Pattern-oriented Enterprise Architecture Management

Tobias Brunner
Reutlingen University, Faculty of Informatics
Architecture Reference Lab of the
SOA Innovation Lab, Germany
tobias1.brunner@student.reutlingen-university.de

Alfred Zimmermann
Reutlingen University, Faculty of Informatics
Architecture Reference Lab of the
SOA Innovation Lab, Germany
alfred.zimmermann@reutlingen-university.de

Abstract – **Current Enterprise Architecture Frameworks are limited in their ability of providing reference architectures and architecture development methods to assist for developing comprehensive, guided and fully-fledged enterprise architectures. The outcome is that in the majority of cases present architecture modeling approaches are rarely validated, have sparse reference and meta-models as well as general and limited statements for building and designing a comprehensive enterprise architecture. Furthermore these frameworks do not include recent approaches for developing enterprise architectures, like cloud computing, service-orientation and broad security reference architectures. This is a real problem of Enterprise Architecture Frameworks and their process models for designing and developing wide-ranging enterprise architectures. The Enterprise Services Architecture Reference Cube (ESARC) defines a integral framework for enterprise architecture artifacts with their main relationships. Our new idea and contribution is to extend existing architecture frameworks and their development methods with a new pattern-oriented Architecture Development Method approach. Based on our research on the ESARC architecture framework we have leveraged a new pattern-oriented Architecture Development Method and have extracted a coherent set of enterprise architecture patterns.**

Keywords – Enterprise Architecture Management; Enterprise-Services-Architecture-Reference-Cube; ESARC; Pattern-oriented Architecture Development Method; Core-Pattern-Catalog; Architecture Framework.

I. INTRODUCTION

Individual software solutions, legacy applications, and different infrastructure components lead to high costs and limited ability to respond quickly to new business requirements. Many companies start initiatives of Enterprise Architecture Management (EAM) and use Enterprise Architecture (EA) Frameworks like The Open Group Architecture Framework (TOGAF) [2, 3] to address these problems. But present Enterprise Architecture Frameworks are limited in their modeling approaches. These approaches are historically grown and do not cover current standards. New topics like cloud computing, service-orientation and especially the security aspect for enterprise architectures are not considered. In addition to these problems most of the EA-frameworks do not provide or provide an insufficient guided development method for building useful and suitable enterprise architectures.

The Enterprise Services Architecture Reference Cube (ESARC) [1] is a new Enterprise Architecture Framework derived from TOGAF and present standards like essential [4], the service model of ITIL [5] and from resources for service-oriented computing [6, 7, 8]. The current release of the ESARC is an original abstract architecture reference model, provides a integral approach for designing, developing, monitoring, evaluation and optimization of enterprise architectures over eight abstract Reference Architectures. Due to the new enterprise framework ESARC the hypothesis of this research integrates the idea to develop a new pattern-oriented [9] Architecture Development Method (ADM) [10] based on the TOGAF-Architecture Development Method [11] and the ITIL service-oriented-lifecycle and provides a relevant basis for high enterprise architecture innovation impacts for practice.

Based on the ESARC, the new Enterprise Architecture Development Method provides a process to create and manage enterprise architectures and integrates different patterns for a full iteration over the eight ESARC Reference Architectures and predefines basic architectural work products and artifacts. The whole development method and all integrated patterns are derived from current Enterprise Architecture Frameworks, process models and best practices from the Enterprise Architecture Management environment.

The aim of this research is to provide a core-pattern-catalog for a structured and guided development of enterprise architectures. Enterprise architects can expand the pattern-catalog for their own needs and requirements. The existing core-pattern-catalog comprises 26 patterns for developing essential artifacts of enterprise architectures.

Section II sets the base of our pattern approach by describing the ESARC – Enterprise Services Architecture Reference Cube. In Section III, we describe our new original pattern-oriented Architecture Development Method as a procedural framework. This architecture method mostly consists of two different types of patterns: “Architecture Development Method Patterns” represented in Section III and “Reference Architecture Development Patterns” embodied in Section IV. Section V describes the “Pattern Evolution Process” for the developed core-pattern-catalog. The conclusion in Section VI summarizes major achievements of this paper.

