Model Driven Approach to Service Oriented Enterprise Architecture

Sedigheh Khoshnevis†, Fereidoon Shams Aliee*, Pooyan Jamshidi*
†Shahr-e-Qods Branch of Islamic Azad University, *Shahid Beheshti University GC
{s_khoshnevis, f_shams, p_jamshidi}@sbu.ac.ir

Abstract

Enterprise Architecture (EA) has become an important means to acquire and maintain knowledge about the structure and behavior of the enterprises and to develop the required IT systems. Two main goals in architecture - high flexibility and low complexity - are addressed in a new architectural style called Service Oriented Architecture (SOA). In this paper, we discuss approaches to have SOA models in the enterprise architecture framework. Based on an analysis of these approaches, we propose an extension to the Zachman Framework called SOEAF, in order to provide the framework with the capability to include service-oriented artifacts representing services aspect of the enterprise. Then we will propose a Model Driven Approach (MDA) to Service Oriented Enterprise Architecture (SOEA), in which, MDA concepts, standards and tools help us achieve a semi-automated service-based analysis and design of the enterprise IT systems. The proposed approach can be very beneficial for increasing efficiency to reach enterprise-wide goals.

1. Introduction

Utilizing ICT is most crucial in today's enterprises, which are continuously exposed to many different changes, which, if are not applied under the control of a well-organized architecture, they will lead to more complexity and inefficient software systems. This fact shows the importance of applying architecture and more precisely enterprise architecture.

One of the architectural styles becoming widespread today is Service Oriented Architecture or shortly SOA. This style views each system or business as a collection of service providers, each providing one or more services. This view is very similar to what is practically achieved in businesses and organizations, remarking the fact that services offered by an enterprise to its customers, parties, etc. is one of its aspects [1] and must be covered in the enterprise architecture.

Applying SOA on EA has benefits [2], most of which are resulted from a proper separation of concerns in the enterprise that leads to definition of smaller-sized IT projects and thus to reduction of risks and increasing efficiency. Independence of services from each other and from technology helps having higher degrees of reusability and flexibility in dealing with changes, as well as less costs by using reusable assessments and resources [2].

The similarity between the service-orientation and what is practically done in enterprises on one hand and the advantages that utilizing SOA can bring to an enterprise on the other hand, led us to the question that "How can we have SOA models in an enterprise architecture?"; in other words "How is it possible to reach service oriented enterprise architecture?"

In order to find a proper answer to the question asked above, we focused on the Zachman Framework (ZF) [3],[4] because it is widely accepted besides other advantages like well-defined perspectives and abstractions in addition to the extensive usage of the framework in practice. It is not explicitly determined how to have SOA artifacts in the ZF or how they can be organized within the framework. The mentioned similarity between service orientation and actual business practices offers service oriented analysis of the enterprise, which has not yet been utilized in analysis and design methods. Therefore, we decided to extend the ZF in order to add the capability of including service-oriented models utilizing enterprise-wide business and software service-orientation concepts.

Another issue to consider was the method, language and tool required to model the desired artifacts practically. For this purpose, we studied applying Model Driven Approach (MDA) on SOEA. Therefore, in this paper we aim at "proposing a model driven approach to reach service-oriented enterprise architecture in which service-oriented artifacts are included, as long as other artifacts representing different facets of the enterprise".

The rest of the paper is organized as follows. In section 2 and 3, basic concepts and the related work are reviewed. Next, we discuss approaches to service oriented enterprise architecture. A detailed description of the framework proposed for SOEA is provided in section 4. In section 5, the proposed model driven approach to SOEA is discussed. We provide a case study in section 6 and finally, there is a comparison between the proposed approach and other approaches, in addition to the conclusion, and suggestions for future works.

