Maturity Assessments of Service- oriented Enterprise Architectures with Iterative Pattern Refinement

Michael Falkenthal[1], Dierk Jugel[1], Alfred Zimmermann[1], René Reiners[2], Wilfried Reimann[3], Michael Pretz[3]

[1]Faculty of Informatics, University of Applied Sciences Reutlingen, Germany
firstname.lastname@reutlingen-university.de
[2]User-Centered Ubiquitous Computing, Fraunhofer FIT Sankt Augustin, Germany
rene.reiners@fit.fraunhofer.de
[3]Enterprise-Architecture & Innovation, Daimler AG, Germany
firstname.lastname@daimler.com

BibTeX:

```
@inproceedings{Falkenthal12,
  author    = {Michael Falkenthal and Dierk Jugel and Alfred Zimmermann and
               Ren\'{e} Reiners and Wilfried Reimann and Michael Pretz},
  title     = {Maturity Assessments of Service- oriented Enterprise
               Architectures with Iterative Pattern Refinement},
  booktitle = {Proceedings der Fachtagung INFORMATIK 2012, Stuttgart,
               GI-Edition Lecture Notes in Informatics (LNI)},
  year      = {2012},
  pages     = {1095--1101},
  series    = {Lecture Notes in Informatics (LNI)},
  volume    = {P-208},
  publisher = {Gesellschaft f\"{u}r Informatik e.V. (GI)}
}
```

Maturity Assessments of Service-oriented Enterprise Architectures with Iterative Pattern Refinement

Michael Falkenthal, Dierk Jugel and Alfred Zimmermann
Architecture Reference Lab of the SIL
Reutlingen University, Germany
first-name.last-name@reutlingen-university.de

René Reiners
User-Centered Ubiquitous Computing
Fraunhofer FIT
Sankt Augustin, Germany
rene.reiners@fit.fraunhofer.de

Wilfried Reimann and Michael Pretz
Enterprise-Architecture & Innovation
Daimler AG, Germany
first-name.last-name @daimler.com

Abstract: Current practices for assessing maturity of service-oriented enterprise information architectures only provide a sparse metamodel and pattern foundation and were rarely validated. This is a real problem for practical architecture assessments in repeated (cyclic) evaluations of service-oriented systems. In preliminary research we have developed and validated an original pattern language for supporting architecture assessments and optimization of enterprise systems, leveraging and extending base frameworks like the Capability Maturity Model Integration and The Open Group Architecture Framework. Traditionally, patterns are derived after long experience by an expert group of pattern authors. This may lead to a decelerated reuse of available design knowledge. Our approach intends to integrate available knowledge from enterprise information architecture methods, services computing and software architects directly from the beginning of the iterative pattern development and refinement process.

1 Introduction

The growing complexity of Enterprise Information Architectures is a challenge for many companies. Typical IT landscapes of enterprise systems consist of more or less process-integrated standard software packages, silos of legacy applications, and different infrastructure components. Innovation oriented companies have introduced services computing systems to assist in closing the gap between business and information technology and thus enabling business opportunities for service and emerging cloud computing paradigms in the context of emerging enterprise information architecture management approaches. One main problem today is the blurred transparency of this innovation change to system architectures based on services and cloud computing.

Our approach supports enterprise architects during architecture maturity assessments for service-oriented enterprise systems by extending our previous researched and validated architecture pattern language [ZLR11] by an iterative pattern formulation process [Re12]. Our pattern approach extends our previous work about architecture maturity frameworks, as in [Bu10] and [Zi11], and connects originally assessment pattern structures, like patterns of an architecture pattern language and collaborative pattern evolution process, with our maturity framework and our pattern evolution process. In this way our researched architecture patterns support enterprise information architects to investigate the ability of heterogeneous enterprise services-based systems. The base

architecture maturity framework integrates system architecture elements from convergent architecture methods, technologies and related software patterns, as in [Ga94], [Er09], and [Bu96] with evaluation methods for service-oriented enterprise systems [BKM07].

