To Establish Enterprise Service Model from Enterprise Business Model

P. Jamshidi
Department of Industrial Engineering
Amirkabir University of Technology
p.jamshidi@aut.ac.ir

M. Sharifi
Department of Computer Engineering
Iran University of Science and Technology
msharifi@iust.ac.ir

S. Mansour
Department of Industrial Engineering
Amirkabir University of Technology
s.mansour@aut.ac.ir

Abstract

One of the key activities that are needed to construct a quality service-oriented solution is the identification of its architectural elements with the right granularity. The selection of an appropriate method for identification of services from business models of an enterprise is thus quite crucial to the success of any service-oriented solution for that enterprise. Existing methods for service identification ignore required performance metrics and semantics integrity of business elements; more importantly, they focus on entity-based services while ignoring process based ones. This paper proposes a new process for identification and specification of enterprise software services and their architectural elements. A novel clustering technique, named Elementary business process and business Entity Affinity analysis Technique (EEAT), is introduced for identifying candidate architectural elements. This technique identifies each candidate service with the right granularity, while satisfying low coupling, high cohesion, and low reuse cost principles for reusable software services.

1. Introduction

To develop a large scale enterprise application, the abstraction level has to be raised up to the level of business domains that the enterprise deals with [1]. Business aligned software services reside on such raised levels of abstraction. The Service-Oriented Architecture (SOA), as an instance of an "architectural style" [2] is currently the leading solution architecture for enterprise applications. In order to construct a service-oriented solution, the customary lifecycle steps consisting of analysis, design, realization, and implementation should be covered. It is widely accepted that traditional methods such as "object-oriented" and "component-based" methods are inadequate for constructing such solutions [3, 4]. Thus there is a need for an enhanced service-based approach.

SOA is not just about the products and standards that help to realize it, for example web services. It is concerned with other aspects for conducting the development of these solutions too. For example, analysis and design aspects of services are important and should be taken care of. In this paper, we are mainly concerned with the initial steps of constructing service-based solutions, i.e. service modeling [4].

Research on service-oriented computing, technologies, products, and standards has been abundant, but limited theoretical and practical experiences have been reported on service-oriented analysis and design. Even these limited works suffer from serious shortcomings. They are not practically applicable at enterprise levels, the identified services do not satisfy identification goals like managerial and technical performance metrics, and the proposed processes for service identification are ambiguous and not traceable.

After identifying the problem, some detailed modeling guidelines leading to a prescriptive modeling technique or process is necessary to answer the following question:

"How good service abstractions with acceptable performance metrics can be derived from high-level business requirements and business process models?"

We have set the following objective for our research reported in this paper:

"To propose a process for identifying and specifying proper service-oriented architectural elements at enterprise level from business models. By proper we mean to satisfy managerial and technical performance metrics."

We have assumed that the models are at the enterprise level and thus satisfy all enterprise's concerns. Furthermore, the proposed process is a forward identification process.

The paper is organized as follows. Section 2 presents the most related works. The components of enterprise business models are presented in Section 3. The enterprise service model is explained in Section 4. Section 5 proposes our new process for identification and specification of service model. Evaluation of the process is given in Section 6. Section 7 concludes the paper with some suggestions for future works.

978-0-7695-3283-7/08 $25.00 © 2008 IEEE
DOI 10.1109/SCC.2008.52

2. Related works

The most related works presented for service identification, service-oriented analysis and design, and clustering techniques are only discussed in this section.

Inganti et. al. [5] have focused on identification of enterprise level services and establishes the choreography of the services via business processes. The proposed methodology for service identification is limited to the service identification procedure and other aspects of methodology are ignored. A top-down (value chain analysis and use-case driven) and a bottom-up (exposing existing asset) service identification method is proposed. The identified services are correlated via activities of business processes. Although the proposed methodology is clear for service identification at enterprise level and it is not limited to the list of identified services, the measures for identification of business activities are not presented, it is not validated with any practical scenarios, and does not prescribe standard modeling notations like BPMN for process modeling and UML profiles for service-oriented architecture.

Zhang [6] has proposed a service-oriented reengineering process for reengineering the software architecture. His technique adopts the legacy components and wraps them as services identified in the domain model.