2. Main Concepts

In this section, we present a brief introduction to the main concepts we use in this paper in order to clarify what we mean by each of them.
2.1. Enterprise Architecture

An enterprise consists of people, information, and technologies. It performs business functions and has a defined organizational structure. It is able to respond to events, both internal and external. And finally, it has a purpose for its activities, which includes providing services and products to its customers [1]. John A. Zachman defines enterprise architecture as the set of primitive, descriptive artifacts that constitute the knowledge infrastructure of the enterprise [5].

2.2. Zachman Framework

Zachman enterprise architecture framework or shortly Zachman Framework [3] is a classic enterprise architecture framework that provides a 6x6 matrix of architecture abstractions and perspectives with products. The framework consists of six rows representing perspectives of different stakeholders including planner (contextual model), owner (conceptual model), designer (logical model), builder (physical model), subcontractor (as built), and finally functioning enterprise. It also consists of six columns describing different abstractions that define each perspective, including data, function, network, people, time, and motivation.

The ZF is not only a matrix of rows and columns, but it follows seven rules that you can find in [3] and [4]. You can also find more details on Zachman framework at the official site of J. A. Zachman at [6].

2.3. Service Oriented Architecture

According to McGovern et al. [7], Service Oriented Architecture is an architectural style that formally separates services, which are the functionality that a system can provide, from service consumers, which are systems that need that functionality. This separation is accomplished by a mechanism known as a service contract, coupled with a mechanism for providers to publish contracts and consumers to locate the contracts that provide the service they desire. Rather than coupling the consumer with the service in terms of technical aspects of invoking the service, SOA separates the contract from the component or implementation of that contract. This separation produces an architecture in which the coupling between the consumer of the service and the modules that produce the work is extremely loose and easily reconfigured.

2.4. Service Oriented Enterprise Architecture

We define service oriented enterprise architecture (SOEA) as an enterprise architectural style, which includes a service-oriented abstraction of the enterprise and thus applying it results in producing service oriented artifacts (such as service models).

2.5. Model Driven Architecture

Model Driven Architecture (MDA) is an architectural framework introduced by the OMG, which increases the power of models in systems development. [8].

MDA specifies three viewpoints on a system, a computation independent, a platform independent and a platform specific viewpoint. Viewing a system from a viewpoint results in a corresponding model, therefore there are computation independent, platform independent and platform specific models (CIM, PIM, PSM) respectively [8]. Figure 1 demonstrates how these models are created in an MDA process.

OMG has adopted a number of technologies, which together enable the Model Driven Architecture including Meta Object Facility (MOF), OCL, UML, and UML profiles. Refer to [9] for more details.

3. Related Work

SOA in EA has been discussed from a couple of different viewpoints. In [10], Arsanjani et al. propose a method for developing service-oriented solutions called SOMA. This approach is generally proposed for any service-based solution and it does not leverage an enterprise architecture framework. Maarten van Steen et al. propose a method for SOEA, based on a framework (matrix) made up of three columns as data, function and network and three rows as business, service and technology [11]. The former paper has MDA-support [12], while the latter does not.

Another category of the research work is about applying MDA to popular enterprise architecture frameworks and especially the Zachman framework as the de-facto standard for EA [13],[14]. However, none of them discuss how it is possible to have SOA-based solution artifacts within the framework.

Since we aimed at utilizing MDA in our approach we studied MDA-based SOA modeling. The researches [15],[16],[12] included discussions about model-based SOA solutions, but none of them were in the context of an enterprise architecture, and especially by applying an EA framework.

4. Approaches to SOEA

Enterprise architectures require a framework [3] for classifying and organizing artifacts resulted from studying the enterprise from different perspectives and views. Therefore, it is necessary to have a framework for SOEA as well. In the following, we present four approaches to service orientation in Zachman Framework and discuss the applicability of each [17].

There are a few issues to notice: Service artifacts are present in every perspective. Thus, it is desirable that the solution addresses all five key roles of the ZF (planner, owner, designer, builder and sub-contractor, according to rows one through five of the ZF). We have
tried to leave the base framework as unchanged as possible and to keep the framework rules valid in order to have the most reusability of current enterprise architecture artifacts.