2 Architecture Maturity Model

The Open Group Architecture Framework (TOGAF) [T11] as the current standard for enterprise architecture provides the basic blueprint and structure for our enterprise software architecture domains of service-oriented enterprise systems. SOA is the computing paradigm that utilizes services as fundamental flexible and interoperable building blocks for both structuring the business and for developing applications. SOA promotes a business-oriented architecture style as promoted in [KBS04] and [Er09], based on best of breed technology of context agnostic business services that are delivered by applications in a business-focused granularity. To provide dynamic composition of services within a worldwide environment SOA uses a set of XML-based standards. A main innovation introduced by SOA is that business processes are not only modeled, but also combined services are executed from different orchestrated services.

In recent work, we have transformed the Capability Maturity Model Integration into a specific framework for architecture assessments of service-oriented enterprise systems. For this reason, we have combined CMMI with current SOA frameworks and maturity models. We used TOGAF and ideas related to the business and information architecture from [E12] as a basic structure for enterprise architecture spanning all relevant levels of service-oriented enterprise systems. We have analyzed and integrated related work about service-oriented architecture maturity models from [T11], [IA07], [O12], and others.

The metamodel for architecture evaluation enlarges the standardized CMMI, which is originally used to assess the quality of software processes and not the quality of software architectures. We have analyzed and systematically integrated evaluation criteria, maturity domains, architecture capabilities, and level rankings from state of the art SOA maturity and evaluation models. In addition, we have adapted architecture assessment elements from [Zi11] and [BKM07], and extended singular architecture patterns from our previous work [ZLR11] to our new architecture assessment patterns and the iterative architecture pattern refinement process (Section 3).

The SOAMMI architecture maturity framework introduces original architecture areas and organizes them within extended architecture domains, which are mainly based on TOGAF. Our intention was to leave most structural parts e.g. *Maturity Levels, Capability Levels, Specific Goals and Practices, Generic Goals and Practices* - of the original CMMI metamodel as untouched concepts. We extend these concepts of the metamodel by reclusively connected architecture patterns, as navigable architecture quality patterns of a pattern language, and enlarge these by other architecture specific structures and contents.

We have derived the architecture domains mainly from TOGAF where they are used as specific architecture subtypes and corresponding phases of the TOGAF-ADM (Architecture Development Method). Architecture areas cover assessable architecture artifacts and are correspondent, but very different, parts of process areas from CMMI.

To fit our architecture assessment scope, we have defined 22 original architecture areas of the SOAMMI framework, as in [Bu10] and [Zi11]), linked them to our architecture maturity levels and ordered them in line with our specific enterprise and software architecture domains. Each of the delimited architecture area is accurately described in a catalog including *name* of architecture area, *short identification* of architecture area and a *detailed description*.

3 Architecture Assessment Patterns

Although design patterns are mainly used to inform the design of a system, they are also applied as test cases for assessing software. Software architecture assessment patterns are based on the seminal work of software patterns originated from the work of [Zi11].

Our pattern language for architecture assessments of service-oriented enterprise systems provides a procedural method framework for the architecture assessment processes and for questionnaire design. This method framework of our new introduced pattern language was inspired from [RAZ11], and derived from the structures of the metamodel of SOAMMI as well as from our initial pattern catalog from previous research [ZLR11]. We organize and represent our architecture assessment patterns according to the following structures: *Architecture Domains, Architecture Areas, Problem Descriptions* - associated with *Specific Goals, Solution Elements* that are connected to *Specific Practices* and *Related Patterns*, which are subsequent connections of applicable patterns within the pattern language.

Connecting elements to specific practices of the SOAMMI framework indicates solutions for architecture assessments and improvements of service-oriented enterprise systems. This assessment and improvement knowledge is both verification and design knowledge, which is a procedural knowledge based on standards, best practices, and assessment experience for architecture assessments of service-oriented enterprise systems. It is therefore both concrete and specific for setting the status of service-oriented enterprise architectures, and helps to establish an improvement path for change. Patterns of our language show what to assess. Our patterns aim to represent verification and improvement knowledge to support cooperative assessments synchronizing people in cyclic architecture assessments.

