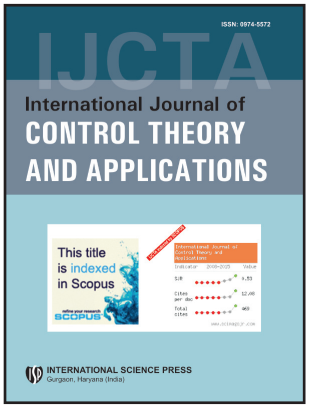


International Journal of Control Theory and Applications

ISSN : 0974-5572

© International Science Press

Volume 9 • Number 44 • 2016

Service-Oriented Business Integration Model

Edwin Puertas del Castillo[a], Nohora Mercado-Caruso[b], Jairo R. Coronado-Hernandez[c], Juan Carlos Martinez[d] and Katherine Salas-Navarro[e]

[a,d]Universidad Tecnológica de Bolívar, Cartagena de Indias, Colombia. Email: [a]epuerta@unitecnologica.edu.co; [d]jcmartinezs@unitecnologica.edu.co
[b,c,e]Universidad de la Costa, Barranquilla, Colombia. Email: [b]nmercado1@cuc.edu.co; [c]jcoronad18@cuc.co; [e]ksalas2@cuc.edu.co

Abstract: This paper presents a service-oriented business integration model that will help enterprises to keep in business by allowing them to use their legacy system at the time they add new developments. Today, there are options that fit very well on big enterprises that can afford to migrate from their legacy software and adopt my standards. However, we are targeting those cases that changing is not an option. We based our integration pattern on the identification of all the components that play a significant role in the enterprise business model. By following the steps we present, it is possible that an enterprise grows under a service-oriented architecture. We have successfully applied our model. We present a study case that details all the proposed methodology.

Keywords: Service-oriented, business integration, service-oriented architecture (SOA).

1. INTRODUCTION

This paper describes a service-oriented business integration model for enterprises. The enterprises are facing problems due to the dynamic market change, the rapid evolution of computer systems and computing paradigms when they think about enterprise integration. For these reasons, many organizations think about adopt or not adopt new architectures or patterns as part of their business. On the other hand, there is a need to support technological growth for companies, as mechanisms for enhancing productivity and competitiveness. This paper proposes a service-oriented business integration for enterprises. Businesses today are faced with legacy systems by which their services tend to become references to the point that the company is adapting the software and not the software to the needs of the company. This problem becomes more noticeable when the modern trends or by the same cycle of software applications out of the market and is no longer supported, and the company is forced to update its database or operating system, or both. This puts at risk the business of the company, which is forced to find ways to integrate their software to the stage to the new computational paradigms. Many times, it would be best to start from scratch. However, the reality is that companies must find ways to integrate their legacy systems reducing the trauma.

Many authors propose that integration architectures should be built systematically in several layers [1] [2]. The idea behind this is to divide the problem into smaller problems and solve each sub problem one at the time. Therefore, integration today can be seen as a structured multilayer model. Usually start building the integration architecture in the lowest layer and climb slowly. The most important types of integration are level data, the level of applications, business processes, presentation, and business to business. Enterprises can reduce the cost of developing its information systems. They can reuse services, reduce redundant functions, and simplify its infrastructure, minimizing the cost for maintenance of the code. Enterprise integration does not merely aim to connect systems processes but also provide flexible and fault tolerant processes [3].

2. RELATED WORK

This section briefly reviews proposals that can apply SOA to obtain best tangible benefits for enterprise. Although there are many solutions for resolved the problem of integration in enterprise, our proposal is distinguished by include the new computational service-oriented approach as essential element for an enterprise integration. As many authors [6]–[10] believe, we think that service-oriented enterprise integration can produce tangible benefits for enterprise. Authors of [11] show a SOA transformation approach from the inside to the outside of large enterprises. They also describe the same evolutionary process led to small and medium enterprises. The difference is that the processes taking place in diverse scenarios. However, by their nature, enterprise often do not have a large IT department or budget. Many of these companies have only a few users with information and communications technology skills. They may not be able to adopt their own approach. However, if the correct services are available and the integration of these services is comparatively effectively, enterprise are more likely to pick those services and integrate them in their companies, as most of them are very flexible in terms of their business processes and the way, they operate.

On the other hand, the authors of [3] proposed a business integration framework based on SOA. The tool provides the following functions: a user framework for configuration files, expansion of the various components basic, a mechanism for processing and exception handling, a distributed support certain load capacity, single-sing on (SSO), cache storage components support, and security administration systems. Similarity, a SOA-based framework that poses the authors [12], combines traditional technology as message-based, middle-ware, and web-based services to achieve a business integration framework. One of the important aspects of progress in computer networking and information processing is the implementation of IT security, because in these networks, information is transferred between various nodes, causing safety hazards in the network, the server and client. These risks have been minimized by implementing IT security [13]. The research presented focus on solving the business integration problem partially. They do not include all elements of integration (operational, technological, and conceptual). They only focus on the operation of integration and not in the other levels of integration. Other examples of enterprise integration where the drawbacks in adopting new computational approaches are shown in [6]–[12]. Our proposal is innovative, because the integration focuses on three levels (conceptual, technological and operational), and includes the principles of Service Oriented Computing as a central focus of Integration.