4.1. First approach: service orientation in the "Function" column

Since the concept of service is more relevant with enterprise functions and processes, the first idea is that, it seems better to find a way to have service-oriented artifacts in the "function" column of Zachman framework. In this case, there will be some issues: First, we will not have the pure process models of the column and instead, there will be models of services, which are completely different from process models. The services might exist in a different level of granularity from processes. In addition, we aim to have an approach that leaves the base framework –and its artifacts- unchanged.

4.2. Second approach: service orientation in every cell

Another idea is to produce artifacts with a service-oriented view to each cell within the framework. For example to produce the conceptual data model, consider them as data services or for instance when producing the business process model try to view them as business services. Thus, each cell comprises a part of the whole set of service-oriented artifacts in a row.

This approach is not applicable either. The third, fourth and fifth rules of ZF state that each row represents a particular perspective, each column represents a particular abstraction of the enterprise, and thus each cell represents a unique abstraction from a unique perspective. Therefore, we cannot represent a service model (containing functionality) in cells of columns (like "data" column) that are not intended to represent functionality.

The idea is also in contradiction with some principles of service orientation (which can be found in [7] and [18]) such as abstraction of a business domain by services. It is obvious that in order to abstract a domain, a service may not include the data only. It also may not be comprised of processes regardless of the data, events, etc. Therefore, we cannot place service-oriented artifacts in cells concerned with just one abstraction of enterprise information.

4.3. Third approach: service row

The third idea is considering one or more rows of ZF, having more relevance with the concept of service orientation, as service rows (The "designer" row can be a candidate).

Applying this approach leads to excluding service-oriented artifacts from some of the framework perspectives, considering them irrelevant with the notion of services. This is in contradiction with the fact of closeness of SOA concepts with what goes on in enterprises in reality, and that all roles mentioned in the ZF can have their own service-oriented viewpoints.

In addition, the problem we mentioned for the previous idea still exists, because each cell in the service row(s) addresses a particular abstraction and thus cannot cover the concept of service, which aims to abstract a domain with all its aspects.

4.4. Fourth approach: service column

As it is obvious from the three previous approaches, for each perspective in the framework, a separate cell is required to specify service oriented artifacts related to that particular perspective. Such a cell cannot be chosen from any of the current columns of the framework because of the issues discussed in second and third approaches.

As the fourth idea, we suggest extending the ZF by adding a new column, named "service column", in which, each cell is a service-oriented viewpoint of the respective row.

In the next section, we define what artifacts must be produced in each cell of the new service column.

5. SOEAF: An Extension to the ZF

Figure 2. shows the extended framework already introduced. For ease of reference, let us name it "SOEAF" as an abbreviation for Service Oriented Enterprise Architecture Framework.

Generally, artifacts in the service column consist of service contracts, service providers and their structure, messages and message exchange patterns, and the relationship of services.

In this new column, in the contextual perspective, a list of important main business services provided by the enterprise is prepared giving a big picture of what the enterprise generally offers or achieves. In the conceptual perspective, business services are modeled. In the logical perspective, the business services identified in the conceptual perspective are mapped to software system services. Identifying software services is easier having business service models at hand, however, making proper changes in software services models is inevitable moving from business scope to software systems scope. In the physical perspective, the proper platforms and standards for implementing service model elements are chosen regarding the technological issues. In the as-built perspective, components, modules and service contracts are implemented in detail; and finally, the functioning enterprise includes the deployed enterprise services.

As you can see, seven aspects of the enterprise are covered in each row of SOEAF. In fact in each row we determine what process is done on what data by whom, why, and in what time and place and finally what services are provided as a result.

Artifacts may be produced utilizing any proper process or methodology and using any appropriate tool(s); therefore, SOEAF is not limited to a particular methodology or specific tools.
6. Model driven SOEA

Although SOEAF is not limited to a method or tool, in order to make it applicable in practice, we need a method to prepare the artifacts. In this section, we discuss model driven approach to service oriented enterprise architecture based on SOEAF.