Associated with our architecture assessment pattern language we have set up an assessment process to show how to assess architecture capabilities. This process is based on a questionnaire for architecture assessment workshops providing concrete questions as in [Zi11], answer types, and helping to direct and standardize the related assessment process. Additionally, we have included process methods for workshops, result evaluations, improvement path information for technology vendors and for application

organizations, as well as change support and innovation monitoring instruments. We have identified in [ZLR11] and distinguish a set of 43 patterns, as parts of a new researched and introduced pattern language in the context of 7 Architecture Domains and 22 Architecture Areas. Even though our architecture quality patterns accord to the Specific Goals, the Specific Practices and the Generic Goals from the SOAMMI framework, they extend these structures by navigable patterns as part of an architecture assessment language. Only this pattern structure enables architecture quality assessors to navigate easily in two directions to support the diagnostics and optimization process, and to provide a clear link to questionnaire and the related answer and result concepts.

Pattern Example: Business Product
Problem: How can we structure, design, model, and represent each business product as an origin for modeling business processes?

Solution:

- Structure business products for product lines
- Design business products by defining product structures and product rules
- Model and represent business products

Related Patterns: Business Services, Value Chain, Business Process, and Business Control Information

Pattern Example: Business Service
Problem: How can we structure, model, and represent each business service needed to support business products?

Solution:

- Structure business services for product types
- Design business services by defining service structures and service levels
- Model and represent business services

Related Patterns: Value Chain, Business Process, and Business Control Information

Figure 1: Pattern Example for Architecture Area "Business Products & Services"

Traditionally, much effort is put into the derivation and evaluation of patterns. However, we see the problem that many findings must be regarded earlier, at the state of an idea in order to be able to consider many findings in a flexible pattern set. This holds the chance to start working with patterns very early – even if it is not yet fully proven. Therefore, we give up the thought to force every pattern to be evaluated before its application. Our process wants to include new ideas and concepts into the project's lifecycle as early as possible. Over time, the idea, which is directly formulated as a *pattern candidate*, gets refined and evaluated. As soon as a pattern candidate is published in the pattern library, every registered user can provide feedback to the pattern or its formulation. It is also possible to support or refute the pattern statement by providing more references in favor of the pattern or against it. This way, the pattern maturity changes over time. To reflect the liveliness and bottom-up approach of the patterns in the design pattern library, we introduce the notion of a pattern's state that is used to track the development of the pattern over time. Our current implementation provides the following maturity states: *(i) Just created* patterns were recently submitted as a non-validated idea. (ii) Patterns *under consideration* look promising but still need further evaluation. (iii) P*attern candidates* are close to being approved. (iv) A*pproved* patterns are finalized within the pattern review process and settled design patterns. Currently, we have not yet defined a measure for the state of a pattern's maturity but consider the number of successful applications of a pattern as used by [GB08].

4 Conclusion and Upcoming Research

In this work we have developed suitable models for assessments of service-oriented enterprise systems. Our specific architecture assessment approach of the SOAMMI framework was founded on current architecture standards like TOGAF and architecture assessment criteria from related work approaches. The need for iteratively updating our assessment pattern collection motivated us to merge the efforts done for SOA assessment with a flexible and iterative pattern refinement and creation process. After talking about SOA maturity and assessment, we looked at the concept of involving *many* stakeholders into the pattern creation and evolution process and to adapt already available knowledge and findings from the project's domain as early as possible.

Our presented first approach of *iterative pattern refinement* allows for continuously evaluating gathered knowledge during the project's lifetime and makes patterns as well as pattern ideas available during the whole development process. Future work additionally has to consider conceptual work on both static and dynamic architecture complexity, and in connecting architecture assessment procedures with prognostic processes on architecture maturity with simulations of enterprise and software architectures. Additional improvement ideas include patterns for visualization of architecture artifacts and architecture control information to be operable on an architecture management cockpit. We are working at extending our pattern language to a full canonical form in order to support fully standardized cyclic architecture assessments for service-oriented products and solutions. The pattern evolution process represents a new aspect to the assembly and structuring of our patterns and will be further explored in the SOA assessment domain. Finally we will also apply the approach from this paper to the more holistic topic of the Enterprise Services Architecture Reference Cube ESARC from [ZZ11] and therefore the development of whole enterprise information architectures. The idea is to use the pattern-based iterative development method described in [BZ12] to derive a pattern language to support assessments of whole enterprise information architectures.