II. ENTERPRISE SERVICES ARCHITECTURE REFERENCE MODEL

ESARC – the Enterprise Services Architecture Reference Cube – is an abstract architecture reference model, which provides an integral view for main interweaved architecture types, and is derived primarily from state of art architecture frameworks. ESARC defines main architecture classification categories and abstracts from a concrete business scenario or technologies. The Open Group Architecture Framework (TOGAF) is the current standard for enterprise architecture and provides the basic blueprint and structure for our extended service-oriented enterprise software architecture domains like: Architecture Governance, Architecture Management, Business & Information Architecture, Information Systems Architecture, Technology Architecture, Operation Architecture, Security Architecture, and the Cloud Services Architecture.

The ESARC – Enterprise Services Architecture Reference Cube as seeded by [1], in Figure 1, unifies a set of close related reference models for essential architecture domains. The ESARC description in this section is fundamentally based on our previous research and extends this with our current presented pattern approach in a manageable and more procedural way. ESARC provides a coherent aid for examination, comparison, classification and quality ratings of specific architecture categories.

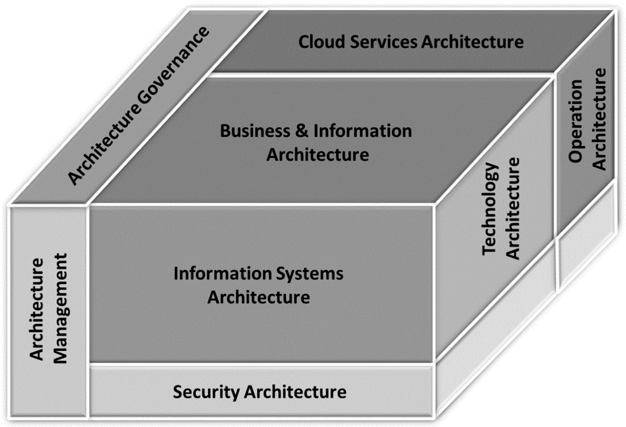

Figure 1. ESARC - Enterprise Software Architecture Reference Cube.

The Architecture Governance and Management framework provides the organizational context for the following main types of enterprise software architectures like Business & Information Architecture, the Information Systems Architecture, and the Technology Architecture.

Architecture Governance defines and maintains the Architecture Governance cycle. The Architecture Governance cycle sets the abstract governance frame for concrete architecture management activities within the enterprise or a product line development and specializes the more abstract Deming Cycle, as in [12], to the following management activities: plan, define, enable, measure, and control (see Section V. Reference Architecture Development Patterns with the Governance Cycle Pattern).

The second aim of Architecture Governance is to set rules for architecture compliance with internal and external standards. In addition policies for governance and decision definition are set to allow a standardized and efficient process for architecture decisions within the enterprise architecture organization. Enterprise and software architects are acting on a sophisticated connection path coming from business and IT strategy to the architecture landscape realization for interrelated business domains, applications and technologies. Architecture Governance has to set rules for the empowerment of people, defining the structures and procedures of an Architecture Governance Board, and setting rules for communication (see Section V. Reference Architecture Development Patterns with the Governance Board Pattern). With specifications from Architecture Governance we define main ESARC Architecture Management procedures for service-oriented enterprise software architectures: service strategy and life cycle management, service security, risk management, quality insurance for services, service testing, and service monitoring and control. Main management aspects include as well the metamodel-based management of service contracts and registries, and the reuse management of services in the enterprise.

The ESARC – Business & Information Reference Architecture defines the link between the enterprise business strategy and the results of supporting strategic initiatives through information systems. The Business & Information Reference Architecture provides a single source and comprehensive repository of knowledge, from which concrete corporate initiatives will evolve and link. This knowledge is model based and defines an integrated enterprise model of the business, which includes the organization and business processes. The Business & Information Reference Architecture sets the base for the business IT alignment. Important concepts of Business & Information Reference Architecture are: business and information strategy, the organization, and main business requirements for information systems like key business processes, business rules, business products, services, and related business control information.

