Notice of Violation of IEEE Publication Principles

"A Systematic Literature Survey for Integrating Design Algebra with Object Oriented Design Methods in the Context of Software Architecture"
by P. Rajarajeswari, A. Ramamohan, D. Vasumathi, R. Sathiyaraj
In the Proceedings of the 2014 International Conference on Issues and Challenges in Intelligent Computing Techniques (ICICT), February 2014, pp. 638-646

After careful and considered review of the content and authorship of this paper by a duly constituted expert committee, this paper has been found to be in violation of IEEE's Publication Principles.

This paper contains copied portions of text from the paper cited below. The original text was copied without attribution (including appropriate references to the original author(s) and/or paper title) and without permission.

Due to the nature of this violation, reasonable effort should be made to remove all past references to this paper, and future references should be made to the following article:

"Software Architecture Optimization Methods: A Systematic Literature Review"
by Aldeida Aleti, Barbora Buhnova, Lars Grunske, Anne Koziolek, and Indika Meedeniya
in the IEEE Transactions on Software Engineering , Vol.39, No.5, May 2013, pp. 658-683

A Systematic Literature Survey for Integrating Design Algebra with Object Oriented Design Methods in the Context of Software Architecture

P.Rajarajeswari
Research Scholar
JNTUH
Assistant Professor-CSE
MITS, Madanapalle
perepicse@gmail.com

A.Ramamohan Reddy
Professor & Head
Department of CSE
S.V.University
Tirupathi, India

D.Vasumathi
Professor
Department of CSE
JNTUH
Hyderabad, India

R.Sathiyaraj
Assistant Professor
Department of CSE
MITS, Madanapalle
sathiyarajr@mits.ac.in

***Abstract*— To structure complex software systems by architecture models and specifications and also to provide a blueprint that is the foundation for later software engineering activities. Thanks to architecture specifications, software engineers are better supported in coping with the increasing complexity of today's software systems. We have performed a Systematic literature survey and also performed analysis of different papers from different research communities. We provide research analysis on software architecture field based on the survey. We consider that a survey could help to clear some of the issues, such as what are the main topics of the sure software architecture field and also what type of quality attributes are more relevant to consider when working with the architecture of a software system future research are. Furthermore current status and future directions in Software architecture field are presented**

Keywords: Software architecture, design alternatives, object oriented design, quality attributes

I. INTRODUCTION

Software systems were growing larger: systems of hundreds of thousands or even millions of lines of code were becoming commonplace. Clearly the foundations of the ideas underlying the field that is today called "software architecture "were laid by Parnas, Brooks Dijkstra and others in the 1960s through the 1980s.software architecture plays an important role to cope up with inherent difficulties of the development of large-scale and complex software systems [Clements & Northrop 96] .The gross-level organization [Bass98] of a software system is defined by software architecture. Architecture-based design is the area of research, which is focused for the solution of design problems. It is also taking as main driver for the organization of components in function of the quality attributes affecting the system. Software architecture community has been recognized and codified the architect [Shaw96, Buschmann96]. The set of design rules identify architecture styles and patterns and also Local or global constraints are used may be used to compose a system within each style with the components and connectors, together with on the way this composition should be done [Monroe97].

.

A. Problem Description and its Motivation

Various quality attributes of the architecture being developed with the process of architecture-based design [Bachmann00] consists of making certain decisions. Hence, architectural requirements admit multiple alternative realizations. Each design alternative may conform to these requirements in different ways, having different characteristics regarding performance, modifiability or security, among others. However, the elaboration of these designs is still a difficult, challenging problem [Tekidernogan00], because it usually involves a complex process of exploration, evaluation and composition of design alternatives. Once an initial architectural model has been outlined, developers tend to apply their knowledge and experience to evaluate candidate design options and then progressively refine the model to make it more consistent and complete. Therefore, design is still considered a creative activity [Shaw96] that must be practiced and learnt by experience and study of existing systems. Such environments should also provide means to capture the design rationale behind a set of principled design decisions. The goal of this report is to analyze the principal issues and problems of architectural-based design, describing several techniques designed to improve this process, such as architectural styles, quality attribute design mechanisms, scenarios and design methods, among others. The strengths and weaknesses of each technique are evaluated in the face of quality-attribute issues and architectural guidance. The work focuses also on the provision of semi-automated support for these techniques, in order to help designers in the exploration of design alternatives.