Ostadzadeh et al. have proposed a method for consistent modeling of the ZF utilizing MDA [14]. In their article, they have offered adopted MDA standards for each cell, except for the first row, that is a list of important things to the enterprise and not a model, for which they suggest natural language as the best option. They have also excluded the sixth row from the problem space because it is actual deployed or running elements of the enterprise; it is not a perspective, but "the real world". They have suggested Computation-Independent Model (CIM) for the second row, Platform-Independent Model (PIM) for the third row and Platform-Specific Model (PSM) for the fourth and fifth rows. Figure 3 depicts the overall solution (for a precise definition of CIM, PIM and PSM see [8]).

6.1. Problem Scope

The problem scope does not include the first and sixth rows because of the reasons already mentioned. We suggest employing CIM for the second row, PIM for the third row, PSM for the fourth row and code, which is also considered as model [19], for the fifth row. Figure 4. shows the compatibility of these models, and their related viewpoints, with SOEAF perspectives.

6.2. Modeling language

Since there are not any adopted OMG standards for modeling SOA in MDA at the time the research was being done, we decided to choose or create a modeling language. There are two mechanisms for defining modeling languages (or metamodels) in MDA, one is Meta Object Facility (MOF) and the other is UML profiles. MOF is an OMG standard, which enables metadata management and modeling language definition [8] (UML is defined within MOF). A UML profile is a set of extensions and constraints that tailors UML to particular use [8]. There is a detailed comparison of the two mechanisms in [20]. The main advantage of UML profiles over MOF metamodels is availability of tools. In addition, UML profiles extension and constraint mechanisms meet the overall needs for our required language. Thus, we decided to utilize UML profiles for this purpose. The acceptable UML profile for modeling SOA must be able to represent all SOA concepts, simple, easy to use and understand, tool-supported, and preferably UML 2.0-compliant "UML 2.0 profile for software services" [12] provided by IBM meets these criteria. The profile is implemented in the IBM Rational Software Architect (RSA) and is simple and easy to understand. It is designed to be transformable to different Web Services artifacts such as WSDL, XSD, EJB, etc. IBM's UML 2.0 profile for software services fits into the category of PSM languages, considering WSDL, XSD, etc. as pieces of target source code, specific for Web Services implementation platforms. The source codes mentioned seem to meet the needs of the fifth row of the service column. In addition, the profile specifies service channels, channel binding, service collaboration binding, allowed binding for service providers and encoding types for messages and message attachments. Thus, it seems thorough for modeling the platform-specific model needed in fourth row of the service column. However, the platform-specific specifications above must not be included in a platform independent model. Therefore, it seems that a subset of the profile, which excludes the platform-specific specifications, is suitable for addressing PIM required in the third row of the service column. Figure 5. depicts the profile proposed and utilized for the SOA PIM showing its
stereotypes and their relationships. See [21] for more details.

Figure 5. Model of SOA PIM UML 2.0 Profile.

In order to model the second row of the service column we require a UML profile with business-specific stereotypes. This is because the scope of the second row is very close to the business domain and that the related viewpoint is "computation independent viewpoint". This led us to design a UML profile for the CIM in the service column, which, despite of similarity to the PIM profile, focuses on business service model elements due to the fact that the business domain is computation-independent and this makes it different from technology domain which is computation-dependent. Figure 6. depicts the conceptual model for UML 2.0 profile for SOA CIM showing its stereotypes and their relationships (Word "Biz" has been used as a short form of "Business"). See [21] for more details.

Figure 6. Model of SOA CIM UML 2.0 Profile.

6.3. Transformation rules

Because of the similarity between the three profiles utilized, extraction of the required transformation rules was quite easy. The transformation rules for transforming PSM to code were available in IBM RSA; therefore, we did not redesign them. In the following, we provide rules for transforming SOA CIM to SOA PIM and SOA PIM to SOA PSM, using stereotypes as marks. For more details on model mappings, marking and transformations see [22].