The ESARC – Information Systems Reference Architecture provides an abstract blueprint of the individual solution architecture for application systems to be deployed and individually customized. The ESARC – Information Systems Reference Architecture contains the main application specific service types and defines their relationship by a layer model of building services. The core functionality of domain services is linked with application interaction capabilities and with the business processes of the customer's organization.

The ESARC – Technology Reference Architecture describes the logical software and hardware capabilities that are required to support the deployment of business, data, and application services. This includes IT infrastructure, middleware, networks, communications, processing, and standards. The layers of the ESARC – Technology Reference Architecture and the layers of the ESARC – Information Systems Reference Architecture correspond to each other.

III. Architecture Development Method Patterns

At this point it should be noted that the various Reference Architectures of the ESARC are strongly connected with each other. Therefore, the individual areas of Reference Architectures have to be developed in coordination with each other. All individual Reference Architectures will be developed in terms of their interdependence.

The pattern-oriented Architecture Development Method offers an entry point and a navigable process for iteration through the eight ESARC Reference Architectures. The whole Architecture Development Method is based on canonical structured patterns (Name, Classification, Problem, Solution, and Description) for the iteration through and for developing and designing architecture artifacts within the eight ESARC Reference Architectures.

The “Architecture Development Method (ADM) Pattern” is based on [11] and describes the pattern-oriented Enterprise Architecture Development Method. This pattern represents a procedural method (see Figure 2) for iterating through the eight Reference Architectures of the ESARC [1].

In addition, we derived from the “Architecture Development Method (ADM) Pattern” an iteration-loop (See Figure 3) by describing five subsequent patterns for iteration within each of the eight Reference Architectures of ESARC.

The aim is a core-pattern-catalog with architecture patterns for developing, structure- and designing integral enterprise architectures. Every enterprise architect can update, expand and enlarge the core-catalog with new patterns for its own needs and requirements.

The novelty of our new-presented pattern approach for Enterprise Architecture Management is its close fit with the integral classification framework from our previous research of ESARC and our specific coarse-granular EAM-process. Pattern approaches for Enterprise Architecture Management from the state of art are more visualization oriented and therefore more detailed, but lacking the overall integration in a more complex EAM-process.

Architecture Development Method Pattern

How can it be realized to set up an Architecture Development Method for developing a holistic enterprise architecture based on the ESARC?

Classification: Architecture Development Method

Problem: The absence of structured and concrete behavior guidelines for developing enterprise architectures ends up in complexity.

Besides these problems, the ESARC has eight highly aligned Reference Architectures they have to be developed in a structured, organized and controlled way.

Solution: An enterprise architect gets a structured way to iterate through the ESARC to build up integral enterprise information architectures. Enterprise architects should use practical guidelines for every Reference Architecture and descriptions for specified outcomes. The Architecture Development Method should be used as a support guideline and be adaptable for every enterprise.

Description: The Architecture Development Method maps the service-lifecycle of ITIL to a structured architecture development process based on the eight ESARC Reference Architectures. The “Architecture Development Method Pattern” is represented in Figure 2.

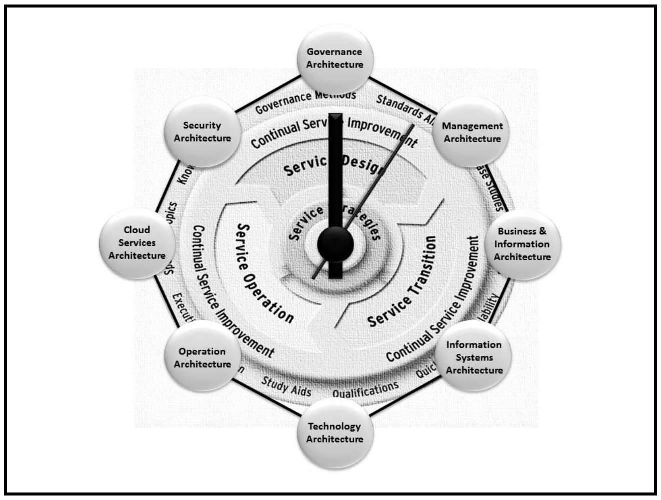

Figure 2. ESARC – Architecture Development Method Pattern.

