion in its early stages. These efforts have resulted in a variety of proposed solutions to combat AEr. These solutions encompass different strategies, such as models, measurements, approaches, algorithms, tools, techniques, and methods, which may be combined to form a comprehensive solution. Some of the most important solutions include a metrics-based detection strategy, Prioritisation of architectural anomalies, investigation and remediation of architectural rule violations, refactoring, architectural recovery, and other proposed solutions.

1.2 Research Motivation

Addressing AEr in software development is essential for the success of software projects, as the multifaceted nature of this issue has garnered significant attention in discussions and research, highlighting its crucial importance. This phenomenon can result in instability in structural issues (Ruiyin Li, Peng Liang, Mohamed Soliman, & Avgeriou, 2021) due to violation of design rules about encapsulation, accumulation of cyclic dependencies, increased coupling, and an inability to meet evolving business requirements. Moreover, maintenance, evolution, quality issues (Ruiyin Li et al., 2021), and increased development costs (Andrews & Sheppard, 2020) can ultimately threaten software projects' success.

Understanding and mitigating AEr is essential for maintaining the long-term viability and sustainability of software systems. It enables organisations to preserve the integrity of their software architectures, enhance system maintainability, and facilitate future evolution and adaptation to changing business needs. Therefore, proactive measures to determine and mitigate AEr are essential for ensuring the resilience and longevity of software systems (Whiting & Andrews, 2020).

The prevalence of AEr in software development has prompted the exploration of various solutions, with the metrics approach emerging as the most used and effective technique (Misra, Adewumi, Fernandez-Sanz, & Damasevicius, 2018). This method, as highlighted by Misra (2011), emphasise the significance of employing software metrics throughout the software development lifecycle (SDLC) to enhance and monitor diverse software engineering practices.

The underlying logic is that "you cannot control what you cannot measure" (DeMarco, 1986). Typically, several systems lack an explicit and precise description of Prescriptive Architecture (PA) in practice, contributing to AEr. It was found that a mere 5% of open-source projects maintain software architectural documentation (Ding, Liang, Tang, & Vliet, 2015), emphasising the challenge of addressing AEr. This indicates the necessity to construct architectural specifications from scratch, a daunting task for systems that have hundreds of thousands or even millions of lines of code. Therefore, it is obviously difficult to draught up a detailed architectural specification in the face of AEr.

When architectural documentation is missing, the code is typically the major source of information concerning the possibility of architectural violations and erosion (Lenhard et al., 2019; Lenhard, Hassan, Blom, & Herold, 2017). Hence, having a solid understanding of the fundamentals behind the metrics strategy is an effective method for determining whether there are issues with the underlying architecture or its source code, particularly in the context of AEr. Thus, the significance of leveraging metrics as a strategy to address AEr is emphasised, guided by the principle that "you cannot control what you cannot measure." Evaluating software systems requires measuring both architectural quality attributes and software metrics (SMs) at the architectural level (Alenezi, 2016). Aligning these attributes with metrics is essential for measuring AEr in software architecture.

The key motivation behind this research is to address the challenges posed by AEr in software development. With growing attention on AEr, the study aims to provide a clear understanding of metrics practices, particularly in analysing source code in the absence of explicit architectural documentation, thus clarifying its role in mitigating issues related to AEr.

1.3 Problem Statement

The practice of utilising metrics is crucial in determining and mitigating the impact of AEr (Le Duc, Carlos, Rafael, & Nenad, 2016; Lenhard et al., 2019; Ruiyin Li et al., 2021). In addition, when evaluating software systems, both quality attributes and (SMs) are essential. Quantifying these attributes at the architectural level provides valuable insights into the software's design, as quality is typically assessed at this higher level rather than at the code level (Alenezi, 2016). Consequently, these attributes are referred to as architectural quality attributes (AQAs). The alignment of quality attributes with relevant metrics is crucial for accurately measuring AEr in software architecture.

The challenge of determining approaches-based metrics to measure AEr is a significant concern for software engineers and researchers. Addressing these challenges is crucial for providing valuable insights and measures to control AEr, forming a basis for further exploration of this critical issue (Lakshitha &

Balasubramaniam, 2012; Ruiyin Li et al., 2021; Whiting & Andrews, 2020). Empirical evidence highlights the adoption of approaches-based metrics to illustrate architectural enhancements (Lindvall, Tvedt, & Costa, 2003), who widely applied these metrics to propose solutions for determining AEr, measuring architectural significant attributes (ASA), and identifying indicators of architectural deviations aligned with architectural guidelines.

Several models have emerged that focus on establishing metrics for evaluation purposes. For instance, Dayanandan and Vivekanandan (2016) developed a framework that emphasizes the importance of quantitative metrics in assessing architectural decisions, while Pan, Liu, Li, Wang, and Li (2023) introduced a model that integrates machine learning techniques to enhance the measurement of AEr, providing a more nuanced understanding of architectural changes. In addition, Lindvall et al. (2002) also contributed a model that assesses the impact of architectural modifications on system quality attributes through scenario-based analysis.

Moreover, Zi Li, Li, and Kang (2016) proposed a model for assessing system resilience using minimal path analysis and various resilience metrics, which aids in identifying critical components and potential failure points within system architecture. These models collectively underscore the need for robust approaches-based metrics to evaluate AEr comprehensively.

However, despite the growing body of empirical evidence, there remains a gap in knowledge regarding the comprehensive elements necessary to evaluate the phenomenon of AEr. The adoption of approaches-based metrics is gaining increased attention, particularly in the context of architectural change approaches, as highlighted by Jin et al. (2023) and Tonu, Ashkan, and Tahvildari (2006). These studies illustrate the necessity for a more integrated model that synthesizes existing approaches and metrics, thereby justifying the method used in the current study. By addressing these gaps, the proposed research aims to establish a more holistic understanding of AEr and the metrics that can measure it.

The gap in the use of metrics often arises from their application based on researchers' perspectives without adequate specificity and clarity (Alsulami, 2021; Fenton & Pfleeger, 2015; Mäntylä, 2008; Nuñez-Varela, Pérez-Gonzalez, Martínez-Perez, & Soubervielle-Montalvo, 2017). This variation in how metrics are presented leads to confusion among practitioners, misunderstandings, and misapplications (Fenton & Pfleeger, 2015). As a result, measurements become ineffective, hindering improvement efforts (Basili & Rombach, 2002). These issues complicate the understanding of fundamental concepts and contribute to increasing the challenges in achieving widespread acceptance of these metrics. Therefore, metrics need to be clearly defined and context-specific, which is crucial for reproducibility, understanding (Barbara & Charters, 2007), and supporting decision-making to enhance the development process (Buse & Zimmermann, 2012).

In view of the importance of approaches-based metrics for determining AEr, it is beneficial for researchers and practitioners to understand the metrics practices pertaining to every approach in the context of AEr, and the essential quality attributes from the AEr perspective. Hence, developing a model for measuring the suitability of the chosen metrics, accompanied by a well-defined classification aimed at clarifying vague metric definitions and their associated approaches and categories, that provides this set of information, is in demand to be researched. This model aims to identify and classify metrics associated with AEr, propose a correlation of metrics with architectural quality attributes to measure AEr, and subsequently evaluate the validity and reliability of the model.

1.4 Research Objectives

The research objectives are as follow:

i. To identify and classify the metrics adopted from approaches related to AEr.
ii. To develop a model for determining metrics and architectural quality attributes to determine AEr.
iii. To assess the validity and reliability of the model.

1.5 Research Scope

This study investigates the metrics approaches that could determine AEr. Therefore, the adopted metrics approaches based on determination of AEr perspective is the purpose of this research. Consequently, the research scopes are:

i. This research is limited to measures, metrics approaches, and AEr attributes that are outcomes of systematic mapping study and experts' opinions. However, this study does not claim exclusivity to these metrics' approaches, measures, or AEr attributes.

ii. Even while this study will provide an empirical report of the determination of each metrics approach on AEr as defined by a set of quality attributes, it will not account for the identification of each metrics approach on each attribute of AEr.

iii. This study primarily focuses on AEr context based on the researcher's strategic use of the chance to find potentially helpful panellists. This could include panellists who work in academic researchers and industry practitioners.