General rules for transforming a SOA CIM to a SOA PIM are as follows:

1. If the source UML 2.0 model has "SOACIM" as an applied profile, then create a target UML 2.0 model with "SOAPIM" profile applied. For each diagram (and sub-diagram) of the source model, create a similar diagram (or sub-diagram) in the target model. For each element and its parts in source diagram, do:

   1.1. Based on Table 1, transform each source element having a stereotype in column A, to a target element of the same metaclass, and set it the corresponding stereotype in column B.

<table>
<thead>
<tr>
<th>A (source stereotype)</th>
<th>B (target stereotype)</th>
</tr>
</thead>
<tbody>
<tr>
<td>BizPartition</td>
<td>ServicePartition</td>
</tr>
<tr>
<td>BizService</td>
<td>Service</td>
</tr>
<tr>
<td>BizServiceProvider</td>
<td>ServiceProvider</td>
</tr>
<tr>
<td>BizServiceConsumer</td>
<td>ServiceConsumer</td>
</tr>
<tr>
<td>BizServiceSpec</td>
<td>ServiceSpecification</td>
</tr>
<tr>
<td>BizServiceCollaboration</td>
<td>ServiceCollaboration</td>
</tr>
<tr>
<td>BizMessage</td>
<td>Message</td>
</tr>
</tbody>
</table>

   Table 1. Mapping CIM Stereotypes to PIM Stereotypes

   1.2. In addition, the following modifications are needed in any transformation:

   1.2.1. Any constraints attached to an element of the source model listed below, shall be transformed to a constraint in the target model, attached to the corresponding target element, and shall be stereotyped "PolicyInfo".
   a. Any operation owned by an interface stereotyped by "BizServiceSpec".
   b. Any interface stereotyped by "BizServiceSpec".
   c. Any class stereotyped by "BizMessage".

   1.2.2. the scope of all source operations, owned by interfaces with stereotype "BizServiceSpec", shall be "public" in the target interface.

   1.2.3. The scope of all source attributes, owned by classes with stereotype "BizMessage", shall be "public" in the target class.

   1.3. Migrate all other source elements to target elements (applying no changes).

   Although identifying software services is easier having business service models at hand, however, making proper changes is inevitable moving from business scope to software systems scope: Pure-manual business services may be fully omitted from the software service model. There may be needs for changes in operations each service offers. New system services may be required, which may not have a peer among the business services. There may be needs for modification of service providers, messages, message exchange patterns and relationships as well.

   General rules for transforming a SOA PIM to a SOA PSM are as follows:

   1. If the source UML 2.0 model has "SOAPIM" as an applied profile, then create a target UML 2.0 model with "Software Services" profile applied. Get the default binding, default encoding, and default allowed binding as transformation parameters from user. For
each diagram (and sub-diagram) of the source model, create a similar diagram (or sub-diagram) in the target model. For each element and its parts in a source diagram, do:

1.1. Any connector, which at least one of its ends is stereotyped "Service", shall be transformed to a connector stereotyped "ServiceChannel" between the corresponding target elements and its "binding" attribute shall be assigned the default binding parameter value.

1.2. All binding, encoding and allowed binding attributes shall be set to the corresponding default parameter values.

1.3. Migrate all other source elements to target elements (applying no changes).

7. Case Study

The solutions proposed in fields like software engineering and software architecture are not usually provable using formal methods. Instead, they are often analyzed, studied and validated via case studies.

The steps used in our case study were as follows:

• Choosing an enterprise, understanding it, and determining the scope of the study.

• Choosing a tool, supporting MDA, UML profile definition, UML modeling and transformations.

• Preparing artifacts of the first row of the service column and modeling the second row of the service column (CIM).

• Generating models of the third through fifth rows of the service column (PIM, PSM and code) using the transformation rules, and making required modifications.

