ASSM: Toward an Automated Method for Service Specification

P. Jamshidi, S. Khoshnevis, R. Teimourzadegan, A. Nikravesh, A. Khoshkbarforoushha, F. Shams

Automated Software Engineering Research Group, Electrical and Computer Engineering Faculty, Shahid Beheshti University GC
{p_jamshidi, s_khoshnevis, r_teimourzadegan, a_nikravesh, a_khoshkbarforoushha, f_shams}@sbu.ac.ir

Abstract

One of the key activities needed to construct a quality service-oriented solution is specification of the architectural elements. Selection of an appropriate and proven method for specification of the elements consisted of services, flows, and components is thus fundamental to successful service-oriented system development in an enterprise. Existing methods for service specification ignore the automation capability while providing human-based prescriptive guidelines, which mostly are not applicable at enterprise scales. This paper proposes a novel method called ASSM (Automated Service Specification Method) that automatically specifies the architecturally significant elements of service-oriented systems. Therefore, ASSM effectively specifies the architectural elements of the service models. Model transformations such as ASSM automate the labor-intensive activities and lead the architect to focus on more important activities, which need human intelligence, and eventually enable efficient development of service-based solutions.

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]. It is widely accepted that traditional methods such as “object-oriented” and “component-based” methods especially in analysis and design aspects are inadequate to support the challenges for constructing such solutions [2], [3], [4]. Therefore, there is a need for a significant paradigm shift [4] toward enhanced service-based approach. First class constructs [4] of the new paradigm reside on such raised levels of abstraction. These constructs include services, service components, business process flows choreographing those services, and information requirements associated with them. Service-Oriented Architecture (SOA) is currently the leading architectural style for enterprise applications [5]. In the present paper, we are mainly concerned with the initial steps, that is, service modeling or service-oriented analysis and design [3] to construct service-based solutions, especially on specifying the architecturally significant elements of SOA.

There are many research work on service computing, technologies, products, and standards, but limited theoretical and practical experiences have been reported on service modeling [2]. However, these limited works suffer from serious shortcomings [3], [6]. The proposed methods do not have the automation capability while focusing on human-based and error-prone guidance and in the promising cases; they have introduced semi-automatic specification process by means of reusable design patterns. After identifying the problem, we have set the following objective for the research reported in the present paper:

“Proposal a method called ASSM (Automated Service Specification Method) to automatically specify the architecturally significant elements of service model work product. This model transformation gets the service model at analysis level and automatically produces the corresponding artifact at design level.”

The constructs of the service-based solution cannot be obtained baselessly, and some source information is needed to specify them. What we need is a mediated model to formalize and unify business models and raise the level of abstraction so that the architectural decisions can be made automatically [7]. This ties the deployed solution to its intended business value [6]. In this work, a mediated model in the form of matrix is used to facilitate automatic transformation of the business model to the solution model. To achieve this objective, a novel method is proposed to fully specify service model’s elements in terms of their structure and behavior. The method is the second step toward model-driven solution development of the service-oriented applications. This automation leads the architect to focus on more important activities, which need human intelligence, and eventually enables efficient development of service-based solutions.

The rest of this paper is organized as follows. Section 2 introduces the most related work. In section 3, an overview of preceding step, ASIM (Automated Service Identification Method), is presented. In section 4 and 5, details and applications of the proposed method, ASSM, are introduced. The ASSM is verified via a case study in section 6. Finally, the conclusion which leads to further research is described in sections 7.
2. Related Work

In this section, several works on service specification methods in the area of service-oriented analysis and design have been studied. In order to analyze the reviewed works, they are compared with the objectives of this research paper in three dimensions including the capability of automation, the capability of adopting performance metrics, and the capability of utilizing model-based principles.