A brief overview of SOA modeling methodology and an analysis technique is given by Zimmermann et. al. in [1]. They have proposed a UML centric service identification approach. Although the possibility of deriving service models from business process models and use-case models with an integrated meta-model has been suggested, the derivation technique has not been given.

Zimmermann et. al. [4] further introduced the key elements of service model and a new discipline called Service-Oriented Analysis and Design (SOAD) for defining these elements. They believe that existing modeling disciplines such as Object Oriented Analysis and Design (OOAD), Enterprise Architecture (EA) frameworks and Business Process Modeling (BPM) techniques, fall short for service modeling. An interdisciplinary approach for constructing the elements of a service model is presented, but no procedure for constructing such a service model is given. He has also presented a model in [7] for service realization, with the main goal of proposing a prescriptive service realization methodology to simplify architectural decisions.

Arsanjani [3] has also presented the key activities of SOAD. A template for this type of architectural style is introduced and the importance of addressing the required techniques is discussed. Although the required activities, such as service identification and service specification are thoroughly explained, nothing is said on how to get them done.

Portier [8] has presented a notable taxonomy of SOAD, but the activities and techniques are not detailed so that it has become suitable for beginners to get acquainted with SOAD.

Johnson [9] has introduced some key themes to adapt RUP for service modeling. A UML profile for supporting the modeling notation in the SOAD context and some scenarios for validating these themes are proposed.

Teale et. al. [10] have proposed a method for software component identification. Their method is based on a commutative clustering on CRUD matrix made of business functions and domain entities. An approach for creating and using business patterns through a guiding architecture for providing a solution is also introduced.

None of the works cited here give a detailed procedure for identifying, specifying, and modeling proper architectural elements (with acceptable performance metrics) from business models. This paper tries to put forward such a procedure.

3. Enterprise business model as input artifact

Enterprise business model is actually composed of a collection of smaller artifacts as given in Table 1. One of the main artifacts of the model is the enterprise business process model that represents a collection of coordinated activities, either manual or automated, that provide added value to one or more internal or external clients [11].

Table 1. Enterprise business model's Components [11]

Artifact	Description
Enterprise business rules specification	The definition of the constraints that influence or guide the everyday workings of an organization
Enterprise business process model	Captures the fundamental business processes, the external entities, and the major workflows between them
Enterprise domain model	Depicts the main business entities of interest to an organization and their relationships
Enterprise mission statement	A statement of the strategies to be followed to achieve the enterprise vision
Enterprise vision	A statement of the primary goal(s) of an organization
Organization model	A definition of the location, positions, organizational units, and their interrelationships within an enterprise

An important part of enterprise business modeling is the creation of a high-level domain/conceptual model that depicts the main business entities and their relationships that are of interest to an organization [11].

In this paper we consider an enterprise business process model as an input artifact, regardless of the technique by

which the model has been constructed. We only suggest and assume that the model has the lowest level of details in order to become useful as a conceptual view. The Elementary Business Process (EBP) serves this purpose all right. Definitions of what constitutes an EBP differ in the literature, but we base our definition on the following more common cited:

"A process performed by one person in one place at one time which adds significant value and which leaves data in a consistent state [12]*."*

Another input of our proposed process is the enterprise entity model that prescribes that enterprise domain model should be analyzed to identify business entities such that each of entity is created in one business process and used in others. So we have two criteria for constructing a business model:

- Business processes should be modeled at the lowest level needed at the conceptual view.
- The granularity of business entities should be at the level which each of them is created only in one elementary process.

4. Enterprise service model as main artifact

Enterprise service model is a model of the core elements of a Service-Oriented Architecture and used as an essential input to activities in implementation and testing. The service model is an abstraction of the software services implemented within an enterprise and supporting the development of one or more service-oriented solutions. It is used to conceive, as well as document, the design of software services. It is a comprehensive, composite artifact encompassing all services, providers, specifications, partitions, messages, collaborations, and the relationships between them [3, 13].

The service model is at the same level of abstraction as the design model; the difference between them is primarily in the granularity of services and their specifications in comparison with design classes and design subsystems in general. It is also expected that component technologies are the most likely implementation choice for services.