B. Research Approach and Contribution

A gateway to new approaches of architecture optimization can be opened, combining different types of architectural decisions during the optimization. Moreover, new tradeoff analysis techniques can be developed by combining results from different optimization domains. All of this can bring significant benefits to the general practice of architecture optimization. In general, with the survey we aim to achieve the following objectives:

Provide an overview of the current state of the art "Quality Optimization of Software Architecture and Design Specifications" in the architecture optimization domain

Identify current trends, gaps, and directions for future research.

We referred 200 journals from different journals which are published in journals and conferences. Initially, we derived taxonomy by performing a formal content analysis.

We collected a set of papers based on initial set of keywords and exclusion and inclusion criteria. A comprehensive overview of the current research in formal software architecture optimization is obtained by the selection and analysis of the papers.

Related Surveys: Various different implementation alternatives may be derived from the same conceptual architecture. Each alternative may have different adaptability, performance and reusability characteristics. Software is, however, rarely designed for optimal quality but rather it is a compromise of multiple considerations. The architecture design may be realized by applying object-oriented analysis and design methods in which a set of heuristic rules are provided to guide software engineers to analyze, design and implement object-oriented software systems. Current object oriented analysis and design methods do not provide explicit means for depicting the set of alternatives, prioritizing these alternatives and balancing them based on several quality factors such as adaptability, reusability and performance. To solve this problem we may derive concepts from the traditional engineering disciplines and observe how they manage the various design alternatives.

Beside this general SBSE survey, other surveys describe subareas of architecture optimization and design-space exploration that are only concerned with specific system Domains or a specific optimization method.

Although these surveys provide a good overview of a specific application domain, optimization method based on the existing research in the area of architecture optimization.

C. Organization

The remaining part of the paper organization is given below.

1) First, section 2 presents systematic literature review of research method and research analysis on Software architecture.

2) Literature review of the papers are given different definitions of Software architecture and also provide popular topics in software architecture in section 3

3) Section 4 provides Software architecture optimization process and existing literature on Quality attributes.

4) Current status of Software architecture field and future directions are given in section 5.

5) Conclusions are presented in Section 6.

II. Research Method

According to Kitchen ham systematic literature review consists of three phases: Planning, Conducting, reviewing the report. Research questions, Research steps and protocol of literature review. Section 2.1 presents the research questions. Section 2.2 provides derivation of research tasks. Section 2.3 then details the literature search step and highlights the inclusion and exclusion criteria. Finally, Section 2.4 discusses threats to the validity of our study.

A. Research Questions

Based on the objectives which are described in the introduction, derives the following research questions, and gives the basis for the literature review.

RQ1. At what levels of analysis does the current research on software architecture optimization be classified?

RQ2. What is the current state of software architecture optimization research based on this classification?

RQ3. How can do topics for further investigation which are derived from the Current research results?

B. Research Approach

Classification of research approaches into three categories as shown below.

TABLE I. RESEARCH APPROACH

Research approach Category	Description
Descriptive	Description of system
	Description of other systems
	Literature Review
Evaluative	deductive method
	Interpretive method
	Critical method
	Other methods
Formulative	framework
	guidelines
	model
	process, method, algorithm
	classification
	concept

C. Research Analysis on Software Architecture

Fundamental Research phase at 1985-1994: Initially several foundational ideas were firmly in place at mid-1980s and they travelled their own 15-20 year Red wine-Riddle cycles. Consider software elements as black boxes based on the contribution of Abstract data types, information hiding and other ideas.

Abstract data types and inheritance are the basic elements in Object oriented development. These ideas are developed by

Dijkstra and Parma's. But these ideas are not enough to produce the correct outcome for a computer program. Software qualities such as dependability and maintainability were also achieved by careful structuring. Deliberately-designed specialized software structures for specific problems are explored in the late 1980's.This work provide a catalog of existing systems for the identification of common architectural styles such as pipe-filter, repository, implicit invocation and cooperating process.