The chosen enterprise was a goods-distribution company. The enterprise was a very good sample of an enterprise for which, service-oriented architecture could be very beneficial, and the concepts could be shown very clearly. There were several services, both manual and automatic, provided to users, using different message exchange patterns. There were good samples for orchestration and composition of services, and service relationships as well. The tool chosen was IBM RSA version 7.0 that provides the UML 2.0 profile for software services as well as a built-in transformation of UML to WSDL, XSD, EJB, etc. It also supports defining transformations as well as support for new languages via defining UML profiles. We generated PIM, PSM and the code, which made the process very easy.

We planned to answer the following key questions in the case study:

• Does the approach provide capability for the SOA architects to describe their viewpoints?

• Does it cover the main enterprise architecture viewpoints (contextual, conceptual, logical, physical, code)?

• Is it compliant with up-to-date standards and modern tools?

• Is it easy to use and understand?

• Is it capable of utilizing modeling language definition, transformation rules, etc. with proper modern MDA tools?

The study showed the importance of the role service-related concepts could play in an enterprise. In addition, it showed that utilizing service-based analysis as a basis for SOA design can be very effective in preparing appropriate models and a better understanding of the enterprise domains to be captured, along with providing a common terminology between the domain practitioners, analysts, designers, architects, etc. In addition, any changes in the business domain models (CIM) can be easily reflected in technology domain models (PIM and PSM) and the code, representing a proper alignment between the two areas of concern.

Service column models in each row are completely integrated with other cells of the row. Different association matrices can be made between SOEAF columns, e.g. data-service, process-service, people-service association matrices and so on.

Almost every main role in the enterprise can take advantage from benefits that the proposed approach brings about. The ZF considers five roles as main, most important enterprise roles. "Planner" is concerned with the context of the enterprise, "owner" has a conceptual view of the current business (business expert), "designer" is mostly concerned with the logical issues in systems architecture and design, "builder" has a physical view to those issues, and "subcontractor" deals with the code and is responsible for the final product.

Planners will not see a great change in the way they used to deal with the framework, except for the new notion of services they will use to define the main general boundaries of the whole enterprise. The planners' familiarity to the service notion helps declaring what services the enterprise generally offers and what services it requires to reach its business goals.

Owners feel much easier to express the requirements and the current enterprise business state since the terminology and the concept of service-orientation is very close to what they deal with in their real day-to-day business job. Therefore, a better interaction will exist between the owners -who are mainly the business experts- and the IT specialists and software engineers. As we will discuss, the approach leads to a better alignment between the business and the IT environments.

Designers will gain a great benefit from the high degree of reusability from requirements stated by the owners in the upper view. It is because there is only a need to revision and making proper changes rather than starting to model the design artifacts from scratch. In the scope of our case study, about 90% business services and their specifications (in the CIM) appeared in the PIM with more than 92% reuse; in other words, 62% of the system services in PIM were totally or partially derived from the CIM, which is a high ratio.
However, service providers undergo a great change, since some business service providers are humans or legacy systems that are to be replaced with the modern product. Among service providers in the PIM, there are some of the legacy systems, that are to be reused, some of the PIM service providers are just a renamed version of the business service providers, since only a person is replaced by a software component serving as a service provider, and some "system service" providers are added to the collection. Business service providers providing fully manual services have no peers in the PIM.

The builder will take advantage from the generated physical model, only concentrating on platform issues and the final preparation for the final code generation task.

The subcontractor may only test, and verify the generated code in order to fit to the requirements stated by the business owners.

SOA on one hand and MDA on the other, result in reaching up-to-date enterprise architectures and a high degree of alignment between business and IT. Service-orientation concepts and principles are compatible with the nature of business rules and environments, and the independence of services from one another and from the implementation platforms makes systems more flexible against changes that are usually emerged from the business area. MDA provides automation in preparing different artifacts, therefore, when a change in the business occurs, the largest changes must be made in the CIM, and then, PIM, PSM and the code are semi-automatically generated subsequently.

The case study includes many models and diagrams. Figures 7, 8 and 9 depict parts of models prepared or generated in rows two through four of SOEAF service column. You can find detailed architecture design for the case study in addition to the generated code in [22].