3. METHODOLOGY

Our model proposes integration through conceptual, technological, and operational levels. These levels will facilitate the service-oriented business integration. Then, the goal of each level and element that comprise it are described below. Our pattern describes three elements of integration (conceptual, technological and operational) that any enterprise must have to adopt to meet the principles of Service-Oriented Computing (SOC) [4], can see Figure 1.

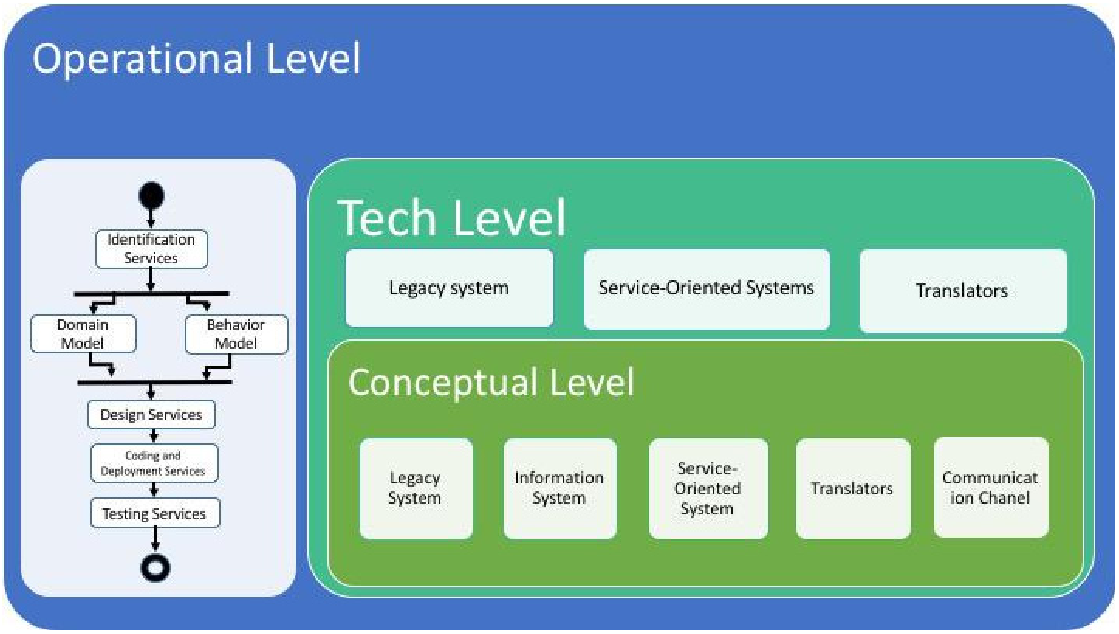
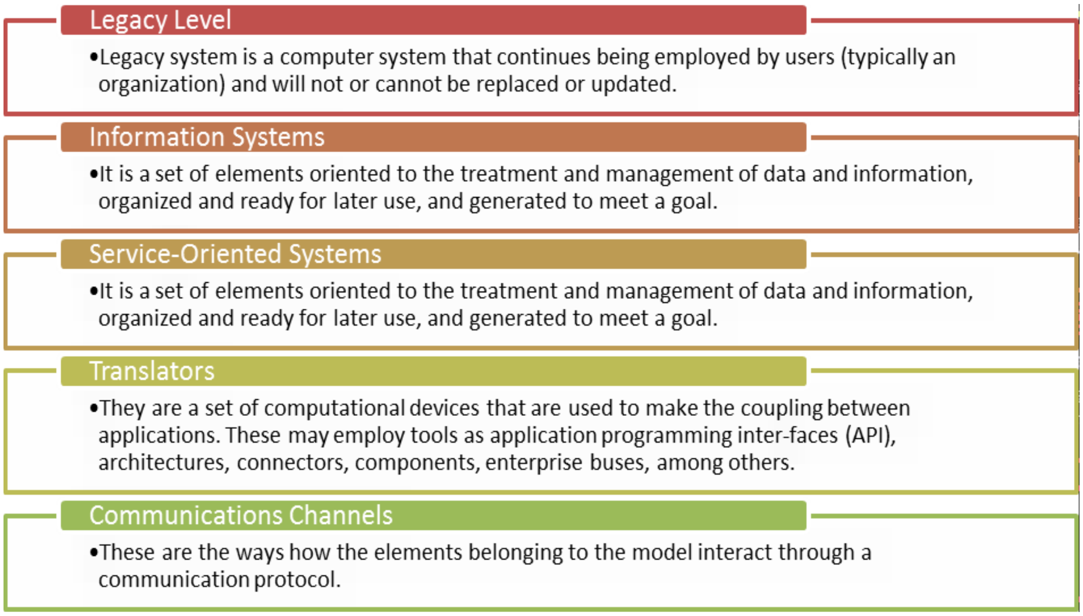


Figure 1: Service-oriented business integration model

A. Conceptual Level

At this level, we identify and describe the software and hardware artifacts, and communication elements necessary for service-oriented enterprise integration. This definition helps to identify each artefact. (view Figure 2)



Legacy Level

- Legacy system is a computer system that continues being employed by users (typically an organization) and will not or cannot be replaced or updated.