The service model can be used for different levels of scope [13]:

- Service-scoped development, where the scope of the project is to develop the service independently (as much as possible) from other services.
- Project-scoped development, where a project involves the specification of a number of services in support of a set of application requirements.
- Enterprise-scoped development, or service portfolio management, where the scope is only to capture the service specifications and logical partitioning but at an enterprise-wide scope. This allows designers and architects to make wide ranging decisions about the entire portfolio, yet separate projects are required to develop the design and implementation models for the identified services (and client applications).

Our proposed process is at enterprise level. It uses a novel technique that is best fitted at enterprise level and satisfies enterprise concerns, although it can be applied at project level too.

5. The proposed process

One of the key activities of the service-oriented analysis and design (i.e. service-oriented modeling) is the service identification and specification. Figure 1 shows the activities and the responsibilities of developers in building an enterprise service model from an enterprise business model. The former model can be derived accordingly in four steps: 1) modeling of business processes, 2) identification of service model elements, 3) categorization of services, and 4) documentation of enterprise service model, as is described in the following.

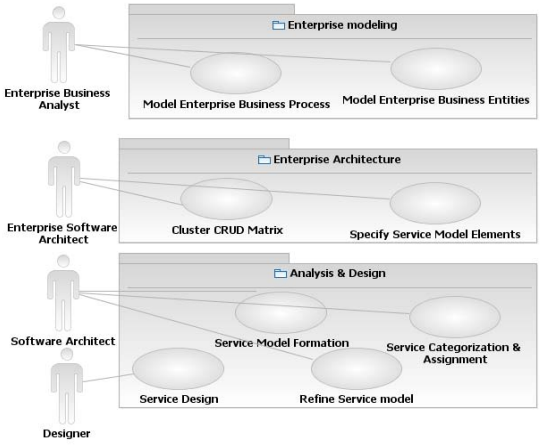

Figure 1. Deriving ESM from EBM process

i. Modeling of business processes

First of all the model of business processes and business entities at the enterprise level with the criteria set before should be constructed. The responsibility of this work is assigned to the business analyst of the specified enterprise. So the result of this activity is the business process and business entity models at the right granularity.

ii. Identification of service model elements

Having defined the required business processes and business entities, a first cut service architecture can be derived by comparing the identified EBPs and BEs. We use a new clustering technique, named Elementary business process and business Entity Affinity analysis Technique (EEAT) in this step. A matrix whose rows are the identified EBPs and the columns are the identified BEs is formed. The following tags are put in the cells then: (priorities as C>U>D>R):

- "C" meanings this EBP CREATES an instance of this BE.
- "R" meanings this EBP READS an instance of this BE.
- "U" meanings this EBP UPDATES an instance of this BE.
- "D" meanings this EBP DELETES an instance of this BE.

Every column (BE) must exactly have one create operation and each row (EBP) must have no conceptual (operational) activities. The tags need not be precise, particularly for read operations. The matrix should be analyzed by using the affinity analysis and clustering technique. The objective is to deduce groups of EBPs and BEs that share create and update operations. The model is adjusted to bring together business processes and business entities with strong affinity. Description of the detailed algorithm used (North West) for clustering is out the scope of this paper, however the result is appearance of mutually exclusive sections (areas) of EBPs and BEs formed around create and update operations. These sections are candidate identified services as shown in Figure 2.

Grouping together all elementary business processes that create and update the same entities using the mentioned clustering technique defines non-redundant building blocks (Enterprise Services). It can be used to construct enterprise applications that in turn support particular business processes. By encapsulating the process and entity into services, software reuse becomes feasible and the hidden benefit of enterprise architecture emerges. After identification of services, other architectural elements of service model should be specified.

The structural and behavioral specification of each service can be specified after service identification through the matrix. The EBPs that is located besides the service section in the first column indicate the behavioral elements of services. For example, EBP1 to EBP8 shape the operations of the service1 in Figure 2, and BE1 and its elements can be used as parameters of service1 operations. CRUD notations located outside the service boundaries can be used as helping media to identify service channels. For example, service channel1 indicated with the U tag within (EBP4, BE9) relates service4 to service1.

A partition may be used to represent any particular organization that an architect may wish to focus on [13]. Service partitioning can be also indicated in the matrix as depicted in Figure 2 in order to show the elements of the service model.

Business processes that are modeled in the first step are marked as service consumers (Figure 2). Each identified service has a service provider that is analyzed further in the design model artifact. For example, service2 has 7 use-cases and 4 entities (Figure 2). It is also clustered to help the identification of the service components. This artifact is intended to be used in describing the realization of a service specification.