Notion on formulation phase at 1992-1996: Ideas are focused on the system structures that are occurred in systems and the results are emphasized for the description of system organization .Ideas on system organization, alternatives on object orientation, were elaborated by using programming languages. These languages are C2, Aesop, Darwin, Meta-H, Rapide and Wright. Architecture description languages' act as a vehicle for software architecture.

Formulization is also developed along with language development. Understanding of the relationship between architectural decisions and a system's quality attributes revealed software architecture validation as a useful risk reduction strategy. General architecture evaluation methods are Software Architecture Analysis method, attribute specific architecture analysis techniques. A series of international workshops on software architecture from 1995-2000 provide information about software architecture.

Improvement and extension phase at 1995-2000: This phase is focused on refining initial results. To move information between architecture description languages and to integrate with other design analysis and development tools.

The IEEE transactions on Software Engineering are special issues on Software engineering in 1995.ICSE 2000 conference on Software Architecture which is related to survey and a more sessions on architecture topics. WICSA began in 1998 and continues to present.

Internal enhancement and exploration phase at 1996-2003: ATAM and SAAM are the methods for Architecture analysis and evaluation methods. These are the methods for providing interaction among quality attributes. In this stage the effect of quality attributes increased with the role of architecture.

External enhancement and exploration phase at 1998-present: UML is the rational Unified Process, and also act as a tool-centered elegant idea of 4+1 views. It provides robust suite of tools for analysis, checking for consistency. Development of object-style architecture and consider public enthusiasm for object-orientedness by the object-oriented software framework. Component based software engineering movement is also to be established.

Popularization phase: 2000-present In this phase is characterized by production-quality, commercialized, supported along with marketed versions of the technology. Service oriented architecture, agent-based architectures, specific languages; tools and development environments are also established .Web based architecture platform which are IT connected are also established.

ACME/IEEE transaction on software engineering provides information about software architecture which is a part in software design.

TABLE II. RESEARCH ANALYSIS ON SOFTWARE ARCHITECTURE FIELD

Research phase	Description	Year
Fundamental Research phase	Establish the architecture for specific problems	1985-1994
Notion on formulation phase	Develop formal analysis of a specific architectural model and architecture evaluation	1992-1996
Improvement and extension phase	International conferences and journals on software architecture, ACME, Focus on classification	1995-2000
Internal enhancement and exploration phase	Provide framework for design analysis and development tools with architecture description languages and provide relationship between architecture to quality attributes.	1996-2003
External enhancement and exploration phase	Object oriented framework ,UML Models, component based software engineering, specific lifecycle models	1998
Popularization phase	Focused on production-quality and market version of technology and industry courses which are related to architectures.	2000

III. Most Supported Definitions On Software Architecture

" Software architecture of a program or computer system is the structure or structures of the system, which comprise software components, the externally visible properties of those components, and the relationships among them" Bengtson et al. 2004 [8] "

We found that analyzed definitions are not supported by 8 papers, 5 papers are not providing any definition at all. Remaining three papers provide definitions, but these were dropped.

TABLE III. ATTRIBUTES DEFINITION

Attributes	Definition	Number of papers
Design	SA is the set of design decisions, specification	**6**
Structure	SA is the structure of components, provide relationship among them and external properties	**11**
Constraints	SA defines the constraints on the realization or on the legal and illegal patterns in the communication between elements.	**Not presented**
quality	Quality attributes of software is influenced by SA.	**3**

The Popular research topics are represented in Table.IV

IV. Software Architecture Optimization Process

The main aim of the optimization task is typically maximize the quality of software-architecture under given constraints. Define the quality of software architecture directly by the Software experts is not possible. But it is related to a number of system attributes, called quality Attributes .The schematic diagram of Software architecture optimization process as shown below.