The “Architecture Development Method Pattern” describes a structured and guided process to iterate through the ESARC and build up an enterprise architecture. The next step shows a further development of the already mentioned “Architecture Development Method Pattern”.

The enhancement is a uniform iteration-loop for each step within the eight Reference Architectures of the ESARC.

The iteration-loop is derived from the Deming Cycle [12] respectively the PDCA-Cycle and it also contains artifacts from TOGAF/ADM and the service-oriented lifecycle of ITIL. The iteration-loop defines a structured development cycle for every step within the “ADM-Pattern”. The combination of the “Architecture Development Method Pattern” with the mentioned iteration-loop is represented in Figure 3.

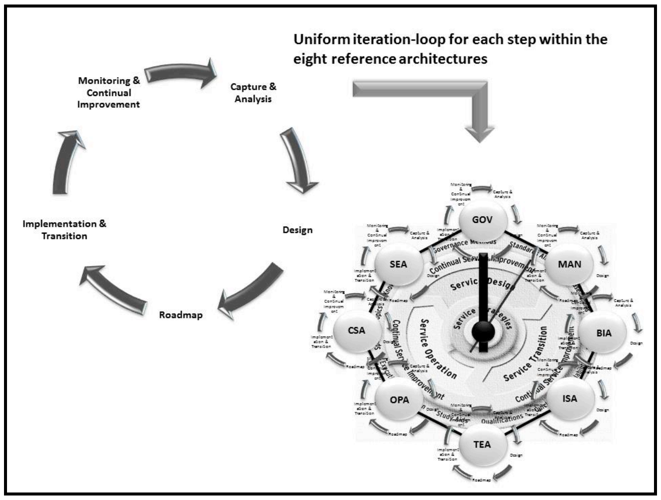
Uniform iteration-loop for each step within the eight reference architectures
Monitoring & Continual Improvement
Capture & Analysis
Design
Roadmap
Implementation & Transition
GOV
MAN
BIA
ISA
TEA
OPA
CSA
SEA

Figure 3. ESARC – Architecture Optimization Pattern.

The newly added iteration-loop relates to the eight Reference Architectures and contains five different steps for developing each of the Reference Architectures of ESARC consistently in the same way, but with a different particular focus.

For each step within the iteration-loop a pattern exists:

1) Architecture Capture & Analysis Pattern
2) Architecture Design Pattern
3) Architecture Roadmap Pattern
4) Architecture Implementation & Transition Pattern
5) Architecture Monitoring & Continual Improvement Pattern

In conclusion, the Architecture Development Method is defined by the "Architecture Development Method Pattern", as a recommended development sequence for the various Reference Architectures of the ESARC. Furthermore there is an iteration-loop defined by five patterns to build each of the eight ESARC Reference Architectures.

Within the Architecture Development Method the enterprise architect has to determine first the scope of Reference Architectures. Each of the Reference Architecture can have a different scope and maturity level, because the developed and recommended sequence of development in the architecture method is an iterative one. The maturity of the enterprise architectures and their outcomes will increase cyclically.

In this section, we have seen a procedural, pattern-based Architecture Development Method for iterating through the eight Reference Architectures of the ESARC by the "Architecture Development Method Pattern". In addition there was defined an extended iteration-loop for developing and designing architecture artifacts for each of the eight reference architectures of the ESARC.

IV. Reference Architecture Development Patterns

This section provides patterns for developing entities/artifacts for each Reference Architecture of the ESARC. The following mentioned patterns are designing and developing concrete architecture artifacts, so they are categorized as "Reference Architecture Patterns" (RA-Patterns). Currently, patterns are available for four Reference Architectures of the ESARC. Patterns available for building and designing the:

- Governance Reference Architecture
- Management Reference Architecture
- Business & Information Reference Architecture
- Cloud Services Reference Architecture

Following is shown a scenario of the core-pattern-catalog. There are RA-Patterns to develop, manage, evaluate and redesign the mentioned reference architectures. Two RA-Patterns (**Governance Board Pattern, Governance Cycle Pattern**) refer to the "Governance Reference Architecture" one RA-Pattern (**Service Lifecycle Pattern**) refers to the "Management Reference Architecture" and the last RA-Pattern that is shown in this paper (**Cloud Service Model Pattern**) refers to the "Cloud Services Reference Architecture". The following mentioned patterns show a small but typical part of the core-pattern-catalog.