Acknowledgement

This paper extends ideas from the SOA Innovation Lab www.soa-lab.de, a major research and innovation network on Enterprise Architecture Management for Services and Cloud Computing in Germany and Europe.

References

[BKM07] Bianco, P.; Kostermanski, R.; Merson, P.: Evaluating a Service- Oriented Architecture. In: Engineering, September, pp. 1-91, 2007.

[Bu10] Buckow, H.; Groß, H.-J.; Piller, G.; Prott, K.; Willkomm, J.; Zimmermann, A.: Analyzing the SOA Ability of Standard Software Packages with a dedicated Architecture Maturity Framework. In: *EMISA,* 2010, pp. 131-143.

[Bu96] Buschmann, F.; Meunier, R.; Rohnert, H.; Sommerlad, P.; Stal, M.: Pattern-Oriented Software Architecture, Volume 1: A System of Patterns. Chichester, UK: Wiley, 1996.

[BZ12] Brunner, T.; Zimmermann, A.: Pattern-oriented Enterprise Architecture Management. In PATTERNS 2012 - The Fourth International Conferences on Pervasive Patterns and Applications, July 22-27, Nice, France 2012.

[E12] Essential Architecture Project [Online] Available: http://www.enterprise-architecture.org. [Accessed: 3-Jun-2012].

[Er09] Erl, T: SOA Design Patterns, Prentice Hall - Pearson, 2009.

[Ga94] Gamma, E.; Helm, R.; Johnson, R. E.; Vlissides, J.: Design Patterns. Elements of Reusable Object-Oriented Software, 1st ed. Amsterdam: Addison-Wesley Longman, 1994, p. 416.

[GB08] Thomas Grill and Margit Blauhut. Design Patterns Applied in a User Interface Design (UID) Process for Safety Critical Environments (SCEs). In Andreas Holzinger, editor, HCI and Usability for Education and Work, volume 5298 of Lecture Notes in Computer Science, pages 459–474. Springer Berlin / Heidelberg, 2008.

[IA07] Inaganti, S.; Aravamudan, S.: SOA Maturity Model, In: BP Trends, April, pp. 1-23, 2007.

[KBS04] Krafzig, D.; Banke, K.; Slama, D.: Enterprise SOA: Service- Oriented Architecture Best Practices (The Coad Series). Upper Saddle River, NJ, USA: Prentice Hall PTR, 2004.

[O12] ORACLE SOA Maturity Model [Online] Available: http://www.scribd.com/doc/2890015/oraclesoamaturitymodelcheatshe et. [Accessed: 3-Jun-2012].

[RAZ11] Reiners, R.; Astrova, I.; Zimmermann, A.: Introducing new Pattern Language Concepts and an Extended Pattern Structure for Ubiquitous Computing Application Design Support. In: PATTERNS 2011, Third International Conferences on Pervasive Patterns and Applications, 2011, no. c, pp. 61-66.

[Re12] Reiners, R.: A Pattern Evolution Process – From Ideas to Patterns. In (Gesellschaft für Informatik e.V. Hrsg.): Lecture Notes in Informatics - Proceedings Informatiktage 2012, Bonn, 2012, pp. 115-118.

[T11] TOGAF Version *9.1*. Van Haren Publishing, 2011.

[Zi11] Zimmermann, A.; Bukow, H.; Groß, H.-J.; Nandico, O. F.; Piller, G.; Prott, K.: Capability Diagnostics of Enterprise Service Architectures Using a Dedicated Software Architecture Reference Model. In: Services Computing, IEEE International Conference on, vol. 0, pp. 592-599, 2011.

[ZLR11] Zimmermann, A.; Laux, F.; Reiners, R.: A Pattern Language for Architecture Assessments of Service-oriented Enterprise Systems. In: PATTERNS 2011, Third International Conferences on Pervasive Patterns and Applications, 2011, no. c, pp. 7-12.

[ZZ11] Zimmermann, A.; Zimmermann, G.: ESARC-Enterprise Services Architecture Reference Cube for Capability Assessments of Service-oriented Systems. In: SERVICE COMPUTATION 2011 - The Third International Conferences on Advanced Service Computing, September 25-30, Rome, Italy, 2011.