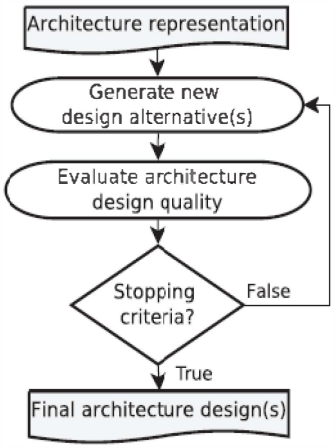

Figure 1: Software Architecture Optimization Process

First, the process input is architecture representation that describes the optimization of Software architecture. Second, the subcategory degrees of freedom describe changes of the architecture which are considered as variables in the optimization. Third, the subcategory quality evaluation can be used to describe the architecture quality evaluation procedures .It is used to make up the objective function(s) of the optimization process. Furthermore, to solve the formulated optimization problem by this technique used:

Subcategories are the overall optimization strategy and constraint handling. The optimization model may be derived from an architecture model. Finally, the architecture and the design decisions have to be encoded into an optimization model describing the decision variables and the objective function with the employment of optimization techniques. This is from a quality evaluation model. To assess the used architecture representation relevant for the user, we classify the approaches based on the input they require so that the possible values are "architecture model" (an architecture model is used as the input), "quality evaluation model" (a quality evaluation model is used as the input, no architecture model is used), and "optimization model" (an optimization model is used as the input, no architecture model or quality evaluation model is used). To start with an architecture model also internally use a quality evaluation model, with all several approaches also internally use an optimization model. An optimal found solution can be traced back to a meaningful solution on the architecture level if the quality evaluation models or optimization models are used as an input, it needs to be guaranteed.

TABLE IV. POPULAR RESEARCH TOPICS IN SOFTWARE ARCHITECTURE

Attributes	Description	Number of papers	Authors	Pub years
Notation	Addresses notation (or description language) is used for representing the SA	6	Perry and Wolf etal Medvidovic and Taylor Mageetal Luckham etal Langetal Garlanetal	1992 2000 1995 1995 2006 1997
Method	Addresses a method for analysis or evaluation of SAs.	1	Abrahamsson etal	2010
Tool	Addresses a tool is used for the analysis or evaluation of SAs	4	Monroe etal	1997
Evaluation	SA evaluation is performed by Evaluation with respect to one or more quality attributes	9	Sarkaretal Ozkayaetal O'Brienetal Immoenen and Niemedia Garlanetal Dorbrica and Nimela Bowers etal	2008 2008 2007 2007 2009 1995 2002 2009
Analysis	Pertains to SA analysis not leading to appreciation of quality attributes	4	Kazmenetal Hofmisteretal Docasse and polletetal Bouwers etal	2005 2007 2009 2010
scenario	Addresses scenarios as a technique analysis of SA.	9	Tekinerdoganetal Kazmanetal Benglstonetal Boschetal Bassetal Babaretal	2008 1998 2004 1998 2003 2004
Metrics	Addresses metrics act as tools for evaluation or analysis of SA	4	Kruchen Allenand garlan	1995 1994
Reviews	Addresses reviews as tools for analysis or evaluation of SA	3	Bouwers and Deursen Babaretal	2010 2004

Furthermore, we drill into the used architecture models can also be described in more detail, with the formalism of the Unified Modeling Language. AADL [203] or PCM [25] are other architecture description languages such as are subsumed in the value "ADL." We denote this latter case as model based (MB). For example, consider the quality attribute performance. A simple additive function that calculates the response time of a specific function sum up the response times of used individual services. A more complex nonlinear mathematical function is used if a queuing behavior of the system analyzed using exact queuing theory formulae. Finally, a model-based procedure is used if the system is represented as an extended queuing network and the performance is evaluated with approximate or simulation-based techniques. In essence, the optimization process aims at optimizing the quality attribute(s) whose evaluation constitutes the objective function(s), also referred to as fitness function(s) in the optimization domain. The architectural degrees of freedom category defines how the architecture representation can be changed to make it optimal with respect to the optimization goal. Example architectural degrees of freedom are component selection, allocation, or hardware parameter change. Thus, this category describes the types of variables of the optimization, i.e., it describes the types of design decision that can be varied by the optimization and thus defines the considered subset of the design space [132]. Another synonymous term is "architecture transformation operators" [101], [105]. More general terms describing the same idea are "design decisions" or "dimensions of variation" [166]. The possible values are those found in the reviewed papers, grouped by synonyms, since no existing classification (such as for quality attributes) is available to use; hence, we explain them in more detail in the next paragraphs.