1.6 Research Contribution

This study's contribution is presented both in theory and in practice. Theoretically, this research contributes to the body of knowledge for software engineering and information systems (ISs) in many different ways; however, the most important contribution is the construction and evaluation of the approaches, adopted metrics, and quality attributes of AEr model to determine erosion. This is especially important given that no studies have been conducted for the potential determination of various types of metrics and their approaches to identify AEr. In addition, the model provides information that could be used to measure different various types of metrics and their approaches to determine AEr. The information lists that are supplied by this model as well as the significance of each one is described in more depth below.

i. A comprehensive classification for the aim of determining AEr, having specific and consistent metrics that can be used to study and evaluate a wide variety of relevant approaches for AEr context.

ii. Additionally, such an approach can provide a better understanding of the complexity of the AEr problem and its underlying causes. Furthermore, it can contribute to the development of better approaches and solutions to manage and control the AEr process. Finally, it can help to identify the most effective metrics strategies to mitigate the risks associated with AEr.

iii. This model has been subjected to both an empirical evaluation and an evaluation from the perspective of Software Engineering Professionals (SEPs), yielding a plethora of information (such as erosion classifications, approach metrics, and quality attributes) that can provide useful insights into the nature of AEr and that it is able to help identification areas requiring improvement.

The development of questionnaires to measure AEr in light of the established approaches metrics through the use of research instruments is a practical significant contribution to the field. Valid and reliable studies provide important guidelines and references for future researchers whose goals are similar to this study intentions.

1.7 Organisation of the Thesis

This thesis is structured to critically review relevant information on approach metrics and AEr, providing details on the research methodology and key findings. It is organised into six chapters. This chapter provides an overview of the background of the research area, highlights the motivation and problem statement, and discusses the objectives, scope, and key contributions of the study.

Chapter 2 provides an in-depth literature review of the research problem. It covers various topics related to the research problem such as approaches metrics, quality attributes, and AEr. It also presents a comprehensive overview of the research findings.

Chapter 3 addresses the research technique and supports the design of the research methodology employed to perform this study. In addition, procedures for the research process, design, instrument development, pilot study, population, sample, and data collecting, and data analysis are described. Detailed descriptions of how each aim was attained are provided in each corresponding chapter.

Chapter 4 presents the conceptualisation of the model and the development of hypotheses. It includes a detailed explanation of the research model, its justification, and the formulation of hypotheses.

Chapter 5 presents the results of the research. It provides a detailed description of the results of the research and a discussion of the implications of the results.

Chapter 6 provides a conclusion to the research and discussing recommendations for future research.

REFERENCES

Abdeen, H., Ducasse, S., Sahraoui, H., & Alloui, I. (2009, 13-16 Oct. 2009). *Automatic Package Coupling and Cycle Minimization.* Paper presented at the 2009 16th Working Conference on Reverse Engineering, Lille, France.

Adams, L., S. Abdelfattah, A., Hossain Chy, M. S., Perry, S., Harris, P., Cerny, T., . . . Taibi, D. (2024). *Evolution and Anti-patterns Visualized: MicroProspect in Microservice Architecture*, Cham.

Alenezi, M. (2016). Software Architecture Quality Measurement Stability and Understandability. *International Journal of Advanced Computer Science and Applications(IJACSA), 7*(7). doi:http://dx.doi.org/10.14569/IJACSA.2016.070775

Alsulami, M. (2021). A Systematic Literature Review on Software Metrics. *International Transaction Journal of Engineering, Management, & Applied Sciences & Technologie, 12*(12), 1-13. doi:http://doi.org/10.14456/ITJEMAST.2021.238

Altınışık, M., Ersoy, E., & Sözer, H. (2017). *Evaluating software architecture erosion for PL/SQL programs*. Paper presented at the Proceedings of the 11th European Conference on Software Architecture: Companion Proceedings, Canterbury, United Kingdom. https://doi.org/10.1145/3129790.3129811

Amalfitano, D., Luca, M. D., & Fasolino, A. R. (2023, 13-17 March 2023). *Documenting Software Architecture Design in Compliance with the ISO 26262: a Practical Experience in Industry.* Paper presented at the 2023 IEEE 20th International Conference on Software Architecture Companion (ICSA-C), L'Aquila, Italy.

Ambros, M. D., Lanza, M., & Robbes, R. (2009, 13-16 Oct. 2009). *On the Relationship Between Change Coupling and Software Defects.* Paper presented at the 2009 16th Working Conference on Reverse Engineering, Lille, France.

Andrew L. Comrey, H. B. L. (1992). *A First Course in Factor Analysis (2nd Edition)*. New York: Lawrence Erlbaum Associates.

Andrews, S., & Sheppard, M. (2020). Software Architecture Erosion: Impacts, Causes, and Management. *International Journal of Computer Science and Security (IJCSS), 14*(2), 82 - 93

Anthony, E., Berntsson, A., Santilli, T., & Wohlrab, R. (2024, 4-8 June 2024). *We are Drifting Apart: Architectural Drift from the Developers' Perspective.* Paper presented at the 2024 IEEE 21st International Conference on Software Architecture (ICSA).

Arcelli Fontana, F., Braione, P., & Zanoni, M. (2012). Automatic detection of bad smells in code: An experimental assessment. *Journal of Object Technology, 11*. doi:10.5381/jot.2012.11.2.a5

Arcelli Fontana, F., Lenarduzzi, V., Roveda, R., & Taibi, D. (2019). Are architectural smells independent from code smells? An empirical study. *Journal of Systems and Software, 154*, 139-156. doi:https://doi.org/10.1016/j.jss.2019.04.066

Arcoverde, R., Guimarães, E., Macía, I., Garcia, A., & Cai, Y. (2013, 1-4 Oct. 2013). *Prioritization of Code Anomalies Based on Architecture Sensitiveness.* Paper presented at the 2013 27th Brazilian Symposium on Software Engineering, Brasilia, Brazil.

Aversano, L., Guardabascio, D., & Tortorella, M. (2019). An Empirical Study on the Architecture Instability of Software Projects. *International Journal of Software Engineering and Knowledge Engineering, 29*(04), 515-545. doi:10.1142/s0218194019500220

Ayre, C., & Scally, A. J. (2013). Critical Values for Lawshe's Content Validity Ratio: Revisiting the Original Methods of Calculation. *Measurement and Evaluation in Counseling and Development, 47*(1), 79-86. doi:10.1177/0748175613513808

Ayyaz, S., Rehman, S., & Qamar, U. (2015). A Four Method Framework for Fighting Software Architecture Erosion. *International Journal of Computer, Information, Systems and Control Engineering, 9*.

Bachmann, F., Bass, L., Klein, M., & Shelton, C. P. (2005). Designing software architectures to achieve quality attribute requirements. *IEE Proceedings - Software, 152*(4), 153-165. doi:10.1049/ip-sen:20045037

Badampudi, D., Wohlin, C., & Petersen, K. (2015). *Experiences from using snowballing and database searches in systematic literature studies*. Paper presented at the Proceedings of the 19th International Conference on Evaluation and Assessment in Software Engineering, Nanjing, China. https://doi.org/10.1145/2745802.2745818

Bandara, V., & Perera, I. (2018, 26-29 Sept. 2018). *Identifying Software Architecture Erosion Through Code Comments.* Paper presented at the 2018 18th International Conference on Advances in ICT for Emerging Regions (ICTer), Colombo, Sri Lanka.

Barbara, K. (2012). *Systematic review in software engineering: where we are and where we should be going*. Paper presented at the Proceedings of the 2nd international workshop on Evidential assessment of software technologies, Lund, Sweden. https://doi.org/10.1145/2372233.2372235

Barbara, K., Budgen, D., & Brereton, O. P. (2011). Using mapping studies as the basis for further research – A participant-observer case study. *Information and Software Technology, 53*(6), 638-651. doi:https://doi.org/10.1016/j.infsof.2010.12.011

Barbara, K., & Charters, S. (2007). *Guidelines for performing systematic literature reviews in software engineering* Retrieved from

Barros, M. d. O., Farzat, F. d. A., & Travassos, G. H. (2015). Learning from optimization: A case study with Apache Ant. *Information and Software Technology, 57*, 684-704. doi:https://doi.org/10.1016/j.infsof.2014.07.015

Basili, V. R., & Rombach, H. D. (2002). The TAME project: towards improvement-oriented software environments. *IEEE Transactions on Software Engineering, 14*(6), 758-773. doi:10.1109/32.6156

Bass, L., Clements, P., & Kazman, R. (2021). *Software Architecture in Practice (4th ed.)*: Addison-Wesley Professional.

