A Comprehensive Methodology for Creating and Evolving Reference Architectures

[Author Name]

Abstract

This paper presents a comprehensive methodology for creating, validating, and evolving reference architectures. We address key methodological gaps in the current literature and propose a structured approach that encompasses problem delineation, requirements specification, theory development, architectural design, validation, evaluation, implementation guidelines, and evolution strategies.

1 Introduction

Reference architectures play a crucial role in guiding the development of complex systems across various domains (Angelov et al., 2012). However, methodological gaps persist in their creation and evolution...

2 Related Work

Existing methodologies for creating reference architectures, such as Galster and Avgeriou's empirically-grounded approach (Galster and Avgeriou, 2011) and Nakagawa's process-based method (Nakagawa et al., 2014), provide valuable insights but have limitations...

3 Research Methodology

3.1 Problem Delineation

The first step in our methodology is a clear delineation of the problem and justification for the reference architecture. This aligns with the problem relevance principle in Design Science Research (DSR) (Hevner et al., 2004)...

3.2 Requirements Specification

We adopt ISO/IEC/IEEE 29148:2018 (International Organization for Standardization, 2018) for requirements specification, focusing on Architecturally Significant Requirements (ASRs) (Chen et al., 2013). ASRs are particularly relevant for reference architectures as they...

4 Artefact Design

4.1 Theory Development

We propose a two-step theory development process, drawing on DSR principles (Gregor and Jones, 2013):

4.1.1 Data Collection and Insight Gathering

This phase involves systematic literature reviews (Kitchenham and Charters, 2007), multivocal literature reviews (Garousi et al., 2019), and expert interviews (Bogner et al., 2009). We propose a novel taxonomic approach for categorizing findings...

4.1.2 Theory Formulation

Using abductive inference (Dubois and Gadde, 2014), we develop kernel theories and design theories. This process is iterative and...

4.2 Translating Theories into Architectural Constructs

We propose a rigorous method for translating theories into architectural components, incorporating variability management techniques (Galster et al., 2014)...

4.3 Architectural Representation

We advocate for the use of ISO/IEC/IEEE 42010:2011 (International Organization for Standardization, 2011) and ArchiMate (Lankhorst, 2017) for standardized architectural representation. This choice is justified by...

4.4 Component Explanation and Views

We propose a comprehensive set of views including structural, behavioral, and deployment models (Kruchten, 1995). Each component should be explained in terms of...

5 Validation and Evaluation

We propose a multi-method approach to validation and evaluation, including case studies (Runeson and Höst, 2009), expert evaluations (Beecham et al., 2005), and simulation (Martens et al., 2010)...

6 Implementation Guidelines

To bridge the gap between abstract architecture and concrete implementation, we propose...

7 Evolution Strategies

Reference architectures must evolve to remain relevant. We propose strategies for continuous refinement and adaptation (Eixelsberger et al., 1998)...

8 Threats to Validity

We acknowledge potential threats to validity in our methodology, including...

9 Discussion

Our proposed methodology addresses several key limitations of existing approaches...

10 Conclusion

This paper has presented a comprehensive methodology for creating, validating, and evolving reference architectures...

References

Angelov, S., Grefen, P., and Greefhorst, D. (2012). A framework for analysis and design of software reference architectures. *Information and Software Technology*, 54(4):417–431.

Beecham, S., Hall, T., and Rainer, A. (2005). Using an expert panel to validate a requirements process improvement model. *Journal of Systems and Software*, 76(3):251–275.

Bogner, A., Littig, B., and Menz, W. (2009). *Interviewing experts*. Springer.

Chen, L., Ali Babar, M., and Nuseibeh, B. (2013). Characterizing architecturally significant requirements. *IEEE software*, 30(2):38–45.

Dubois, A. and Gadde, L.-E. (2014). Abductive reasoning in logistics research. *International Journal of Physical Distribution & Logistics Management*.

Eixelsberger, W., Ogris, M., Gall, H., and Bellay, B. (1998). Software architecture reconstruction: Practice needs and current approaches. *IEEE Transactions on Software Engineering*, 24(9):797–812.

Galster, M. and Avgeriou, P. (2011). Empirically-grounded reference architectures: a proposal. In *Proceedings of the joint ACM SIGSOFT conference–QoSA and ACM SIGSOFT symposium–ISARCS on Quality of software architectures–QoSA and architecting critical systems–ISARCS*, pages 153–158.

Galster, M., Weyns, D., Tofan, D., Michalik, B., and Avgeriou, P. (2014). Variability in software systems—a systematic literature review. *IEEE Transactions on Software Engineering*, 40(3):282–306.

Garousi, V., Felderer, M., and Mäntylä, M. V. (2019). Guidelines for including grey literature and conducting multivocal literature reviews in software engineering. *Information and Software Technology*, 106:101–121.

Gregor, S. and Jones, D. (2013). The anatomy of a design theory. *Journal of the Association for Information Systems*, 8(5):312–335.

Hevner, A. R., March, S. T., Park, J., and Ram, S. (2004). Design science in information systems research. *MIS quarterly*, pages 75–105.

International Organization for Standardization (2011). Iso/iec/ieee 42010: 2011-systems and software engineering–architecture description.

International Organization for Standardization (2018). Iso/iec/ieee 29148: 2018 systems and software engineering—life cycle processes—requirements engineering.

Kitchenham, B. and Charters, S. (2007). Guidelines for performing systematic literature reviews in software engineering.

Kruchten, P. B. (1995). The 4+ 1 view model of architecture. *IEEE software*, 12(6):42–50.

Lankhorst, M. (2017). *Enterprise architecture at work*. Springer.

Martens, A., Koziolek, H., Becker, S., and Reussner, R. (2010). Automated evaluation of reference architectures: A case study. *Central European Journal of Computer Science*, 1(1):100–116.

Nakagawa, E. Y., Oquendo, F., and Becker, M. (2014). A process to create reference architectures for software ecosystems. In *European Conference on Software Architecture*, pages 320–337. Springer.

Runeson, P. and Höst, M. (2009). Guidelines for conducting and reporting case study research in software engineering. *Empirical software engineering*, 14(2):131.