Governance Board Pattern

How can it be ensured that important command and control tasks are addressed within an enterprise?

Classification: Reference Architecture - Governance

Problem: If command, control and other governance tasks are not transferred from one central location, chaos will result. It is important to form a central governance steering-position.

Solution: A Governance Board must be set up. The Governance Board owns all command and control functions within the enterprise and has all necessary rights and abilities to fully perform these activities. The board may transfer rights and delegate tasks to subsequent instances.

Description: A Governance Board should be created in each company. A Governance Board should fulfill the following tasks:

- Defining a clear mission/vision to lead and strategically align the company.
- Acquisition of corporate assets.
- Definition of corporate programs, tasks and services.
- All business opportunities and chances have to be analyzed and a strategic has to be created on the existing results and resources.
- Monitoring and compliance of legal and financial requirements. This includes the monitoring of the budget and providing the investments.
- Review of financial statements.
- Development of appropriate risk management practices and activities. Risk factors must be recognized and protected in the company.
- Determine individuals for leadership positions (e.g. the position of the CEO).

Governance in a company is not about implementation of tasks or requirements. Rather it is about monitoring, that the requirements be done and completed.

Governance Cycle Pattern

How can fundamental control- and management processes be implemented uniformly in an enterprise?

Classification: Reference Architecture - Governance

Problem: Companies are formed by complex structures and exchange relationships. A procedure to accomplish all enterprise command and control tasks is necessary.

Solution: The aim of the Governance Cycle is a fixed-cyclical approach for planning, defining, evaluating, measuring and controlling of all control, governance and management measures that established by the Governance Board.

Description: The Governance Cycle comprises the following architecture processes:

- Plan: The Governance Board schedules all necessary control and management tasks.

- Define: All tasks must be planned in detail. Actors and roles must be defined to be responsible for implementing and achieving planned goals.
- Enable: The Governance Board communicates all the objectives and tasks to the responsible persons, they ensure and guarantee that all objectives and tasks going to be reached and implemented.
- Measure: It is important that the progress of implementation, the ability of the actors and the success or failure of the goals and tasks will be measurable through appropriate performance indicators.
- Control: If the defined control and management functions are measurable, it is possible to recognized error-sources and potential for new development-possibilities. The results can be converted into a new governance cycle plan.

Service Lifecycle Pattern

How can the development of enterprise-wide services be targeted, standardized, successfully used and implemented?

Classification: Reference Architecture - Management

Problem: If external and internal services are developed without any structured and specified orientation to a consistent approach, there can be no assurance that the developed services are without redundancy and that they are goal-oriented and following the suggested strategy plan.

Solution: ITIL describes a continuous improvement process which is called the service-oriented lifecycle. Based on the strategic direction of the company (Service Strategy) the services are being developed (Service Design) and transferred to the operating mode (Service Transition). Then, the services are operated and so they are available for the overlying business processes (Service Operation). These phases are surrounded by a continuous improvement process (Continual Service Improvement) [5].

Description: The detailed description of the service lifecycle can be read in the five publications [13, 14, 15, 16, 17] of ITIL.
- Service Strategy - Service Design - Service Transition
- Service Operation - Continual Service Improvement

These five core documents form and detail the phases of an iterative and multidimensional service-oriented lifecycle for the company's existing- or to be created services.

RA - Cloud Service Model Pattern

*How can enterprise-resources be represented as services (*aaS) in a cloud-environment?*

Classification: Reference Architecture – Cloud Services

Problem: Due to the currently and not adequate existing service-models (IaaS, PaaS, SaaS) is it not possible to represent a whole and extensive architecture within an enterprise.