Bass, L., Weber, I., & Zhu, L. (2015). *DevOps: A Software Architect's Perspective*: Addison-Wesley Professional.

Behnamghader, P., Le, D. M., Garcia, J., Link, D., Shahbazian, A., & Medvidovic, N. (2017). A large-scale study of architectural evolution in open-source software systems. *Empirical Software Engineering, 22*(3), 1146-1193. doi:10.1007/s10664-016-9466-0

Belle, A. B., Boussaidi, G. E., & Kpodjedo, S. (2016). Combining lexical and structural information to reconstruct software layers. *Information and Software Technology, 74*, 1-16. doi:https://doi.org/10.1016/j.infsof.2016.01.008

Bertran, I. M. (2011, 21-28 May 2011). *Detecting architecturally-relevant code smells in evolving software systems.* Paper presented at the 2011 33rd International Conference on Software Engineering (ICSE), Honolulu, HI, USA.

Bhattacharya, S., & Perry, D. E. (2007, 6-9 Jan. 2007). *Architecture Assessment Model for System Evolution.* Paper presented at the 2007 Working IEEE/IFIP Conference on Software Architecture (WICSA'07), Mumbai, India.

Biaggi, A., Fontana, F. A., & Roveda, R. (2018, 29-31 Aug. 2018). *An Architectural Smells Detection Tool for C and C++ Projects.* Paper presented at the 2018 44th Euromicro Conference on Software Engineering and Advanced Applications (SEAA), Prague, Czech Republic.

Bindal, D. A., & Singh, R. (2019). Reducing maintenance efforts of developers by prioritizing different code smells. *International Journal of Innovative Technology and Exploring Engineering (IJITEE), 8*(8S3), 139–144.

Bollen, K. A. (1989). *Structural equations with latent variables* (Vol. 210). United States: John Wiley & Sons.

Bosch, J. (2004). *Software Architecture: The Next Step.* Paper presented at the Software Architecture, Berlin, Heidelberg.

Bosch, J., & Molin, P. (1999). *Software architecture design: evaluation and transformation.* Paper presented at the Proceedings ECBS'99. IEEE Conference and Workshop on Engineering of Computer-Based Systems, Nashville, TN, USA.

Brace, I. (2018). *Questionnaire design: How to plan, structure and write survey material for effective market research*: Kogan Page Publishers.

Brown, N., Cai, Y., Guo, Y., Kazman, R., Kim, M., Kruchten, P., . . . Zazworka, N. (2010). *Managing technical debt in software-reliant systems*. Paper presented at the Proceedings of the FSE/SDP workshop on Future of software engineering research, Santa Fe, New Mexico, USA. https://doi.org/10.1145/1882362.1882373

Brunet, J., Bittencourt, R. A., Serey, D., & Figueiredo, J. (2012, 15-18 Oct. 2012). *On the Evolutionary Nature of Architectural Violations.* Paper presented at the 2012 19th Working Conference on Reverse Engineering, Kingston, ON, Canada.

Bryman, A., & Cramer, D. (2005). *Quantitative data analysis with SPSS 12 and 13: A guide for social scientists*: Psychology Press.

Buse, R. P. L., & Zimmermann, T. (2012, 2-9 June 2012). *Information needs for software development analytics.* Paper presented at the 2012 34th International Conference on Software Engineering (ICSE), Zurich, Switzerland.

Byrne, B. M. (2010). *Structural Equation Modeling with AMOS: Basic Concepts, Applications and Programming (second edition)* (Vol. 5). New York:: Taylor & Francis Group.

Cai, Y., Xiao, L., Kazman, R., Mo, R., & Feng, Q. (2019). Design Rule Spaces: A New Model for Representing and Analyzing Software Architecture. *IEEE Transactions on Software Engineering, 45*(7), 657-682. doi:10.1109/TSE.2018.2797899

Capiluppi, A., & Knowles, T. (2009, 2009//). *Software Engineering in Practice: Design and Architectures of FLOSS Systems.* Paper presented at the Open Source Ecosystems: Diverse Communities Interacting, Berlin, Heidelberg.

Carvalho, L. P. d. S., Novais, R., & Mendonça, M. (2018). *Investigating the Relationship between Code Smell Agglomerations and Architectural Concerns: Similarities and Dissimilarities from Distributed, Service-Oriented, and Mobile Systems*. Paper presented at the Proceedings of the VII Brazilian Symposium on Software Components, Architectures, and Reuse, Sao Carlos, Brazil. https://doi.org/10.1145/3267183.3267184

Černý, T., Chy, M. S. H., Abdelfattah, A., Soldani, J., & Bogner, J. (2024). *On Maintainability and Microservice Dependencies: How Do Changes Propagate?*

Chin, W. W. (2010). Bootstrap Cross-Validation Indices for PLS Path Model Assessment. In V. Esposito Vinzi, W. W. Chin, J. Henseler, & H. Wang (Eds.), *Handbook of Partial Least Squares: Concepts, Methods and Applications* (pp. 83-97). Berlin, Heidelberg: Springer Berlin Heidelberg.

Churchill, G. A., & Iacobucci, D. (2006). *Marketing research: methodological foundations* (Vol. 199): Dryden Press New York.

Clements, P., Bachmann, F., Bass, L., Garlan, D., Ivers, J., Little, R., . . . Stafford, J. (2003). *Documenting software architectures: views and beyond*: Addison-Wesley Professional.

Cohen, J. (2013). *Statistical power analysis for the behavioral sciences*. Academic Press: Cambridge, MA, USA.

Cooper, D. R., Schindler, P. S., & Sun, J. (2006). *Business research methods* (Vol. 9): Mcgraw-hill New York.

Das, D., Islam, R., Kim, S., Cerny, T., Frajtak, K., Bures, M., & Tisnovsky, P. (2023). *Analyzing Technical Debt by Mapping Production Logs with Source Code*, Cham.

Dayanandan, U., & Vivekanandan, K. (2016). *An Empirical Evaluation model for Software Architecture Maintainability for Object oriented Design*. Paper presented at the Proceedings of the International Conference on Informatics and Analytics, Pondicherry, India. https://doi.org/10.1145/2980258.2980459

DeMarco, T. (1986). *Controlling Software Projects: Management, Measurement, and Estimates*: Prentice Hall PTR.

Derbel, I., Jilani, L. L., & Mili, A. (2015, 29-30 April 2015). *Computing attributes of software architectures a static method and its validation.* Paper presented at the 2015 International Conference on Evaluation of Novel Approaches to Software Engineering (ENASE), Barcelona, Spain.

Díaz-Pace, J. A., Tommasel, A., & Godoy, D. (2018, 23-24 Sept. 2018). *Towards Anticipation of Architectural Smells Using Link Prediction Techniques.* Paper presented at the 2018 IEEE 18th International Working Conference on Source Code Analysis and Manipulation (SCAM), Madrid, Spain.

Dieste, O., & Padua, A. G. (2007, 20-21 Sept. 2007). *Developing Search Strategies for Detecting Relevant Experiments for Systematic Reviews.* Paper presented at the First International Symposium on Empirical Software Engineering and Measurement (ESEM 2007), Madrid, Spain.

Dillon, W. R., Kumar, A., & Mulani, N. (1987). Offending estimates in covariance structure analysis: Comments on the causes of and solutions to Heywood cases. *Psychological Bulletin, 101*, 126-135. doi:10.1037/0033-2909.101.1.126

Ding, W., Liang, P., Tang, A., & Vliet, H. v. (2015). Understanding the Causes of Architecture Changes Using OSS Mailing Lists. *International Journal of Software Engineering and Knowledge Engineering, 25*(09n10), 1633-1651. doi:10.1142/s0218194015400367

Duc, L., Behnamghader, P., Garcia, J., Link, D., Shahbazian, A., & Medvidovic, N. (2015). *An Empirical Study of Architectural Change in Open-Source Software Systems.* Paper presented at the 2015 IEEE/ACM 12th Working Conference on Mining Software Repositories, Florence, Italy.

Duc, L., Carlos, C., Rafael, C., & Nenad, M. (2016, 5-8 April 2016). *Relating Architectural Decay and Sustainability of Software Systems.* Paper presented at the 2016 13th Working IEEE/IFIP Conference on Software Architecture (WICSA), Venice, Italy.

Duc, L., Link, D., Shahbazian, A., & Medvidovic, N. (2018). *An Empirical Study of Architectural Decay in Open-Source Software.* Paper presented at the 2018 IEEE International Conference on Software Architecture (ICSA), Seattle, WA, USA.