By considering service-oriented solution lifecycle, we can divide the proposed service specification methods into two main categories. The first category consists of the methods, which are based on IBM service-oriented solution lifecycle, SOMA [8]. In these methods, service specification is a phase in which structural and behavioral aspects of services would be defined. Methods, which are proposed by Johnson [9], Portier [10], Amsden [6], Arsanjani et al. [4], Zhang et al. [11], and Strosnider et al. [12], are some instances of this category. Johnson’s [9] method does not cover the full specification of service model and its elements; moreover, it does not mention how to get them done automatically. Portier [10] uses model driven approaches for designing service model but does not propose a procedure for constructing such a model. Amsden [6] focuses on the description of service model’s elements, which should be specified in service specification phase, but does not consider the process of designing service model and specifying its sub models. Arsanjani et al. [4] explain service specification phase thoroughly, but says nothing on how to achieve the artifacts produced in this phase, therefore, by ignoring the technical metrics, the automation of the specification process would not be achieved. Zhang et al. [11] have introduced a modeling environment namely SOMA-ME which acts as a framework for model-driven design of SOA solutions using SOMA method. Although adopting the reusable patterns is a sign of design automation, in reality it needs the human intervention to customize the patterns. Strosnider et al. [12] have introduced an approach namely model-driven business transformation (MDBT), which uses a model-driven software synthesis technology to (semi-)automatically generate business service component from high-level business process model. Since their focus has been on entities and transition between them rather than activities, the behaviors, which are not associated with the entities, would be missed.

Second category consists of the methods, which are based on specific service-oriented solution lifecycle such as Erl [13], and Papazoglou [14]. They have introduced a thorough methodology for service-oriented solution development based on prescriptive guidance. Although there are differences between our point of view and theirs about service modeling, the adoption of technical metrics and also automation capabilities have been ignored in their work.

Capabilities of automation, adopting technical metrics, and utilizing model-based principles are the most important capabilities of the specification method, which is proposed in the present paper. Although service specification has remarkable role in the service-oriented modeling lifecycle, none of the work cited here have satisfied all the capabilities. Most of them are not fully automated, that is, they have to be done manually by architect. Some of them only pay attention to part of technical metrics and ignore others and this leads to incompleteness of component performance [15]. Moreover, a few of them take the principles of model driven development into account. The most qualified methods have been resulted from IBM initiatives (like SOMA-ME [13] framework in specification phase).

3. Overview of the ASIM method

According to the IBM SOMA [8] [4], service-oriented modeling lifecycle has three main phases:

- **Service identification**: This phase is about identifying the architecturally significant elements of the target solution. The output artifact of this phase is analysis-level service model.
- **Service specification**: This phase is about describing a service: what it offers, what it requests, how it is exposed. It also describes dependencies with other services, service composition, and service messages. The main model related to this phase is the design-level service model.
- **Service realization**: This phase is about providing a solution for a particular service in which we represent how a service is realized. The model related with this phase is the design model. This model has to be traced back to the service model, to determine if it covers the characteristics of corresponding service model.

In this regard, we proposed a method called ASIM (Automated Service Identification Method) [16] to automatically identify the candidate services as the main architectural element of service model work product with respect to specified technical metrics. In this method, service abstractions with acceptable technical metrics can be derived automatically from high-level business requirements and business process models. In this regard, a mediated model is needed through which the architectural decisions could be made automatically. We define the CRUD matrix [7] as desired abstraction level. By adopting this model, technical metrics could be utilized [7] in making architectural decision for constructing service based enterprise solutions. This ties the solution to its intended business value [6]. A cluster of the matrix would represent a business-aligned service. Therefore, a clustering method, which derives optimum mutually exclusive clusters, would be the solution of this problem. We formulate the service identification as a multi-objective optimization problem [7]. In order to solve the problem, we then propose a meta-heuristic optimization algorithm [16], namely ASIM method, which gets the matrix as an input and derive the enterprise level service set as an output. In this method, each candidate service would be identified with the right
granularity, while considering coupling, cohesion, business value, reuse efficiency, and maintainability.

4. Automated Service Specification Method

In order to improve productivity, enforce architectural integrity and improve the quality of the solution, the aim is to automate the service modeling activities. Defining and applying model transformation is a critical technique to automate the development process [17] [18]. Moreover, adopting technical metrics is a complementary tactic to have an efficient and quality solution [16]. Therefore, this goal can be achieved by introducing model transformations, which utilize technical metrics to automate the service-oriented modeling lifecycle.