A) FORMAL CONCEPT ANALYSIS:

Formal concept analysis can also be used for the optimization of software architecture .It comes from the mathematical field. It can be described as follows.

"The aim and meaning of Formal Concept Analysis as mathematical theory of concepts and concept Hierarchies is to support the rational communication of humans by mathematically developing appropriate conceptual structures which can be logically activated. [39]

The input for FCA is a context, which is a relation (typically a matrix) between objects and attributes. This relation forms ontology of concepts and their relationships. A concept can be considered a sub concept if its extension (the set of its objects) is contained in the extension of its super concept or, dually, if its intension (the set of its attributes) contains the intention of its super concept.

Problem: Survey of the papers provides problem-specific aspects. In the following, we summarize the main results for each problem subcategory. Literature review provides information about types of design goals with this architecture optimization approaches. Some quality attributes are addressed based on the analysis of the existing approaches. Numbers of papers are used generic approaches for the quality attributes.

B) MOST RELEVANT QUALITY ATTRIBUTES OF SOFTWARE ARCHITECTURE:

The relevant quality attributes are represented in Table-V

C) DISCUSSIONS:

So many similarities are there in between AVAILABILITY and RELIABILITY

Maintainability covers by Sarkar, bouwers,vandeursen during the years of 2008,2009,2010.Bass and John discussed about usability in the year 2003.Reusability covers by Molter and Garlan during the years of 1999,2009.Immonen and Nieiemel¨a [19] and Lung et al. 1997 [26], cover two quality attributes (respectively RELIABILITY and AVAILABILITY, and MODIFIABILITY and REUSABILITY). [5] Covers 6 quality attributes are discussed by Babar et al. 2004 . Finally, the paper is written by O'Brien et al. 2007 [31] covers 9 attributes. From the perspective of the quality attributes, three (SCALABILITY, TESTABILITY, and INTEROPERABILITY) are covered by a single paper (O'Brien et al. 2007 [31]). SECURITY is always covered together with other quality attributes (RELIABILITY, AVAILABILITY, PERFORMANCE, and USABILITY). AVAILABILITY is always covered together with RELIABILITY.

V. **Current status:** The concept of software architecture is used to run the Redwine-Riddle model, pretty much right on schedule. some of the resources are available for contemporary software architecture .These are 1) Off-the –shelf industrial training and certification programs 2) Standard architecture for countless domains and applications.3) End to end lifecycle models 4) Architecture evaluation and validation by using Robust and repeatable approaches 4) Architectural documentation 5) Robust tool environment for capturing the design 5) Commercial-quality architectural infrastructure layers 6) Career tracks.

A large-scale survey of UML use and its associated benefits and problems is described with this article "UML Software Architecture and Design Description" by Christian Lange, Michel Chaudron, and Johan Muskens. Current object-oriented analysis and design methods do not provide adequate means to describe and identify the possible design alternatives, i.e. the design space. so that , designing the object models by the software engineers with their knowledge, experience and intuition to compare the design alternatives.

Design algebra concepts can be integrated with the object oriented analysis and design methods can be used to improve the quality of software architecture and balancing quality factors for software architecture implementation.