Eposhi, A., Oizumi, W., Garcia, A., Sousa, L., Oliveira, R., & Oliveira, A. (2019, 25-26 May 2019). *Removal of Design Problems through Refactorings: Are We Looking at the Right Symptoms?* Paper presented at the 2019 IEEE/ACM 27th International Conference on Program Comprehension (ICPC), Montreal, QC, Canada.

Ernst, N. A., Bellomo, S., Ozkaya, I., Nord, R. L., & Gorton, I. (2015). *Measure it? Manage it? Ignore it? software practitioners and technical debt*. Paper presented at the Proceedings of the 2015 10th Joint Meeting on Foundations of Software Engineering, Bergamo, Italy. https://doi.org/10.1145/2786805.2786848

Falk, R. F., & Miller, N. B. (1992). *A primer for soft modeling*: University of Akron Press.

Fenton, N. E., & Pfleeger, S. L. (2015). *Software Metrics: A Rigorous and Practical Approach*: PWS Publishing Co.

Ferreira, M., Barbosa, E., Macia, I., Arcoverde, R., & Garcia, A. (2014). *Detecting architecturally-relevant code anomalies: a case study of effectiveness and effort*. Paper presented at the Proceedings of the 29th Annual ACM Symposium on Applied Computing, Gyeongju, Republic of Korea. https://doi.org/10.1145/2554850.2555036

Filho, J. L. M., Rocha, L., Andrade, R., & Britto, R. (2017, 2017//). *Preventing Erosion in Exception Handling Design Using Static-Architecture Conformance Checking.* Paper presented at the Software Architecture, Cham.

Fontana, F. A., Avgeriou, P., Pigazzini, I., & Roveda, R. (2019, 28-30 Aug. 2019). *A Study on Architectural Smells Prediction.* Paper presented at the 2019 45th Euromicro Conference on Software Engineering and Advanced Applications (SEAA), Kallithea, Greece.

Fontana, F. A., Ferme, V., & Zanoni, M. (2015, 16-16 May 2015). *Towards Assessing Software Architecture Quality by Exploiting Code Smell Relations.* Paper presented at the 2015 IEEE/ACM 2nd International Workshop on Software Architecture and Metrics, Florence, Italy.

Fontana, F. A., & Pigazzini, I. (2021, 3-3 June 2021). *Evaluating the Architectural Debt of IoT Projects.* Paper presented at the 2021 IEEE/ACM 3rd International Workshop on Software Engineering Research and Practices for the IoT (SERP4IoT), Madrid, Spain.

Fontana, F. A., Pigazzini, I., Raibulet, C., Basciano, S., & Roveda, R. (2019). *PageRank and criticality of architectural smells*. Paper presented at the Proceedings of the 13th European Conference on Software Architecture - Volume 2, Paris, France. https://doi.org/10.1145/3344948.3344982

Fontana, F. A., Pigazzini, I., Roveda, R., Tamburri, D. A., Zanoni, M., & Nitto, E. D. (2017, 5-7 April 2017). *Arcan: A Tool for Architectural Smells Detection.* Paper presented at the 2017 IEEE International Conference on Software Architecture Workshops (ICSAW), Gothenburg, Sweden.

Fontana, F. A., Pigazzini, I., Roveda, R., & Zanoni, M. (2016, 2-7 Oct. 2016). *Automatic Detection of Instability Architectural Smells.* Paper presented at the 2016 IEEE International Conference on Software Maintenance and Evolution (ICSME), Raleigh, NC, USA.

Fontana, F. A., Roveda, R., Vittori, S., Metelli, A., Saldarini, S., & Mazzei, F. (2016). *On evaluating the impact of the refactoring of architectural problems on software quality*. Paper presented at the Proceedings of the Scientific Workshop Proceedings of XP2016, Edinburgh, Scotland, UK. https://doi.org/10.1145/2962695.2962716

Fontana, F. A., Roveda, R., Zanoni, M., Raibulet, C., & Capilla, R. (2016, 5-8 April 2016). *An Experience Report on Detecting and Repairing Software Architecture Erosion.* Paper presented at the 2016 13th Working IEEE/IFIP Conference on Software Architecture (WICSA).

Francesca, F., Roveda, R., Zanoni, M., Raibulet, C., & Capilla, R. (2016, 5-8 April 2016). *An Experience Report on Detecting and Repairing Software Architecture Erosion.* Paper presented at the 2016 13th Working IEEE/IFIP Conference on Software Architecture (WICSA), Venice, Italy.

Garcia, J., Ivkovic, I., & Medvidovic, N. (2013, 11-15 Nov. 2013). *A comparative analysis of software architecture recovery techniques.* Paper presented at the 2013 28th IEEE/ACM International Conference on Automated Software Engineering (ASE), Silicon Valley, CA, USA.

Garcia, J., Kouroshfar, E., Ghorbani, N., & Malek, S. (2021). Forecasting Architectural Decay from Evolutionary History. *IEEE Transactions on Software Engineering*, 1-1. doi:10.1109/TSE.2021.3060068

Garcia, J., Krka, I., Mattmann, C., & Medvidovic, N. (2013, 18-26 May 2013). *Obtaining ground-truth software architectures.* Paper presented at the 2013 35th International Conference on Software Engineering (ICSE), San Francisco, CA, USA.

Garlan, D. (2000). Software Architecture: A Roadmap. *Proc. of the 22nd International Conference on Software Engineering, Future of Software Engineering Track*. doi:10.1145/336512.336537

Garlan, D., Allen, R., & Ockerblo, J. (1995). Architectural mismatch: why reuse is so hard. *IEEE Software, 12*(6), 17-26. doi:10.1109/52.469757

Geisser, S. (1974). A predictive approach to the random effect model. *Biometrika, 61*(1), 101-107.

Ghorbani, N., Garcia, J., & Malek, S. (2019, 25-31 May 2019). *Detection and Repair of Architectural Inconsistencies in Java.* Paper presented at the 2019 IEEE/ACM 41st International Conference on Software Engineering (ICSE), Montreal, QC, Canada.

Goldstein, M., & Segall, I. (2015, 16-24 May 2015). *Automatic and Continuous Software Architecture Validation.* Paper presented at the 2015 IEEE/ACM 37th IEEE International Conference on Software Engineering, Florence, Italy.

Gorsuch, R. (1983). *Factor analysis (2nd ed.)*. Hillsdale, NJ: Lawrence Erlbaum Associates.

Greifenberg, T., Müller, K., & Rumpe, B. (2015). *Architectural Consistency Checking in Plugin-Based Software Systems*. Paper presented at the European Conference on Software Architecture Workshops (ECSAW'15).

Guerra-García, C., Perez-Gonzalez, H. G., Ramírez-Torres, M., Juárez-Ramírez, R., & González, H. (2020, 4-6 Nov. 2020). *MOSCAF – Specifying Data Quality Requirements according Web Functionalities.* Paper presented at the 2020 8th International Conference in Software Engineering Research and Innovation (CONISOFT), Chetumal, Mexico.

Guimaraes, E., Garcia, A., & Cai, Y. (2014, 21-25 July 2014). *Exploring Blueprints on the Prioritization of Architecturally Relevant Code Anomalies -- A Controlled Experiment.* Paper presented at the 2014 IEEE 38th Annual Computer Software and Applications Conference, Vasteras, Sweden.

Guimarães, E., Garcia, A., & Cai, Y. (2015). *Architecture-sensitive heuristics for prioritizing critical code anomalies*. Paper presented at the Proceedings of the 14th International Conference on Modularity, Fort Collins, CO, USA. https://doi.org/10.1145/2724525.2724567

Guimaraes, E., Garcia, A., Figueiredo, E., & Cai, Y. (2013, 18-19 May 2013). *Prioritizing software anomalies with software metrics and architecture blueprints.* Paper presented at the 2013 5th International Workshop on Modeling in Software Engineering (MiSE), San Francisco, CA, USA.

Gurgel, A., Macia, I., Garcia, A., Staa, A. v., Mezini, M., Eichberg, M., & Mitschke, R. (2014). *Blending and reusing rules for architectural degradation prevention*. Paper presented at the Proceedings of the 13th international conference on Modularity, Lugano, Switzerland. https://doi.org/10.1145/2577080.2577087

Hair, J. (2011). Multivariate Data Analysis: An Overview. In M. Lovric (Ed.), *International Encyclopedia of Statistical Science* (pp. 904-907). Berlin, Heidelberg: Springer Berlin Heidelberg.