ASSM [16] as a method for service identification had been developed as a first component of the automated lifecycle. ASSM method, which is the main focus of this paper, is intended to automate service specification. It is followed by ASRM (Automated Service Realization Method), which will be developed later to complete the lifecycle. ASSM method has four main tasks. These tasks are as follows:

4.1. Task 1: Launch Input

ASSM method gets two matrices as inputs. The main input is the analysis-level service model (i.e. clustered CRUD matrix), which is an artifact resulted from ASIM method. Rows of this matrix are Elementary Business Processes (EBPs) and its columns are Business Entities (BEs) [7].

Secondary input is the enterprise business process model in terms of a matrix through which the sequence of EBPs in business processes can be represented. Each row in this matrix represents a Business Process (BP) and its columns are EBPs. If order of one EBP depends on a condition, that condition would be located in an appropriate cell of the matrix. This matrix would be used during service interaction design to determine the behavioral elements of service model.

Using matrices instead of the business process models helps design a computational model to improve the formalization of the automation algorithms.

4.2. Task 2: Structure service architecture

In this task, we aim at modeling the structural aspect of the service architecture. This task takes the outputs of service identification (i.e. clustered CRUD matrix) and creates a structured service model in preparation for modeling service interactions. The constituent steps of this task are as follows:

Step 2-1: Model messages. Messages are modeled in this step resulting in a message diagram. A message diagram consists of "message" model elements. Input and output messages of a service, correspond with the BEs that the service operation (the corresponding EBP) deals with. Therefore, any BE that is created, read, updated or deleted by one service (i.e. is included in one cluster), is considered as a message.

Step 2-2: Model service specification. In this step, service specifications are modeled in terms of a service specification diagram. A service specification diagram mainly consists of "service specification" model elements. A service specification contains information about the interfaces that the service offers, including service operations and their input and output messages.

Service specifications are derived from the clusters of the CRUD matrix formed by ASIM. In the scope of a cluster, EBPs dealing with at least one BE, are interpreted as operations that the service contains. The operation input and output messages, can be extracted from the BEs that the EBP deals with, through one of the following semantic relationships.

- **Creation** of a BE by an EBP results in an output message by the service through the corresponding operation.
- **Reading** a BE by an EBP means an input message to the operation of the service.
- **Updating** a BE by an EBP includes an input message to the service, and the updated information as an output message.
- **Deletion** of a BE by an EBP requires an input message to the service.

Step 2-3: Model service providers. In this step, a service providers diagram is produced, which mainly consists of "service provider" model elements, "services they provide", "partitions", and "gateways".

Referring to ASIM method, a weight is assigned to every semantic relationship (i.e. the cells of the CRUD matrix), showing relationship intensity between EBPs and BEs. The intensity between two services is measured according to the semantic relationships ("R", "U" or "D"). The relationship intensity can be used to identify the quantitative value of interdependencies of services. The number of inter-service relationships and their intensity can help us group services logically in separate sets to be exposed by a single service provider.

\[ I_{i(a,b)} \] is defined as the intensity of relationship \( i \) between elements \( a \) and \( b \), where \( a \) and \( b \) can each be a service, a service provider, or a service partition.

Assuming that there are \( n \) relationships between elements \( a \) and \( b \), then total relationship intensity is defined as follows:

\[
T(a,b) = \sum_{i=1}^{n} I_{i}(a,b), T(a,a)=0 \quad (1)
\]

**Service Providers:** If \( S_i \) and \( S_j \) are two sample services and \( T(S_i, S_j) \geq \alpha \) then, \( S_i \) and \( S_j \) can be provided by a single service provider. The parameter \( \alpha \) is calculated as follows:

\[
\alpha = \frac{\sum_{i=1}^{n} \sum_{j=1}^{n} T(a_i, a_j)}{N} \quad (2)
\]

where \( a_i \) and \( a_j \) are services, \( n \) is the number of services, and \( N \) is the total number of possible relationships between services, calculated as:
We adopt a threshold accepting optimization algorithm \[16\] and use the T as objective function and \(\alpha\) as threshold to specify the service providers. Due to space limitation, we could not present the pseudo-code of the algorithm, but it could be found on our research group web site (http://aser.sbu.ac.ir).

**Service Partitions:** Our approach in specifying service partitions is based on the intensity of relationships between service providers like the technique adopted for specifying service providers. Interactions between services require service providers to interact. Thus, it makes sense and it is useful to classify service providers having a high degree of interaction, in a single partition as a logical system boundary to depict the entire enterprise-wide service portfolio \[19\].