TABLE V. RELEVANT QUALITY ATTRIBUTES OF SOFTWARE ARCHITECTURE

Quality attributes	Criteria	Number of papers	Percentage of occurrences	authors	year
Maintainability	How to maintain software easily	15	40%	Sarkar,bouwers,vandeursen	2008,2009,2010
Usability	How to use software easily	7	10%	Bass and John	2003
Reusability	How to reuse software easily	8	11%	Morlter, garlan,	1999,2000
Performance	The amount of useful work accomplished by software compared to the time and resources used	12	35%	Williams and smith	2002
Reliability .	To maintain performance under stated conditions for a stated period of time with the capability of software	17	43%	Tekiernerdogn	2008
Availability	The capability of a software product to be in a functioning state.	16	42%	Immeoneon and Niemela	2007
Security	The capability of software to prevent unintended usage and unintended access to its data.	15	41%	O'Brienteal	2007
Modifiability	How easily can software be modified	23	55%	Bengtesol	2004
Interoperability	How the software can interact with other systems	13	36%	O'Brienteal	2007
Testability	How to test the software easily	10	13%	O'Brienteal	2007
Scalability	The capability of software to handle growing workloads without prejudice of its service quality	18	45%	O'Brienteal	2007
Generic .	The quality attribute can be considered as a parameter in the evaluation	20	50%	0zkaya,kazman,Hofmeister	2008,1998,2007

Robert L. Nord and James E. Tomayko (recently deceased), presents "Software Architecture-Centric Methods and Agile Development" which is the interesting concept for Combining of leveraging the best aspects of architecture- centric approach methods and agile development. these can be used to help address quality attributes and maintain flexibility.

Michael Stal's article, SOA-based system's fundamental structures is explained by "Using Architectural Patterns and Blueprints for Service-Oriented Architecture, "uses published patterns. This subject area is particularly relevant at the moment.

"Using Architecture Models for Runtime Adaptability" which extends the need for architectural reasoning to runtime analysis by Jacqueline Floch, Svein Hallsteinsen, Erlend Stav, Frank Eliassen, Ketil Lund, and Eli Gjørven , because of the advent of software platforms that support component plug-ins and dynamic binding. It shows an interesting use of architecture models that are available at runtime and also takes

architecture in new directions, including mobile computing and runtime adaptability.

"Architecture Description Languages for High-Integrity Real-Time Systems" addresses architectural concerns beyond the typical structural ones and the use of an ADL for that purpose by Alek Radjenovic and Richard Paige,.

Finally, by Rogério de Lemos, Paulo Asterio de Castro Guerra, and Cecília Mary Fischer Rubira describes the process ingredients to arrive at a dependable system with the concept of "A Fault- Tolerant Architectural Approach for Dependable Systems", It sketches the role of exception handling in fault tolerance, and introduces an ideal fault-tolerant architectural element.

A) What's next?

More problems are raised in the production and use of real-world software in the area of software engineering research. Technical ideas often begin as qualitative descriptions of problems. We see that no of opportunities for new contribution in the area of Software architecture. Some of the promising areas are given below.

1) Architectural decisions and quality attributes are interrelated by formal relationships.
2) To find right language for the representation of architectures.
3) To assure conformance between architecture and code by finding ways
4) Test plans, test cases and other test artifacts can be applied on our approach, based on Software architecture.
5) To create reference materials for the organization of architectural knowledge.
6) To develop the architectural support for systems which can be dynamically adapt to changes in resources and user's expectations and preferences.

Some researchers are looking into architectural knowledge—that is, architectural design decisions and their rationale. Tools which are specially tailored for architects that will be made available to support their design, representation, analysis, and implementation tasks. The concept of aspects will affect research on the quality of software architecture.

VI. CONCLUSION

In this paper, we examined the state of Software architecture field from the point of view of the following questions:

1) What topics are addressed in the Software engineering research field?
2) What type of research methods are used in software architecture field?
3) What reference disciplines do SE researchers use as the theoretical basis for their publication?
4) What levels of analysis do Software engineering researchers conduct research?

REFERENCES

[1] L. Dobrica and E. Niemel¨a, "A survey on software architecture analysis methods," IEEE TSE, vol. 28, pp. 638– 653, Jul 2002.

[2] C. Hofmeister, P. Kruchten, R. L. Nord, H. Obbink, A. Ran,and P. America, "A general model of software architecture design derived from five industrial approaches," JSS, vol. 80, no. 1, pp. 106 – 126, 2007.

[3] R. Kazman, L. Bass, M. Webb, and G. Abowd, "SAAM:A method for analyzing the properties of software architectures," in ICSE. IEEE CS, 1994, pp. 81–90.