Hair, J., Anderson, R. E., Tatham, R., & Black, W. (2015). *Multivariate data analysis*: New Jersey: Pearson education.

Hair, J., Black, W., Babin, B. J., & Anderson, R. E. (2014). *Multivariate data analysis (Seventh edition Pearson new international)*. London, UK: Pearson Education Limited.

Hair, J., Hult, G. T. M., Ringle, C., & Sarstedt, M. (2016). *A Primer on Partial Least Squares Structural Equation Modeling (SECOND EDITION)*. Thousand Oaks: SAGE, London.

Hall, T., Zhang, M., Bowes, D., & Sun, Y. (2014). Some Code Smells Have a Significant but Small Effect on Faults. *ACM Transactions on Software Engineering and Methodology, 23*(4), pp 1–39. doi:10.1145/2629648

Hassaine, S., Guéhéneuc, Y. G., Hamel, S., & Antoniol, G. (2012, 27-30 March 2012). *ADvISE: Architectural Decay in Software Evolution.* Paper presented at the 2012 16th European Conference on Software Maintenance and Reengineering, Szeged, Hungary.

Hayashi, S., Minami, F., & Saeki, M. (2018). Detecting Architectural Violations Using Responsibility and Dependency Constraints of Components. *IEICE Transactions on Information and Systems, E101.D*(7), 1780-1789. doi:10.1587/transinf.2017KBP0023

Henseler, J., Ringle, C. M., & Sarstedt, M. (2015). A new criterion for assessing discriminant validity in variance-based structural equation modeling. *Journal of the Academy of Marketing Science, 43*(1), 115-135. doi:10.1007/s11747-014-0403-8

Herold, S., Knieke, C., Schindler, M., & Rausch, A. (2020). *Towards Improving Software Architecture Degradation Mitigation by Machine Learning*. Paper presented at the ADAPTIVE 2020, The Twelfth International Conference on Adaptive and Self-Adaptive Systems.

Herold, S., & Mair, M. (2014, 2014//). *Recommending Refactorings to Re-establish Architectural Consistency.* Paper presented at the Software Architecture, Cham.

Herold, S., & Rausch, A. (2013, 18-19 May 2013). *Complementing model-driven development for the detection of software architecture erosion.* Paper presented at the 2013 5th International Workshop on Modeling in Software Engineering (MiSE), San Francisco, CA, USA.

Hertzog, M. A. (2008). Considerations in determining sample size for pilot studies. *Res Nurs Health, 31*(2), 180-191. doi:10.1002/nur.20247

Hinton, P. R., McMurray, I., & Brownlow, C. (2014). *SPSS Explained (2nd ed.)*. London: Routledge.

Hochstein, L., & Lindvall, M. (2005). Combating architectural degeneration: a survey. *Information and Software Technology, 47*(10), 643-656. doi:https://doi.org/10.1016/j.infsof.2004.11.005

Höck, M., & Ringle, C. M. (2006). *Strategic networks in the software industry: An empirical analysis of the value continuum.* Paper presented at the IFSAM VIIIth World Congress.

Imran, M. A. A., Lee, S. P., & Ahsan, M. A. M. (2017, 22-23 Aug. 2017). *Quality driven architectural solutions selection approach through measuring impact factors.* Paper presented at the 2017 International Conference on Electrical Engineering and Computer Science (ICECOS), Palembang, Indonesia.

Isela, M., Arcoverd, R., Garcia, A., Chavez, C., & von Staa, A. (2012, 27-30 March 2012). *On the Relevance of Code Anomalies for Identifying Architecture Degradation Symptoms.* Paper presented at the 2012 16th European Conference on Software Maintenance and Reengineering, Szeged, Hungary.

Izurieta, C., & Bieman, J. M. (2013). A multiple case study of design pattern decay, grime, and rot in evolving software systems. *Software Quality Journal, 21*(2 %J Software Quality Journal), 289–323. doi:10.1007/s11219-012-9175-x

Jin, W., Zhang, Y., Shang, J., Hou, Y., Fan, M., & Liu, T. (2023, 14-15 May 2023). *Identifying Code Changes for Architecture Decay via a Metric Forest Structure.* Paper presented at the 2023 ACM/IEEE International Conference on Technical Debt (TechDebt), Melbourne, Australia.

Kadri, S., Aouag, S., & Hedjazi, D. (2019, 15-16 Dec. 2019). *Multi-level approach for controlling architecture quality with Alloy.* Paper presented at the 2019 International Conference on Theoretical and Applicative Aspects of Computer Science (ICTAACS), Skikda, Algeria.

Kasi, P. M. (2009). *Research: What, Why and How?: A Treatise from Researchers to Researchers*. United State: AuthorHouse.

Klein, M. H., Kazman, R., Bass, L., Carriere, J., Barbacci, M., & Lipson, H. (1999). Attribute-Based Architecture Styles. In P. Donohoe (Ed.), *Software Architecture: TC2 First Working IFIP Conference on Software Architecture (WICSA1) 22–24 February 1999, San Antonio, Texas, USA* (pp. 225-243). Boston, MA: Springer US.

Kline, R. B. (2015). *Principles and practice of structural equation modeling (Third edition)*. United State: Guilford publications.

Lakshitha, d. S., & Balasubramaniam, D. (2012). Controlling software architecture erosion: A survey. *Journal of Systems and Software, 85*(1), 132-151. doi:https://doi.org/10.1016/j.jss.2011.07.036

Laser, M. S., Medvidovic, N., Le, D. M., & Garcia, J. (2020). *ARCADE: an extensible workbench for architecture recovery, change, and decay evaluation*. Paper presented at the Proceedings of the 28th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering, Virtual Event, USA. https://doi.org/10.1145/3368089.3417941

Lawshe, C. H. (1975). A quantitative approach to content validity. *Personnel Psychology, 28*(4), 563-575. doi:https://doi.org/10.1111/j.1744-6570.1975.tb01393.x

Le, D., & Medvidovic, N. (2016). *Architectural-based speculative analysis to predict bugs in a software system*. Paper presented at the Proceedings of the 38th International Conference on Software Engineering Companion, Austin, Texas. https://doi.org/10.1145/2889160.2889260

Lehman, M. M. (1979). On understanding laws, evolution, and conservation in the large-program life cycle. *Journal of Systems and Software, 1*, 213-221. doi:https://doi.org/10.1016/0164-1212(79)90022-0

Lehman, M. M. (1996). *Laws of software evolution revisited*, Berlin, Heidelberg.

Lei, P.-W., & Wu, Q. (2007). Introduction to Structural Equation Modeling: Issues and Practical Considerations. *Educational Measurement: Issues and Practice, 26*(3), 33-43. doi:https://doi.org/10.1111/j.1745-3992.2007.00099.x

Leite, P., Gonçalves, J., Teixeira, P., & Rocha, Á. (2014). Towards a model for the measurement of data quality in websites. *20*(4 %J New Rev. Hypermedia Multimedia), 301–316. doi:10.1080/13614568.2014.968638

Lenarduzzi, V., Saarimaki, N., & Taibi, D. (2019, 26-26 May 2019). *On the Diffuseness of Code Technical Debt in Java Projects of the Apache Ecosystem.* Paper presented at the 2019 IEEE/ACM International Conference on Technical Debt (TechDebt), IEEE.

Lenhard, J., Blom, M., & Herold, S. (2019). Exploring the suitability of source code metrics for indicating architectural inconsistencies. *Software Quality Journal, 27*(1), 241-274. doi:10.1007/s11219-018-9404-z

Lenhard, J., Hassan, M. M., Blom, M., & Herold, S. (2017). *Are code smell detection tools suitable for detecting architecture degradation?* Paper presented at the Proceedings of the 11th European Conference on Software Architecture: Companion Proceedings, Canterbury, United Kingdom. https://doi.org/10.1145/3129790.3129808

Li, Z., Li, X., & Kang, R. (2016, 19-21 Oct. 2016). *System resilience modeling and assessment based on minimal paths.* Paper presented at the 2016 Prognostics and System Health Management Conference (PHM-Chengdu), Chengdu, China.