**Service Gateways:** A service exposed by a service provider inside a partition can be promoted to a service gateway if the total relationship intensity between the service and other partitions is more than parameter \(\gamma\), or is the maximum intensity compared with other services inside the current partition. That is because such a service seems to be a composite service or participates in at least one service orchestration.

### 4.3. Task 3: Model Service Behavior

This sub-section discusses modeling the behavioral aspect of the service architecture in task 3. The task steps are as follows:

**Step 3-1: Model service collaboration.** In this step, service collaborations are modeled in terms of "service collaboration" elements and their structure diagrams. A structure diagram consists of "service provider" model elements that participate in the collaboration, the services they provide, and the service channels that form the relationships between services of the service providers.

Service channels are discovered via the semantic relationships "R, U or D" located out of the clusters areas in the CRUD matrix after service identification process has been completed. Since the semantic relationships "C" are located near the diagonal \[7\], the semantic relationships that will result in a service channel, resides on a cell located out of the clusters, and its column and row are pertain to two different clusters.

**Step 3-2: Model service interaction.** In order to realize service collaborations, service interactions are modeled in terms of a service interaction diagram. Service Interactions deal with the service compositions or orchestrations behavior, focusing on their relevant interaction orders. To specify a service interaction, we utilize the sequence matrix introduced earlier. The details of the procedure for specifying service interactions could be found on our research group web site.

### 4.4. Task 4: Create initial design model

Having specified the service model, ASSM focuses on realization aspects of each service. Service components that are required to realize specified services would be identified in this task, which is briefly introduced as follows. ASSM assigns one service component for service providers that none of their services is promoted \[10\]. For service providers that at least one of their services are promoted, services that are not promoted are considered to be realized on one service component and each service that is promoted is considered to be on a distinct service component.

### 5. ASSM applicability

The applications of the proposed method are discussed in the following categories:

- **Solution lifecycle automation**

  Existing service modeling methods are designed to create a solution for a specific and static set of requirements. These methods result in service-oriented systems that are slow in their response to dynamic conditions and changing requirements, expensive to maintain over extended periods of time, and prone to system failure \[14\]. As an alternative to this approach, this paper describes a model-driven approach to service modeling, which extends traditional methods and is well-suited to a systems development environment characterized by rapidly changing conditions and requirements. In addition, you can use the automation to improve the quality of the solution, as well as to support the overall governance process.

- **Software architect’s task simplification**

  Identification and discovery of architecturally significant elements of target solution, and their modeling and specification are labor-intensive tasks of the architects. Decision-making and validation of architectural constructs are critical tasks of this role. This research work put forward the automation of the first categories of task to reduced overall costs of ownership to achieve enhancement of the architect role.

- **Business-IT alignment**

  In our approach, the services have been identified and specified automatically from business models. These business models are the representatives of the business requirements. Therefore, this method aligns the enterprise service model as IT solution with the business model.

- **Reusability**

  As organizations seek to obtain greater business efficiency, there has been increased emphasis on reuse as a principle that applies throughout the software development life cycle \[18\]. For practical reasons, reuse is accompanied by automation practices for development methods in solution lifecycle.

### 6. Validation of the proposed method

This section presents verification and evaluation of our method based on the applicability of the method in a real case study and users’ evaluations of the method. The verification of the method was tested by using it during the development of building a sales system of a goods
6.1. Specified model elements

Having prepared the CRUD matrix, we replace each semantic relationship by the proper weight: 1 for "C", 0.75 for "U", 0.5 for "D" and 0.25 for "R". The matrix is then clustered and services are identified through ASIM method. Four services are identified utilizing ASIM, namely S1, S2, S3 and S4 (Figure 1). Due to space limitation, each model element is specified according to the order of ASSM steps. Thus, the detailed explanation has been ignored.