Nine activities must thus be carried out in the second step:

1. Matrix Formation
2. Column Commutation
3. Clustering
4. Service's name indication
5. Service Specification indication
6. Representing Service Collaboration with Service Channel
7. Service Partition indication
8. Service Consumers marking
9. Modeling Service Providers with Service Components

iii. Categorization of services

The identified services provide a stable part of the overall solution architecture (such as service interface, business entity components, data access components, data access components in Microsoft solution architecture), which is then completed with other elements. Most of solution architectures for realizing SOA utilize Business Process Management System (BPMS) tools. BPMS is used for service choreography in service-oriented solutions and has modules for business process design, running and monitoring. Choosing the right segmentation of business logic between processes that are embedded in BPMS and external services is sometimes challenging and crucial. One rule is that external services should be responsible for computationally intensive or complex logic; whereas processes contain logics that may change in response to changes in business requirements.

So some identified services are modeled in BPMS as a portion of business process and others that are more computationally complex are implemented in service development tools and collaborate with BPMS via specified protocols.

iv. Documentation of enterprise service model

The final step to build an ESM is to clearly document the identified architectural elements with CASE modeling tools. In this paper the UML profile for software services [14] is used for constructing enterprise service model. It is a profile for UML 2.0 that allows for modeling of services, service-oriented architecture (SOA), and service-oriented solutions.

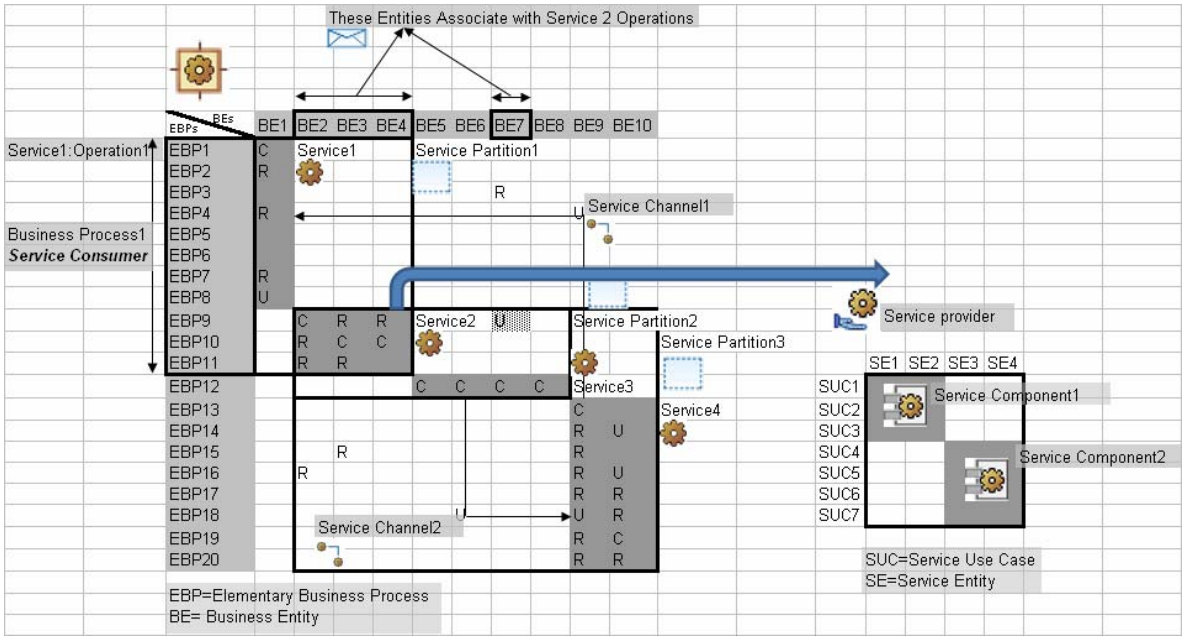

Figure 2. Clustered CRUD matrix

The profile has been implemented in IBM Rational Software Architect (RSA) and IBM Rational Software Modeler (RSM), used successfully in developing models of complex customer scenarios, and used to help educate people about the concerns relevant to developing service-oriented solutions.