Solution: The previously existing service models must be expanded. The service models must be based on the enterprise structures. These services within all different levels of an enterprise architecture, have to be transferred and mapped into a wide range of enterprise service-models (*aaS) in the cloud-environment.

Description: The below illustrated subdivision of different services-models (*aaS) illustrates a potential way to structure and represent all enterprise resources and artifacts in a cloud-environment.

Examples of current discussed cloud services can be found in [18]: Testing-as-a-Service, Management-as-a-Service, Governance-as-a-Service, Application-as-a-Service, Process-as-a-Service, Information-as-a-Service, Database-as-a-Service, Storage-as-a-Service, Infrastructure-as-a-Service, Platform-as-a-Service, Integration-as-a-Service, Security-as-a-Service, Software-as-a-Service. Further *-as-a-Services are likely to follow.

In the section above, we described the so called "Reference Architecture Patterns" (RA-Patterns). These patterns provide design models and important artifacts which have to be fundamentally addressed by developing all eight Reference Architectures. These patterns can be used in the "**design**"-phase within the "Architecture Development Method Pattern" and its iteration-loop (see Section III, Architecture Development Method Patterns).

V. Pattern Evolution Process

The partially described core-pattern-catalog in the sections before can be centrally accessed via a web-application. The pattern-catalog can be used, enlarged, evaluated and new patterns can be easily added in a predefined canonical structure.

One continuative idea is to open the current pattern-catalog to a wide range of interested stakeholders and involve them to the pattern creation and evolution process and to adapt already available knowledge and findings from the project's domain as early as possible. For that possibility has to be an implementation of a well-defined Pattern Evolution Process within the core pattern catalog.

René Reiners [19] already introduced a Pattern Evolution Process and in our vision that evolution process will be merged with the established Architecture Development Method. Reiners introduced the notion of a pattern's state that is used to track the development of a pattern over time. The current implementation provides the following pattern status information:

- **Just created** patterns were recently submitted as a non-validated idea.
- **Under consideration** means that the pattern looks promising but needs further evaluation.
- **Pattern candidate** states that the pattern is close to being approved.
- **Approved** finalizes the pattern review process and settles the pattern as a design pattern.

The pattern-lifecycle process allows for continuously evaluating the design knowledge gathered during the project's lifetime and makes patterns as well as pattern ideas available during the whole development process. Successful findings or surprisingly failing results will be communicated as anti-patterns. In addition patterns can originate both from the project itself and from external sources.

Reiners [19] distinguishes three different categories:

- **Derived from project:** The pattern derives directly from the work within the project and will automatically be assigned the state under consideration. The pattern will be reviewed, perhaps re-worked and finally validated through an approval process by a validator.
- **Adapted to project:** The pattern originates from external sources, but has been adapted for use in the context of the project.
- **External:** The pattern exists in other pattern collections (e.g., a standard UI pattern) and is implemented in the current project's products and services.

With this background, the pattern-oriented Architecture Development Method can be developed in a comprehensive, structured and extensive knowledge-based manner.

VI. CONCLUSION

The research approach provides an innovative and so far not available Architecture Development Method, which is based on the our EA-Framework ESARC. Our new introduced Architecture Development Method relies on current standards for IT Enterprise Architectures like the TOGAF-Architecture Development Method, the service-oriented lifecycle of ITIL and the Deming Cycle.

We note that the entire Architecture Development Method is based on canonical structured patterns which form a basic core-pattern-catalog. The vision that is pursued with the pattern-based Architecture Development Method and the core pattern catalog is a constantly growing catalog. Enterprise architects can use the core catalog adapt them for their own needs and requirements and expand the catalog with new self-developed canonical structured architecture patterns.

The whole pattern catalog is public and can be centrally accessed, increased and evaluated via a web-application. The core pattern catalog was applied in a research assessment workshop with students, researchers and professors. That provided new transparent results for subsequent changes on the Architecture Development Method, the patterns and the related processes. The results of these assessments need to be interpreted in the context of company specific strategies and use cases.

The outcome of our research is a first cut of a new core-pattern-catalog for the guided development and architecture modeling. Additional future work has to consider additional patterns, because the current pattern catalog does not contain "Reference Architecture Patterns" for all reference architectures of ESARC.