**Figure 1. Clustered CRUD matrix**

**Messages**: For each business entity (BE), there is a corresponding message, therefore, there are ten messages listed below:

Messages = \{Customer, Credit, Order, Invoice, Draft, Shipping Schedule, Inventory, Note Receivable, Voucher, Discount Info.\}

**Service Specifications**: Add customer (out: Customer, Credit), Place order (in: Customer; out: order), Check credit (in: Customer, Credit, Order) …

**Service providers**: Figure 1 is shown an adjacency matrix indicating total relationship intensity between services. This will be used to identify and specify service providers:

**Figure 2. Service providers**

According to (2),

\[ \alpha = \frac{0.75(1.25+0.25+1.25+0.5+0.75+0.5)}{6} = \frac{4.6}{6} = 0.667 \]

Pr1 = \{S1, S2, S4\}, Pr2 = \{S3\}

**Service Partitions**: According to figure 3, there are only two service providers and therefore they will both be placed in a single partition: Prt1 = \{Pr1, Pr2\}

**Gateways**: Since there is only one partition in the sample system, there is no need to define a gateway.

**Service collaboration**: A service collaboration is created for the only service partition prt1. The structure diagram for the service collaboration (Figure 4) contains the two service providers Pr1 and Pr2 with services shown as their ports.

**Figure 4. Service collaboration structure**

**Service Interactions**: The sequence diagram showing the service interaction is depicted in Figure 5.

**6.2 Experts’ evaluation of the Method**

The users’ evaluation of our process was collected through a survey. The survey covered the team who participated in the case study. The aim of the survey was to obtain an independent evaluation of the ASSM method from persons, who were potential future users of the method and its associated tool but not involved in its creation. All the survey participants had rich experience in the fields of service-oriented enterprise system development, and extensive knowledge of the latest developments in service-oriented analysis and design specially SOMA. They were looking for a more efficient and effective way of building such systems in the future using service oriented analysis and design methods. Before conducting the case study, the team had used the prescriptive guidelines, which they had found in [4], [19]. They had some difficulties in specifying the services. In the design and implementation or occasionally in deployment phases they face services with complicated inter-dependencies. Hence, they encounter many
difficulties for maintaining their systems. Furthermore, in
the case of huge domains, the productivity of service
modeling activities was not promising.

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. There were seven participants in the
survey. In the survey table 1, $m$ 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].

Our first aim was to demonstrate to the team the
benefits and roles of an automated method when building
enterprise solution. The second aim was to show how
adopting technical metrics in a quantitative manner offers
a number of advantages and benefits over a traditional
prescriptive method.

### 7. Conclusion and future work

We have provided a novel method called ASSM to
fully specify the elements of the service model that are
architecturally significant. The proposed method in terms
of a process consisted of several tasks each constitutes
some steps, automatically transforms analysis level of the
service model to the design level of it. This method by
utilizing technical metrics, which influence design
decisions and result in an efficient design of the target
solution, automates the service specification activity
based on principles of model driven development. The
method has been verified via a case study.

Strengthening the ASSM by adding additional
functionalities, which support model validation and
evolution, is considered as future work. Designers use the
fully specified service model to create and specify the
design model. The authors intend to further strengthen the
solution lifecycle by introducing appropriate automated
method. In addition, work is planned to develop an
integrated toolset to effectively implement these methods.
Moreover, we have initiated a project which aims to
introduce a framework consisted of cohesive assembly of
methods, techniques, tools, and content in order to support
all of the service modeling activities.

### 8. References


University of California, Irvine (2000).

IBM® developerWorks®, (2007)


[8] Brown, A.W., Delbaere, M., Eeles, R., Johnston, S., and
Weaver, R.: Realizing Service-oriented solutions with the IBM
Rational Software Development Platform, IBM System Journal,
Vol. 44, No. 4 (2005)

developerWorks® (2005)

and design, IBM® developerWorks® (2007)

Ponnalagu, K., Sindhagatta, R. R., Arsanjani, A., Bernardini, F.: SOMA-ME: A platform for the model-driven design of SOA

[12] Strosnider, J. K., Nandi, P., Kumaran, S., Ghosh, S.,
Arsanjani, A.: Model-driven synthesis of SOA solutions, IBM

(2007)

and development methodology, Int. J. Web Engineering and

Component Identification Methods and Related Techniques,

Transformation of Enterprise Business Model to Service Model,
IEEE Transactions on Service Computing (under review)


Rational SDP, IBM Redbook (2007)

[20] Z. Stojanović, "A Method for Component-Based and