Li, Z., Madhavji, N. H., Murtaza, S. S., Gittens, M., Miranskyy, A. V., Godwin, D., & Cialini, E. (2011). Characteristics of multiple-component defects and architectural hotspots: a large system case study. *Empirical Software Engineering, 16*(5), 667-702. doi:10.1007/s10664-011-9155-y

Lindvall, M., Tesoriero, R., & Costa, P. (2002, 4-7 June 2002). *Avoiding architectural degeneration: an evaluation process for software architecture.* Paper presented at the Proceedings Eighth IEEE Symposium on Software Metrics, Ottawa, ON, Canada.

Lindvall, M., Tvedt, R. T., & Costa, P. (2003). An Empirically-Based Process for Software Architecture Evaluation. *Empirical Software Engineering, 8*(1), 83-108. doi:10.1023/A:1021772917036

Little, R. J. A. (1988). A Test of Missing Completely at Random for Multivariate Data with Missing Values. *Journal of the American Statistical Association, 83*(404), 1198-1202. doi:10.1080/01621459.1988.10478722

Lohr, S. L. (2022). *Sampling Design and Analysis* (Vol. 3rd Edition): Chapman & Hall.

Lynn, M. R. (1986). Determination and Quantification Of Content Validity. *Nursing Research, 35*(6).

MacCormack, A., & Sturtevant, D. J. (2016). Technical debt and system architecture: The impact of coupling on defect-related activity. *Journal of Systems and Software, 120*, 170-182. doi:10.1016/j.jss.2016.06.007

Macia, I., Garcia, A., Chavez, C., & Staa, A. v. (2013, 5-8 March 2013). *Enhancing the Detection of Code Anomalies with Architecture-Sensitive Strategies.* Paper presented at the 2013 17th European Conference on Software Maintenance and Reengineering, Genova, Italy.

Macia, I., Garcia, J., Popescu, D., Garcia, A., Medvidovic, N., & Staa, A. v. (2012). *Are automatically-detected code anomalies relevant to architectural modularity? an exploratory analysis of evolving systems*. Paper presented at the Proceedings of the 11th annual international conference on Aspect-oriented Software Development, Potsdam, Germany. https://doi.org/10.1145/2162049.2162069

MacKenzie, S. B., Podsakoff, P. M., & Podsakoff, N. P. (2011). Construct Measurement and Validation Procedures in MIS and Behavioral Research: Integrating New and Existing Techniques. *MIS Quarterly, 35*(2), 293-334. doi:10.2307/23044045

Maffort, C., Valente, M. T., Terra, R., Bigonha, M., Anquetil, N., & Hora, A. (2016). Mining architectural violations from version history. *Empirical Software Engineering, 21*(3), 854-895. doi:10.1007/s10664-014-9348-2

Maisikeli, S. G. (2018, 28-29 Nov. 2018). *Measuring Architectural Stability and Instability in the Evolution of Software Systems.* Paper presented at the 2018 Fifth HCT Information Technology Trends (ITT), Dubai, United Arab Emirates.

Malhotra, R. (2016). *Empirical research in software engineering: concepts, analysis, and applications*: Chapman & Hall.

Mäntylä, M. V., & Lassenius, C. (2008). Software Metrics: A Survey of the State of the Art. *Software Quality Journal, 14*(1), 1-29.

Martin, R. C. (2003). *Agile software development: principles, patterns, and practices*. Upper Saddle River, NJ, USA: Prentice Hall PTR.

Medvidovic, N., Egyed, A., & Grünbacher, P. (2003). Stemming Architectural Erosion by Coupling Architectural Discovery and Recovery. *Proc. of the 2nd International Software Requirements to Architectures Workshop*.

Mendoza, C., Bocanegra, J., Garcés, K., & Casallas, R. (2021). Architecture violations detection and visualization in the continuous integration pipeline. *51*(8), 1822-1845. doi:https://doi.org/10.1002/spe.3004

Mirakhorli, M., & Cleland-Huang, J. (2011). *Tracing architectural concerns in high assurance systems (NIER track)*. Paper presented at the Proceedings of the 33rd International Conference on Software Engineering, Waikiki, Honolulu, HI, USA. https://doi.org/10.1145/1985793.1985942

Mirakhorli, M., & Cleland-Huang, J. (2016). Detecting, Tracing, and Monitoring Architectural Tactics in Code. *IEEE Transactions on Software Engineering, 42*(3), 205-220. doi:10.1109/TSE.2015.2479217

Misra, S. (2011). An Approach for the Empirical Validation of Software Complexity Measures. *Acta Polytechnica Hungarica, 8*.

Misra, S., Adewumi, A., Fernandez-Sanz, L., & Damasevicius, R. (2018). A Suite of Object Oriented Cognitive Complexity Metrics. *IEEE Access, 6*, 8782-8796. doi:10.1109/ACCESS.2018.2791344

Mo, R., Cai, Y., Kazman, R., Xiao, L., & Feng, Q. (2016). *Decoupling level: a new metric for architectural maintenance complexity*. Paper presented at the Proceedings of the 38th International Conference on Software Engineering, Austin, Texas. https://doi.org/10.1145/2884781.2884825

Mo, R., Cai, Y., Kazman, R., Xiao, L., & Feng, Q. (2021). Architecture Anti-Patterns: Automatically Detectable Violations of Design Principles. *IEEE Transactions on Software Engineering, 47*(5), 1008-1028. doi:10.1109/TSE.2019.2910856

Mohsin, S., Akhtar, H., Jalbani., Adil, A., Ahmed, A., & Kashif, M. (2017). Evaluating Dependency based Package-level Metrics for Multi-objective Maintenance Tasks. *International Journal of Advanced Computer Science and Applications(IJACSA), 8*(10). doi:http://dx.doi.org/10.14569/IJACSA.2017.081045

Mondal, A., Schneider, K., Roy, B., & Roy, C. (2022). A Survey of Software Architectural Change Detection and Categorization Techniques. *Journal of Systems and Software*.

Mourão, E., Kalinowski, M., Murta, L., Mendes, E., & Wohlin, C. (2017, 9-10 Nov. 2017). *Investigating the Use of a Hybrid Search Strategy for Systematic Reviews.* Paper presented at the 2017 ACM/IEEE International Symposium on Empirical Software Engineering and Measurement (ESEM).

Navarro Sada, & Maldonado, A. (2007). Research Methods in Education. *British Journal of Educational Studies, 4*(55), 469-470.

Nayebi, M., Cai, Y., Kazman, R., Ruhe, G., Feng, Q., Carlson, C., & Chew, F. (2019, 25-31 May 2019). *A Longitudinal Study of Identifying and Paying Down Architecture Debt.* Paper presented at the 2019 IEEE/ACM 41st International Conference on Software Engineering: Software Engineering in Practice (ICSE-SEIP), Montreal, QC, Canada.

Nguyen, P. H., Kramer, M., Klein, J., & Traon, Y. L. (2015). An extensive systematic review on the Model-Driven Development of secure systems. *Information and Software Technology, 68*, 62-81. doi:https://doi.org/10.1016/j.infsof.2015.08.006

Nuñez-Varela, A. S., Pérez-Gonzalez, H. G., Martínez-Perez, F. E., & Soubervielle-Montalvo, C. (2017). Source code metrics: A systematic mapping study. *Journal of Systems and Software, 128*, 164-197. doi:https://doi.org/10.1016/j.jss.2017.03.044

Oizumi, W., Garcia, A., Colanzi, T., Ferreira, M., & Staa, A. (2015). On the relationship of code-anomaly agglomerations and architectural problems. *Journal of Software Engineering Research and Development, 3*(1), 11. doi:10.1186/s40411-015-0025-y

Oizumi, W., Garcia, A., Da Silva Sousa, L., Cafeo, B., & Zhao, Y. (2016, 14-22 May 2016). *Code Anomalies Flock Together: Exploring Code Anomaly Agglomerations for Locating Design Problems.* Paper presented at the 2016 IEEE/ACM 38th International Conference on Software Engineering (ICSE), Austin, TX, USA.

Olbrich, S., Cruzes, D. S., Basili, V., & Zazworka, N. (2009, 15-16 Oct. 2009). *The evolution and impact of code smells: A case study of two open source systems.* Paper presented at the 2009 3rd International Symposium on Empirical Software Engineering and Measurement, Lake Buena Vista, FL, USA.

Olerup, A. (1991). Design Approaches: A Comparative Study of Information System Design and Architectural Design. *The Computer Journal, 34*(3), 215-224. doi:10.1093/comjnl/34.3.215 %J The Computer Journal

Olsson, T., Ericsson, M., & Wingkvist, A. (2017a). *Motivation and Impact of Modeling Erosion Using Static Architecture Conformance Checking.* Paper presented at the 2017 IEEE International Conference on Software Architecture Workshops (ICSAW), Gothenburg, Sweden.