The aim of the profile is to provide a common language for describing services, one which covers a number of activities through the development lifecycle and also provides views to different stakeholders [14]. Different views of the enterprise service model are depicted in Figures 3, 4, 5 and 6. Different viewpoints of the service-oriented solutions are illustrated in Figure 3 encompassing message view, service view, and collaboration view. Generally, the message view contains class models representing the messages provided in and out of a service (Figure 4). The collaboration view shows the collaboration of services implied by their contained business processes (Figure 5). The service view serves to contain the service definitions, specifications, providers, partitions, and dependencies/ associations between them (Figure 6).

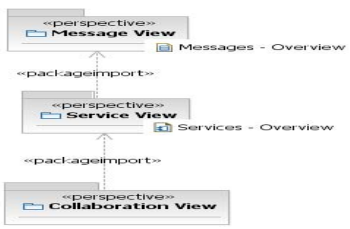

Figure 3. Service model overview

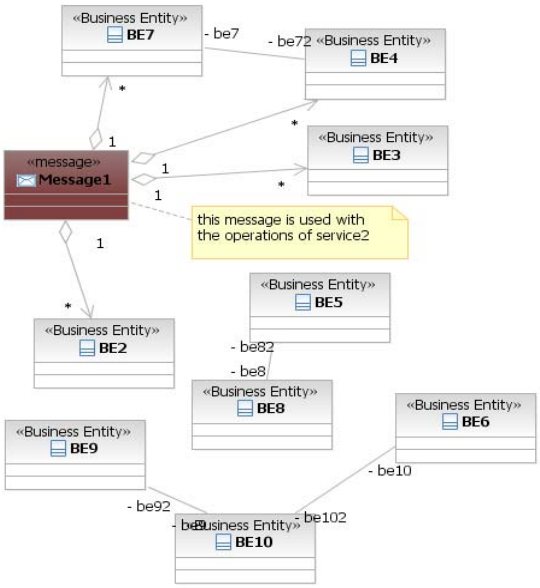

Figure 4. Message view

Figure 5. Collaboration view

Figure 6. Service view

6. Evaluation

In this section, the performance metrics for evaluating identified services are listed and the evaluation methodology also its process are introduced and finally the results are shown.

6.1. Performance metrics

With the gradual improvements of service notions and applications for constructing large scale enterprise software systems, various performance metrics have been put forward and widely applied in practice. Various performance metrics for evaluating component reusability have been defined and applied in various works such as managerial metrics (reusability) and technical metrics (coupling and cohesion) [6, 15, 16, 17, 18]. Performance metrics for evaluating identified services of service-oriented solutions are defined in Table 2. These items have been identified, customized, and extracted from research on component based software development, and also from our own experiences.

Table 2. Performance metrics

Metrics	**Meanings**	**Influence Factors**	**Optimization**
Reusability	The scope that S could be reused	Semantics commonality and variability of operations	Max
Reuse Efficiency	The contribution that S has to construct applications	Granularity	Max
Maintainability	Ease of reconfigure/modify S to fit proposed requirements	Cohesion, Coupling	Max
Granularity	The scale of S	Number of operations in S	Min
Business value	The contribution that S has to enterprise business domains	Granularity	Max
Cohesion	Semantic closeness between operations in S	Semantic affinity	Max
Coupling	Semantic closeness between operations in S and in other services	Semantic affinity to the other services	Min

The above metrics are somehow mutually exclusive and we cannot expect to satisfy them all. Therefore, service identification is a multi objective optimization problem. These metrics are used as validation parameters in our work.

6.2. Evaluation methodology framework

To evaluate our proposed process, we used the Sol [19] methodology framework, which pays explicit attention to all the important aspects of a development methodology. Sol's framework defines a set of essential factors that characterizes an information system development process and classifies them into a way of thinking, a way of modeling, a way of working, and a way of controlling, as is shown in Figure 7.

The way of thinking of the process provides an abstract description of the underlying concepts together with their interrelationships and properties. The way of modeling of the method structures the models, which can be used in the information system development. The way of working of the process organizes the way in which an information system is developed. It defines the possible tasks, including sub-tasks and ordering of tasks, to be performed as part of the development process. The way of controlling of the process deals with specific management aspects of the development process in terms of the management of resources, actor roles, intermediate and final results [19, 20].