[4] R. Kazman, L. Bass, M. Klein, T. Lattanze, and L. Northrop,"A basis for analyzing software architecture analysis methods,"SQJ, vol. 13, pp. 329–355, 2005.

[5] P. Kruchten, "The 4+1 view model of architecture," IEEE Software., vol. 12, no. 6, pp. 42 –50, Nov 1995.

[6] C. Lange, M. Chaudron, and J. Muskens, "In practice: UML software architecture and design description," IEEE Software., vol. 23, no. 2, pp. 40 – 46, Mar/Apr 2006.

[7] D. Luckham, J. Kenney, L. Augustin, J. Vera, D. Bryan, and W. Mann, "Specification and analysis of system architecture using rapide," IEEE TSE, vol. 21, no. 4, pp. 336 –354, Apr1995.

[8] C.-H. Lung, S. Bot, K. Kalaichelvan, and R. Kazman, "An approach to software architecture analysis for evolution and reusability," in CCASCR, ser. CASCON '97. IBM Press, 1997, pp. 15

[9] D. E. Perry and A. L. Wolf, "Foundations for the study of software architecture," SIGSOFT SEN, vol. 17, pp. 40–52, Oct 1992.

[10] S. Sarkar, A. C. Kak, and G. M. Rama, "Metrics for measuring the quality of modularization of large-scale object t oriented software," IEEE TSE, vol. 34, no. 5, pp. 700–720, 2008.

[11] M. Shaw and P. Clements, "The golden age of software architecture," IEEE Software., vol. 23, no. 2, pp. 31 – 39,Mar/Apr 2006

[12] B. Tekinerdogan, H. Sozer, and M. Aksit, "Software architecture reliability analysis using failure scenarios," JSS, vol. 81, no. 4, pp. 558 – 575, 2008.

[13] L. G. Williams and C. U. Smith, "PASASM: a method for the performance assessment of software architectures," in IWSP, ser. WOSP '02. ACM, 2002, pp. 179–189.

[14] P. Clements, R. Kazman, and M. Klein, Evaluating Software Architectures: Methods and Case Studies. Addison-Wesley, 2001.

[15] R. Wille, "Formal concept analysis as mathematical theory of concepts and concept hierarchies," in FCA, ser. LNCS, B. Ganter, G. Stumme, and R. Wille, Eds., vol. 3626. Springer, 2005, pp. 1–33.

[16] R. Wille, "Formal concept analysis as mathematical theory of concepts and concept hierarchies," in FCA, ser. LNCS, B. Ganter, G. Stumme, and R. Wille, Eds., vol. 3626. Springer, 2005, pp. 1–33.

[17] J. Poelmans, P. Elzinga, S. Viaene, and G. Dedene, "Formalconcept analysis in knowledge discovery: A survey," in ICCS, ser. LNCS, M. Croitoru, S. Ferr´e, and D. Lukose, Eds., vol. 6208. Springer, 2010, pp. 139–153.

[18] M. Harman, "The Current State and Future of Search Based Software Engineering," Proc. Int'l Conf. Software Eng., L.C. Briand and A.L. Wolf, eds., pp. 342-357, 2007.

[19] M. Harman, S.A. Mansouri, and Y. Zhang, "Search Based Software Engineering: A Comprehensive Analysis and Review of Trends Techniques and Applications," Technical Report TR- 09-03, Dept. of Computer Science, King's College London, Apr. 2009.

[20] J. Henkel, R. Ernst, U. Holtmann, and T. Benner, "Adaptation ofPartitioning and High-Level Synthesis in Hardware/Software Co-Synthesis," Proc. IEEE/ACM Int'l Conf. Computer-Aided Design, pp. 96-100, 1994.

[21] ISO/IEC Standard for Software Engineering Product Quality, Int'lStandards Organization, ISO/IEC 9126-1, first ed., 2001.

[22] ISO/IEC Standard for Systems and Software Engineering—Recommended Practice for Architectural Description of Software-Intensive Systems, Int'l Standards Organization, ISO/IEC 42010 IEEE Std 1471-2000, first ed. 2007-07-15, p. c1-24, 2007.