Olsson, T., Ericsson, M., & Wingkvist, A. (2017b). *The relationship of code churn and architectural violations in the open source software JabRef*. Paper presented at the Proceedings of the 11th European Conference on Software Architecture: Companion Proceedings, Canterbury, United Kingdom. https://doi.org/10.1145/3129790.3129810

Organisation, I. S. (1991). *ISO/IEC 9126: Information Technology - Software Product Evaluation - Quality Characteristics and Guidelines for Their Use*.

Otero, C. (2012). *Software Engineering Design Theory and Practice (1st Edition)*. New York: CRC Press.

Palomba, F., Bavota, G., Penta, M. D., Fasano, F., Oliveto, R., & Lucia, A. D. (2018). On the diffuseness and the impact on maintainability of code smells: a large scale empirical investigation. *Empirical Software Engineering, 23*(3), 1188-1221. doi:10.1007/s10664-017-9535-z

Pan, J., Liu, Z., Li, D., Wang, L., & Li, B. (2023). An empirical study of software architecture resilience evaluation methods. *Journal of Systems and Software, 202*, 111726. doi:https://doi.org/10.1016/j.jss.2023.111726

Parnas, D. L. (1994). *Software aging*. Paper presented at the Proceedings of the 16th international conference on Software engineering, Sorrento, Italy.

Perry, D. E., & Wolf, A. L. (1992). Foundations for the study of software architecture. *17*(4 %J SIGSOFT Softw. Eng. Notes), 40–52. doi:10.1145/141874.141884

Petersen, K., Feldt, R., Mujtaba, S., & Mattsson, M. (2008). *Systematic mapping studies in software engineering*. Paper presented at the Proceedings of the 12th international conference on Evaluation and Assessment in Software Engineering, Italy.

Petersen, K., Vakkalanka, S., & Kuzniarz, L. (2015). Guidelines for conducting systematic mapping studies in software engineering: An update. *Information and Software Technology, 64*, 1-18. doi:https://doi.org/10.1016/j.infsof.2015.03.007

Pfeifer, S., Akgül, D., Roebenack, S., Tihlarik, A., Albert, B., Anacker, H., & Dumitrescu, R. (2022). *Design Decisions in the Architecture Development of Advanced Systems: Towards traceable and sustainable Documentation and Communication*. Paper presented at the Proceedings of NordDesign 2022.

Polit, D. F., & Beck, C. T. (2006). The content validity index: are you sure you know what's being reported? Critique and recommendations. *Res Nurs Health, 29*(5), 489-497. doi:10.1002/nur.20147

Reimanis, D., Izurieta, C., Luhr, R., Xiao, L., Cai, Y., & Rudy, G. (2014). *A replication case study to measure the architectural quality of a commercial system*. Paper presented at the Proceedings of the 8th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement, Torino, Italy. https://doi.org/10.1145/2652524.2652581

Riaz, M., Sulayman, M., & Naqvi, H. (2009). *Architectural Decay during Continuous Software Evolution and Impact of 'Design for Change' on Software Architecture.* Paper presented at the International Conference on Advanced Software Engineering and Its Applications, Berlin, Heidelberg.

Rizzi, L., Fontana, F. A., & Roveda, R. (2018). *Support for architectural smell refactoring*. Paper presented at the Proceedings of the 2nd International Workshop on Refactoring, Montpellier, France. https://doi.org/10.1145/3242163.3242165

Rocha, H., Durelli, R. S., Terra, R., Bessa, S., & Valente, M. T. (2017). DCL 2.0: modular and reusable specification of architectural constraints. *Journal of the Brazilian Computer Society, 23*(1), 12. doi:10.1186/s13173-017-0061-z

Roveda, R., Fontana, F. A., Pigazzini, I., & Zanoni, M. (2018). *Towards an architectural debt index.* Paper presented at the Proceedings - 44th Euromicro Conference on Software Engineering and Advanced Applications, SEAA 2018, Prague, Czech Republic.

Rubin, A., & Bellamy, J. (2012). *Practitioner's guide to using research for evidencebased practice*: Malaysia: John Wiley & Sons.

Ruiyin, L., Peng, L., Mohamed, S., & Paris, A. (2022). Understanding software architecture erosion: A systematic mapping study. *Journal of Software: Evolution and Process, 34*(3), e2423. doi:https://doi.org/10.1002/smr.2423

Ruiyin Li, Peng Liang, Mohamed Soliman, & Avgeriou, P. (2021, 20-21 May 2021). *Understanding Architecture Erosion: The Practitioners' Perceptive.* Paper presented at the 2021 IEEE/ACM 29th International Conference on Program Comprehension (ICPC), Madrid, Spain.

Sae-Lim, N., Hayashi, S., & Saeki, M. (2018). Context-based approach to prioritize code smells for prefactoring. *J Softw Maint Evol Proc, 30*(6), e1886. doi:https://doi.org/10.1002/smr.1886

Samah, B. A. (2016) Enhancing extension education research using structural equation modelling. In*. Inaugural lecture series*. Universiti Putra Malaysia (UPM).

Sangwan, R. S., Vercellone-Smith, P., & Neill, C. J. (2010). Use of a multidimensional approach to study the evolution of software complexity. *Innovations in Systems and Software Engineering, 6*(4), 299-310. doi:10.1007/s11334-010-0133-0

Santos, J. C. S., Suloglu, S., Ye, J., & Mirakhorli, M. (2020). Towards an Automated Approach for Detecting Architectural Weaknesses in Critical Systems. In *Proceedings of the IEEE/ACM 42nd International Conference on Software Engineering Workshops* (pp. 250–253): Association for Computing Machinery.

Santos, J. C. S., Tarrit, K., Sejfia, A., Mirakhorli, M., & Galster, M. (2019). An empirical study of tactical vulnerabilities. *Journal of Systems and Software, 149*, 263-284. doi:https://doi.org/10.1016/j.jss.2018.10.030

Sarkar, S., Maskeri, G., & Ramachandran, S. (2009). Discovery of architectural layers and measurement of layering violations in source code. *Journal of Systems and Software, 82*(11), 1891-1905. doi:https://doi.org/10.1016/j.jss.2009.06.039

Schwanke, R., Xiao, L., & Cai, Y. (2013, 18-26 May 2013). *Measuring architecture quality by structure plus history analysis.* Paper presented at the 2013 35th International Conference on Software Engineering (ICSE), San Francisco, CA, USA.

Segre, S. (2014). *Contemporary Sociological Thinkers and Theories*. United Kingdom: Ashgate Publishing, Ltd.

Sejfia, A. (2019, 27-27 May 2019). *A Pilot Study on Architecture and Vulnerabilities: Lessons Learned.* Paper presented at the 2019 IEEE/ACM 2nd International Workshop on Establishing the Community-

Wide Infrastructure for Architecture-Based Software Engineering (ECASE), Montreal, QC, Canada.

Sekaran, U., & Bougie, R. (2016). *Research methods for business: A skill building approach*. Chichester, England: john wiley & sons.

Shahbazian, A., Lee, Y. K., Le, D., Brun, Y., & Medvidovic, N. (2018, 30 April-4 May 2018). *Recovering Architectural Design Decisions.* Paper presented at the 2018 IEEE International Conference on Software Architecture (ICSA), Seattle, WA.

Shahbazian, A., Nam, D., & Medvidovic, N. (2018, 27 May-3 June 2018). *Toward Predicting Architectural Significance of Implementation Issues.* Paper presented at the 2018 IEEE/ACM 15th International Conference on Mining Software Repositories (MSR), Gothenburg, Sweden.

Sharma, T., & Spinellis, D. (2018). A survey on software smells. *Journal of Systems and Software, 138*, 158-173. doi:https://doi.org/10.1016/j.jss.2017.12.034

Shaw, M., & Clements, P. (2006). The golden age of software architecture. *IEEE Software, 23*(2), 31-39. doi:10.1109/MS.2006.58

Shaw, M., & Garlan, D. (1996). *Software architecture: perspectives on an emerging discipline*: Prentice-Hall, Inc.

Silva, M. D., & Perera, I. (2015, 18-20 Dec. 2015). *Preventing software architecture erosion through static architecture conformance checking.* Paper presented at the 2015 IEEE 10th International Conference on Industrial and Information Systems (ICIIS).