Since we have been only concerned with proposing a proper analysis and design method for service-oriented solutions, here we ignore those aspects of the framework that relate to software implementation. We used the framework for capability classification of different aspects of our proposed process, and then used these capabilities to design questionnaires for conducting users' evaluation of our process, as well as for assessing the potential impacts and benefits our process produces compared to RUP methodology by experts.

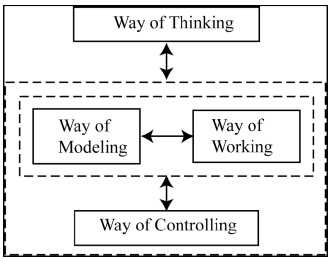

Figure 7. Methodology framework [20]

6.3. Evaluation process

Our proposed process was evaluated based on its usability, users' evaluation, and eminence over existing methods.

The usability of our process was tested through the development of a "supportive service system" in IKRF relief organization. Also a survey was used to gather users' opinions. Finally, the method criteria against traditional software development methods such as RUP were compared to highlight the potential benefits of practicing our process as the development process.

The system developed as the case study for evaluating our process was designed to provide supportive services in a relief enterprise. It had one main business process, 30 EBPs and 10 BEs. After adoption of the proposed process, 7 candidate services were identified. In the categorization of services, 3 services were considered as external services and 4 services were assigned to BPMS.

The users' evaluation of our process was collected through a survey. The aim of the survey was to obtain independent evaluation of the proposed process from potential future users of the process [20]. The survey participants were asked about the general properties of the process, and specific capabilities of service modeling processes structured in Sol's process framework. We obtained indications of positive evaluation of our proposed process from persons involved in the case study through statistical analysis of the answers which are shown in the following section.

In section 6.2 a framework for comparison of service modeling processes was described. We used this framework to indicate the eminence of our proposed process over existing methods. The experts favored our process to RUP in general and in specific capabilities for modeling processes.

6.4. Evaluation results

The users' evaluation of our process was collected through a survey. All the survey participants had rich experience in the fields of software systems development. The interview questions were formulated as propositions in an attempt to force the interviewees to make their opinions explicit.

The interviewees could answer a question based on a five-point scale ranging from (1) strongly disagree, (2) disagree, (3) neutral, 4 (agree) to (5) strongly agree. We used a statistical test to gain support for the directions of the outcomes. We had seven participants in the survey. In the survey tables, m denotes the mean, i.e. the average of the given grades, sd denotes the standard deviation, and np represents the number of positive responses, i.e. responses 4 or 5 [20].

Table 3. Experts' answer w.r.t general capabilities

Basic Method Capabilities	1	2	3	4	5	np	m	sd
Participation in this case study was valuable for my organization.	0	0	1	3	3	6	4.2	0.5
The method showed the importance of Identification and specification process in building a system.	0	0	0	1	6	7	4.8	0.1
The method showed the importance of architecture-driven development in building a system.	0	0	2	2	3	5	4.1	0.7
The method was shown to be simple.	0	0	1	1	5	6	4.5	0.5
The method was shown to be practical.	0	0	0	2	5	7	4.7	0.2
The method was shown to be flexible.	0	1	2	2	2	4	3.7	1.1
The method was shown to be systematic.	0	0	0	3	4	7	4.5	0.3

Table 4. Experts' answer w.r.t specific capabilities

Capabilities for Service analysis and design process	1	2	3	4	5	np	m	sd
The service concept was clearly defined.	0	0	1	2	4	6	4.4	0.5
The identified services were suitable w.r.t performance metrics.	0	0	0	1	6	7	4.8	0.1
The process had the capability to automate it.	0	0	0	0	7	7	5	0
Standard modeling notations was used in the process.	0	0	0	5	2	7	4.2	0.2
Business-driven identification of services was used.	0	0	0	1	6	7	4.8	0.1

Different viewpoints of SOA were used in the method.	0	1	1	4	1	5	3.7	0.8
Service model elements were completely specified.	0	0	0	2	5	7	4.7	0.2
The process was iterative and incremental.	0	0	2	3	2	5	4	0.6
Service assignment to solution architecture was described.	0	0	0	6	1	7	4.1	0.1
The roles of participants were specified.	0	0	3	2	2	4	3.8	0.7

7. Conclusion and future work

This paper proposed a new process for identifying and specifying appropriate service-oriented architectural elements at enterprise level from business models. This process defines a minimal set of coherent concepts, principles, model elements, guidelines, and techniques to construct service-oriented solutions. The paper also prescribed a stepwise process to guide the development of models. This process was applied to the development of a real world software, and its usability, users' evaluation and eminence over previous methods were pointed out. Extending the process to cover full construction of service-oriented solution is considered for further works. We intend to further strengthen and refine the process by developing a formal framework in order to support it. In addition, we plan to develop an integrated toolset to effectively support the process.