[23] H. Jiang, C. Chang, D. Zhu, and S. Cheng, "A Foundational Study on the Applicability of Genetic Algorithm to Software Engineering Problems," Proc. IEEE Congress Evolutionary Computation, pp. 2210-2219, 2007

[24] A. Kishor, S.P. Yadav, and S. Kumar, "Application of a Multi-Objective Genetic Algorithm to Solve Reliability Optimization Problem," Proc. Int'l Conf. Computational Intelligence and Multimedia Applications, pp. 458-462, 2007

[25] B. Kitchenham, "Procedures for Performing Systematic Reviews," Technical Report TR/SE-0401, Dept. of Computer Science, Keele Univ., United Kingdom, 2004.

[26] A. Koziolek, "Automated Improvement of Software Architecture Models for Performance and Other Quality Attributes," PhD dissertation, Inst. fu¨ r Programmstrukturen und Datenorganisation (IPD), Karlsruher Inst. fu¨ r Technologie, Karlsruhe, Germany, July

[27] A. Koziolek and R. Reussner, "Towards a Generic Quality Optimisation Framework for Component-Based System Models," Proc. 14th Int'l ACM Sigsoft Symp. Component Based Software Eng., pp. 103-108, June 2011

[28] C. Lee, S. Kim, and S. Ha, "A Systematic Design Space Exploration of MPSoC Based on Synchronous Data Flow Specification," Signal Processing Systems, vol. 58, no. 2, pp. 193-213, 2010

[29] T. Mantere and J.T. Alander, "Evolutionary Software Engineering, a Review," Applied Soft Computing, vol. 5, no. 3, pp. 315-331, 2005.

[30] A. Martens, H. Koziolek, S. Becker, and R. Reussner, "Automatically Improve Software Architecture Models for Performance, Reliability, and Cost Using Evolutionary Algorithms," Proc. First Joint WOSP/SIPEW Int'l Conf. Performance Eng., pp. 105-116, 2010

[31] D.A. Menasce, V.A.F. Almeida, and L.W. Dowdy, Performance by Design. Prentice-Hall, 2004.

[32] D.A. Menasce´, E. Casalicchio, and V. Dubey, "A Heuristic Approach to Optimal Service Selection in Service Oriented Architectures," Proc. Seventh Workshop Software and Performance, A. Avritzer, E. J. Weyuker, and C.M. Woodside, eds., pp. 13-24,2008.

[33] Z. Michalewicz, "A Survey of Constraint Handling Techniques in Evolutionary Computation Methods," Proc. Conf. Evolutionary Programming, pp. 135-155, 1995

[34] M. Nicholson, A. Burns, and Y. Dd, "Emergence of an Architectural Topology for Safety-Critical Real-Time Systems," technical report, Dept. of Computer Science, Univ. of York, 1997

[35] M. Nicholson and D. Prasad, "Design Synthesis Using Adaptive Search Techniques and Multi-Criteria Decision Analysis," Proc.IEEE Int'l Conf. Eng. Complex Computer Systems, pp. 522-529, 1996.]

[36] A. Pretschner, M. Broy, I.H. Kru¨ ger, and T. Stauner, "Software Engineering for Automotive Systems: A Roadmap," Proc. Conf. Future of Software Eng., pp. 55-71, 2007

[37] O. Ra¨iha¨, "A Survey on Search-Based Software Design," Computer Science Rev., vol. 4, no. 4, pp. 203-249, 2010

[38] N. Trcka, M. Hendriks, T. Basten, M. Geilen, and L.J. Somers,"Integrated Model-Driven Design-Space Exploration for Embedded Systems," Proc. Int'l Conf. Embedded Computer Systems:Architectures, Modeling, and Simulation, pp. 339-346, 2011

[39] D.E. Perry, A.L.Wolf, 'Foundations for the Study of Software Architecture', Software Engineering Notes, Vol. 17, No. 4, pp. 40-52, October 1992

[40] M. Shaw, D. Garlan, Software Architecture - Perspectives on an Emerging Discipline, Prentice Hall, 1996