Steff, M., & Russo, B. (2011, 22-23 Sept. 2011). *Measuring Architectural Change for Defect Estimation and Localization.* Paper presented at the 2011 International Symposium on Empirical Software Engineering and Measurement, Banff, AB, Canada.

Stevanetic, S., Haitzer, T., & Zdun, U. (2014). *th*. Paper presented at the Proceedings of the 2014 European Conference on Software Architecture Workshops, Vienna, Austria. https://doi.org/10.1145/2642803.2642822

Stevens, J. P. (2012). *Applied multivariate statistics for the social sciences*: Routledge.

Stone, M. (1974). Cross-validatory choice and assessment of statistical predictions. *Journal of the royal statistical society: Series B (Methodological), 36*(2), 111-133.

Stringfellow, C., Amory, C. D., Potnuri, D., Andrews, A., & Georg, M. (2006). Comparison of software architecture reverse engineering methods. *Information and Software Technology, 48*(7), 484-497. doi:https://doi.org/10.1016/j.infsof.2005.05.007

Tabachnick, B. G., Fidell, L. S., & Ullman, J. B. (2007). *Using multivariate statistics* (Vol. 5): pearson Boston, MA.

Taciano, M. S., Serey, D., Figueiredo, J., & Brunet, J. (2019). *Automated design tests to check Hibernate design recommendations*. Paper presented at the Proceedings of the XXXIII Brazilian Symposium on Software Engineering, Salvador, Brazil. https://doi.org/10.1145/3350768.3351796

Tavares, G. E., Vidal, S., Garcia, A., Pace, A. D., & Marcos, C. (2018). Exploring architecture blueprints for prioritizing critical code anomalies: Experiences and tool support. *Software: Practice and Experience, 48*(5), 1077-1106. doi:https://doi.org/10.1002/spe.2563

Taylor, R., Medvidovic, N., & Dashofy, E. (2009). *Software architecture: foundations, theory, and practice*: John Wiley & Sons.

Teimour, A., Narmin Hassanzadeh, R., Yahya, K., & Farid, Z. (2011). Development and Evaluation of a New Questionnaire for Rating of Cognitive Failures at Work. *International Journal of Occupational Hygiene, 3*(1).

Tenenhaus, M., Vinzi, V. E., Chatelin, Y.-M., & Lauro, C. (2005). PLS path modeling. *Computational Statistics & Data Analysis, 48*(1), 159-205. doi:https://doi.org/10.1016/j.csda.2004.03.005

Terra, R., Valente, M. T., Czarnecki, K., & Bigonha, R. S. (2015). A recommendation system for repairing violations detected by static architecture conformance checking. *Software: Practice and Experience, 45*(3), 315-342. doi:https://doi.org/10.1002/spe.2228

Tonu, S. A., Ashkan, A., & Tahvildari, L. (2006, 22-24 March 2006). *Evaluating architectural stability using a metric-based approach.* Paper presented at the Conference on Software Maintenance and Reengineering (CSMR'06).

Tran, J. B., Godfrey, M. W., Lee, E. H. S., & Holt, R. C. (2000, 10-11 June 2000). *Architectural repair of open source software.* Paper presented at the Proceedings IWPC 2000. 8th International Workshop on Program Comprehension, Limerick, Ireland.

Vidal, S., Oizumi, W., Garcia, A., Díaz Pace, A., & Marcos, C. (2019). Ranking architecturally critical agglomerations of code smells. *Science of Computer Programming, 182*, 64-85. doi:https://doi.org/10.1016/j.scico.2019.07.003

Wang, T., & Li, B. (2019, 22-26 July 2019). *Analyzing Software Architecture Evolvability Based on Multiple Architectural Attributes Measurements.* Paper presented at the 2019 IEEE 19th International Conference on Software Quality, Reliability and Security (QRS), Sofia, Bulgaria.

Wang, T., Wang, D., & Li, B. (2019). *A Multilevel Analysis Method for Architecture Erosion*. Paper presented at the The 31st International Conference on Software Engineering and Knowledge Engineering.

Webster, J., & Watson, R. T. (2002). Analyzing the past to prepare for the future: writing a literature review. *26*(2 %J MIS Q.), xiii–xxiii.

Werts, C. E., Linn, R. L., & Jöreskog, K. G. (1974). Intraclass Reliability Estimates: Testing Structural Assumptions. *Educational and Psychological Measurement, 34*(1), 25-33. doi:10.1177/001316447403400104

Wetzels, M., Odekerken-Schröder, G., & Van Oppen, C. (2009). Using PLS path modeling for assessing hierarchical construct models: Guidelines and empirical illustration. *MIS Quarterly*, 177-195.

Whiting, E., & Andrews, S. (2020). *Drift and Erosion in Software Architecture: Summary and Prevention Strategies*. Paper presented at the Proceedings of the 2020 the 4th International Conference on Information System and Data Mining, Hawaii, HI, USA. https://doi.org/10.1145/3404663.3404665

Wohlin, C. (2014). *Guidelines for snowballing in systematic literature studies and a replication in software engineering*. Paper presented at the Proceedings of the 18th International Conference on Evaluation and Assessment in Software Engineering, London, England, United Kingdom. https://doi.org/10.1145/2601248.2601268

Wohlin, C., & Prikladniki, R. (2013). Editorial: Systematic literature reviews in software engineering. *55*(6 %J Inf. Softw. Technol.), 919–920. doi:10.1016/j.infsof.2013.02.002

Wohlin, C., Runeson, P., Hst, M., Ohlsson, M. C., Regnell, B., & Wessln, A. (2012). *Experimentation in Software Engineering*. Springer Berlin, Heidelberg: Springer Publishing Company, Incorporated.

Wong, S., Cai, Y., Kim, M., & Dalton, M. (2011, 21-28 May 2011). *Detecting software modularity violations.* Paper presented at the 2011 33rd International Conference on Software Engineering (ICSE), Honolulu, HI, USA.

Wynne, C. (1998). The partial least squares approach to structural equation modeling. *Modern Methods for Business Research, 295*(2), 295-336.

Xiao, L., Cai, Y., Kazman, R., Mo, R., & Feng, Q. (2016, 14-22 May 2016). *Identifying and Quantifying Architectural Debt.* Paper presented at the 2016 IEEE/ACM 38th International Conference on Software Engineering (ICSE), Austin, TX, USA.

Zamanzadeh, V., Ghahramanian, A., Rassouli, M., Abbaszadeh, A., Alavi-Majd, H., & Nikanfar, A. R. (2015). Design and Implementation Content Validity Study: Development of an instrument for measuring Patient-Centered Communication. *J Caring Sci, 4*(2), 165-178. doi:10.15171/jcs.2015.017

Zapalowski, V., Nunes, I., & Nunes, D. J. (2018a). *Understanding architecture non-conformance: why is there a gap between conceptual architectural rules and source code dependencies?* Paper presented at the Proceedings of the XXXII Brazilian Symposium on Software Engineering, Sao Carlos, Brazil. https://doi.org/10.1145/3266237.3266261

Zapalowski, V., Nunes, I., & Nunes, D. J. (2018b). The WGB method to recover implemented architectural rules. *Information and Software Technology, 103*, 125-137. doi:https://doi.org/10.1016/j.infsof.2018.06.012

Zengyang, L., Liang, P., Avgeriou, P., Guelfi, N., & Ampatzoglou, A. (2014). *An empirical investigation of modularity metrics for indicating architectural technical debt*. Paper presented at the Proceedings of the 10th international ACM Sigsoft conference on Quality of software architectures, Marcq-en-Bareul, France. https://doi.org/10.1145/2602576.2602581

Zhong, C., Huang, H., Zhang, H., & Li, S. (2022). Impacts, causes, and solutions of architectural smells in microservices: An industrial investigation. *Software: Practice and Experience, n/a*(n/a). doi:https://doi.org/10.1002/spe.3138

Zikmund, W. G., Babin, B. J., Carr, J. C., & Griffin, M. (2013). *Business research methods*. United State: Mc Graw Hill: Cengage learning.

Zude, L., & Jun, L. (2011). *A Case Study of Measuring Degeneration of Software Architectures from a Defect Perspective*. Paper presented at the Proceedings of the 2011 18th Asia-Pacific Software Engineering Conference, Ho Chi Minh City, Vietnam. https://doi.org/10.1109/APSEC.2011.51