10. References

[1] O. Zimmermann, N. Schlimm, G. Waller, and M. Pestel, "Analysis and Design Techniques for Service-oriented Development and Integration", IBM Deutschland, www.perspectivesonwebservices.de/download/INF05ServiceModelingv11.pdf, 2005.

[2] R.T. Fielding, "Architectural Styles and the Design of Network based Software Architectures", Doctoral Thesis, University of California, Irvine, 2000.

[3] A. Arsanjani, "Service-Oriented Modeling and Architecture (SOMA)", IBM developer-Works, 2004.

[4] O. Zimmermann, P. Krogdahl, and C. Gee, "Elements of Service-Oriented Analysis and Design", http://www.-ibm.com/developerworks/webservices/library/wssoad1/index.html, 2004.

[5] S. Inaganti, and G.K. Behara, "Service Identification: BPM and SOA Handshake", Technical Report. Business Process Trends, www.bptrends.com, 2007.

[6] Zh. Zhang, R. Liu, and H. Yang, "Service Identification and packaging in Service-oriented Reengineering", www.cse.dmu.ac.uk/STRL/research/publications/pubs/2005/2005-8.pdf, 2005.

[7] O. Zimmermann, J. Koehler, and F. Leymann, "Architectural decision models as micro-methodology for Service-Oriented Analysis and Design", Proceedings of the SEMSOA Workshop 2007 on Software Engineering Methods for Service-Oriented Architecture, 46-60, 2007.

[8] B. Portier, "SOA terminology overview, Part 3: Analysis and design", http://www.ibm.com/developerworks/-webservices/library/ws-soa-term3/, 2007.

[9] S. Johnson, "Modeling service-oriented solutions", http://www.ibm.com/developerworks/rational/library/jul05/johnston, 2005.

[10] Ph. Teale, and R. Jarvis, "Business Patterns for Software Engineering Use", *The Architecture Journal*, http://msdn2.microsoft.com/enus/arcjournal/aa480036.aspx, 2004.

[11] Ambler S.W., J. Nalbone, and M.J. Vizdos, *The Enterprise Unified Process: Extending the Rational Unified Process*, ISBN: 0131914510, Prentice Hall, 2005.

[12] S. D. Mcsheffrey, "Integrating Business Process Models with UML System Models", Technical Report, Popkin Software, 2001.

[13] S. Johnston, and J. Smith, "RUP Plug-In for SOA V1.0", http://www.ibm.com/developerworks/rational/library/05/510_soaplug, 2005.

[14] S. Johnston, "UML 2.0 Profile for Software Services", http://www.ibm.com/developerworks/rational/library/05/419_soa, 2005.

[15] M.E. Fayad, "Accomplishing Software Stability", *J. Commun. ACM*, 45, 1 (2002), pp. 111–115.

[16] J.K. Lee, S.J. Jung, and S.D. Kim, "Component Identification Method with Coupling and Cohesion", Proceedings of the 8th Asia–Pacific Software Engineering (Macau, China, 2001). 79–86, 2001.

[17] H. Mili, A. Mili, S. Yacoub, and E. Addy, "Reuse-Based Software Engineering: Techniques, Organization, and Controls", New York, John Wiley and Sons Ltd, 2002.

[18] Szyperski, C., *Component software: Beyond Object–Oriented Programming*, Addison-Wesley, 1998.

[19] H.G. Sol, "Information Systems Development: A Problem Solving Approach", Proceedings of the International Symposium on System Development Methodologies, 1990.

[20] Z. Stojanović, "A Method for Component-Based and Service-Oriented Software Systems Engineering", Doctoral Thesis, Delft University of Technology, Netherlands, ISBN: 90-9019100-3, 2005.