![Figure 7. Part of the CIM for the second row of SOEAF (business services)](image)

![Figure 8. Part of the PIM for the third row of SOEAF (service specifications)](image)

8. Proposed approach vs. other approaches

In order to compare the proposed approach with other approaches, an evaluation was done based on the following criteria:

A- Support for SOA (how to model and organize models, exactly defining SOA artifacts, defining the modeling language for modeling SOA)

B- Provide capability for the SOA architects to describe (model) their viewpoints (service contracts, messaging, message structures, complying with implementation standards, orchestration and choreography, capability of representing structure and behavior of service provider components)

C- Utilizing MDA (MDA viewpoints clearly described within the framework, transformation rules specified, supporting MDA standards)

D- Covering viewpoints of the framework used (contextual, conceptual, logical, physical, as built, functioning)

E- Being up-to-date (compliance with up-to-date standards and utilizing modern tools)

F- Ease of use and understanding (less complexity and redundancy)

G- Tool support (capability of defining modeling language, defining transformation rules, testability, and verifiability)

We have decomposed each of the main criteria (A to G) into more detailed criteria, mentioned above in front of each criterion in parentheses, because the main criteria are very general in nature. The scoring is as follows: if the approach does not satisfy each detailed criterion then a zero score was considered and otherwise score (1) was dedicated to the detailed criterion for the method. Then we recorded the total score for each main criterion in a table. Table 2 shows the comparison results.

<table>
<thead>
<tr>
<th>Method Criteria</th>
<th>[14]</th>
<th>[22]</th>
<th>[11]</th>
<th>[23]</th>
<th>[13]</th>
<th>[24]</th>
<th>Ours</th>
</tr>
</thead>
<tbody>
<tr>
<td>A(of 3)</td>
<td>0</td>
<td>0</td>
<td>3</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>B(of 6)</td>
<td>0</td>
<td>0</td>
<td>6</td>
<td>0</td>
<td>0</td>
<td>6</td>
<td>6</td>
</tr>
<tr>
<td>C(of 3)</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>0</td>
<td>3</td>
</tr>
<tr>
<td>D(of 3)</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>E(of 2)</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>2</td>
</tr>
</tbody>
</table>

Figure 9. Part of the PSM for the fourth row of SOEAF (services and their relationships)
Table 2. The Proposed Approach Rank

<table>
<thead>
<tr>
<th>F(of 2)</th>
<th>1</th>
<th>1</th>
<th>1</th>
<th>0</th>
<th>0</th>
<th>0</th>
<th>2</th>
</tr>
</thead>
<tbody>
<tr>
<td>G(of 4)</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Total Score</td>
<td>5</td>
<td>6</td>
<td>12</td>
<td>6</td>
<td>4</td>
<td>11</td>
<td>18</td>
</tr>
</tbody>
</table>

9. Conclusion and future work

In this paper, a model driven approach for service-oriented enterprise architecture is proposed based on SOEAF. The project showed how the closeness of SOA concepts and business concepts can help us move from business domain to technology domain; and especially by adoption of MDA, this can become much easier and more formal.

The contribution of this paper is to propose an approach to achieve SOEA based on SOEAF. In section 7, we discussed criteria and the proposed approach satisfying them to some extent. In addition, the approach has the following characteristics:

- Utilizing service-based analysis as a basis for service-oriented design.
- Providing a common terminology and ease of understanding for service-oriented models.
- Integration of service column cell artifact with artifacts in the relating row and column.
- Ease of generating artifacts through defining semi-automatic model transformation rules.
- Reaching a good level of business-technology alignment.

Future studies may include the following:

- Studying model driven SOA in other most widely-used enterprise architecture frameworks.
- Designing and developing two-way mappings and transformations between CIM, PIM, PSM and code in model driven development of service oriented models and artifacts, and finally,
- Proposing an appropriate methodology for service oriented enterprise architecture.

10. References