An additional improvement idea deals with patterns for visualization of architecture artifacts and architecture control information to be operable on an architecture management cockpit. To improve semantic-based navigation within the complex space of EAM-visualization and service-oriented enterprise architecture management we are currently working on ontology models [20] for the ESARC – The Enterprise Services Architecture Reference Cube.

ACKNOWLEDGMENT

Our paper extends current work from the SOA Innovation Lab (www.soa-lab.de) Germany, which is a major business innovation network of service-oriented software technology and application companies.

REFERENCES

[1] A. Zimmermann and G. Zimmermann, "*ESARC - Enterprise Services Architecture Reference Cube for Capability Assessments of Service-oriented Systems.,* " The Third International Conferences on Advanced Service Computing, Rome, Italy, ISBN 978-1-61208-152-6, IARIA Proceedings of SERVICE COMPUTATION 2011, pp. 63-68, 25-30 September 2011.

[2] T. O. Group, "*The Open Group Architecture Framework Version 9.1,* " 2009. [Online]. Available: http://www.opengroup.org/togaf/. [Accessed 24 February 2012].

[3] P. R. Harrison and T. O. Group, "TOGAF 9 Foundation Study Guide", Wilco, Amersfoort: Van Haren Publishing, 2009.

[4] Essential, "*Essential Architecture Project*", [Online]. Available: http://www.enterprise-architecture.org. [Accessed 19 June 2011].

[5] OGC, Office of Governance Commerce, "*Introduction to ITIL*", UK London: Van Haren Publishing, 2005.

[6] C. MacKenzie, K. Laskey, F. McCabe, P. Brown and R. Metz, "*OASIS "Reference Model dor Service Oriented Architecture*" 1.0, OASIS Standard," 12 October 2006.

[7] J. Estefan, K. Laskey, F. McCabe and D. Thomton, "*OASIS Reference Architecture for Service Oriented Architecture*" Version 1.0, OASIS Committee Draft 02," 14 October 2009.

[8] J. Estefan, K. Laskey, F. McCabe and D. Thomton, "*OASIS Reference Architecture for Service Oriented Architecture*" Version 1.0, OASIS Public Review Draft 1," 23 April 2008.

[9] T. Erl, "*SOA Design Patterns*", Boston: PRENTICE HALL / Pearson Education, Inc, 2009.

[10] T. Brunner, "*Ein patternbasiertes Vorgehensmodell für den Enterprise Services Architecture Reference Cube,"* Masetr-Thesis, Reutlingen University, 2012.

[11] The Open Group, "*The Open Group Architecture Development Method (ADM),"* [Online]. Available: http://pubs.opengroup.org/architecture/togaf8-doc/arch/chap03.html. [Accessed 24 February 2012].

[12] W. E. Deming, "*Out of the Crisis*", Massachusetts: Massachusetts Institute of Technology, 1982.

[13] M. Iqbal and M. Nieves, "*ITIL® V3 Service Strategy*", The Stationery Office , 2007.

[14] C. Rudd and V. Lloyd, "*Service Design*", The Stationery Office , 2007.

[15] S. Lacy and I. Macfarlane, "*Service Transition*", The Stationery Office , 2007.

[16] O. o. G. Commerce, "*Service Operation*", The Stationery Office, 2007.

[17] G. Spalding, "*Continual Service Improvement*", The Stationery Office, 2007.

[18] D. S. Linthicum, "*Cloud Computing and SOA Convergence in Your Enterprise*," 2009.

[19] R. Reiners, "*A Pattern Evolution Process – From Ideas to Patterns*", Proceeedings Informatiktage 2012 Bonn - Germany, in Lecture Notes in Informatics, Vol. S-11,, pp. 115-118, 2012.

[20] S. Bourscheidt, T. Breuer, T. Brunner, B. Fetler and G. Fogel, "*ESARC - Referenzmodell und Ontologie für Enterprise Architecture Management*", Hochschule Reutlingen, Fakultät Informatik, Reutlingen, February 2012.