Information Systems

- It is a set of elements oriented to the treatment and management of data and information, organized and ready for later use, and generated to meet a goal.

Service-Oriented Systems

- It is a set of elements oriented to the treatment and management of data and information, organized and ready for later use, and generated to meet a goal.

Translators

- They are a set of computational devices that are used to make the coupling between applications. These may employ tools as application programming inter-faces (API), architectures, connectors, components, enterprise buses, among others.

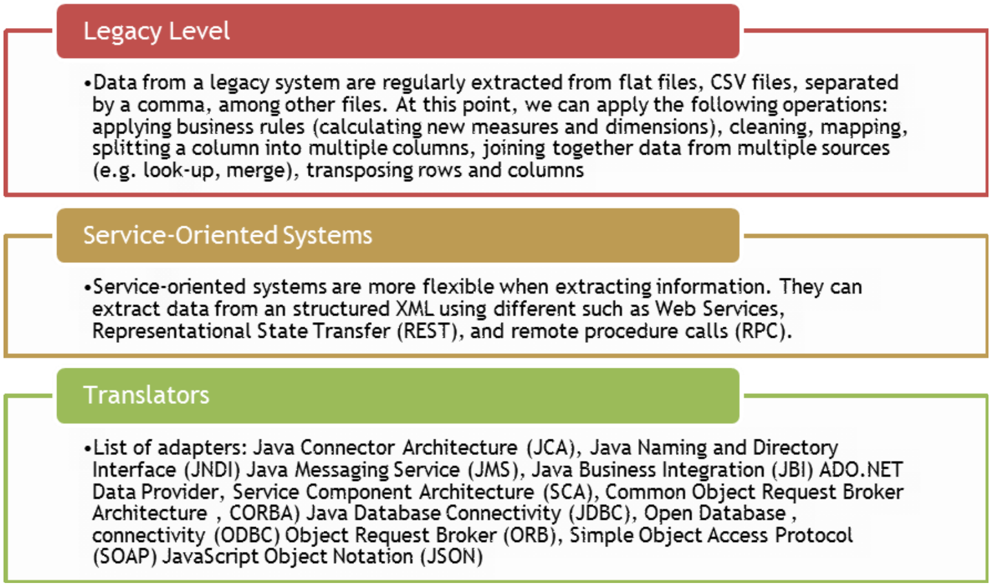
Communications Channels

- These are the ways how the elements belonging to the model interact through a communication protocol.

Figure 2: Identify each artefact in conceptal level

B. Tech Level

At this level, we describe how to extract, transform and exchange information (input/output) of artifacts identified in the conceptual level, to effectively guarantee integration. We will describe the technological elements necessary for integration (view Figure 3).



Legacy Level

- Data from a legacy system are regularly extracted from flat files, CSV files, separated by a comma, among other files. At this point, we can apply the following operations: applying business rules (calculating new measures and dimensions), cleaning, mapping, splitting a column into multiple columns, joining together data from multiple sources (e.g. look-up, merge), transposing rows and columns

Service-Oriented Systems

- Service-oriented systems are more flexible when extracting information. They can extract data from an structured XML using different such as Web Services, Representational State Transfer (REST), and remote procedure calls (RPC).

Translators

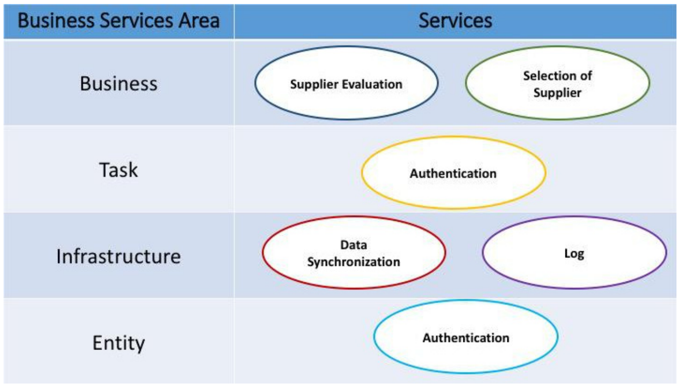
- List of adapters: Java Connector Architecture (JCA), Java Naming and Directory Interface (JNDI) Java Messaging Service (JMS), Java Business Integration (JBI) ADO.NET Data Provider, Service Component Architecture (SCA), Common Object Request Broker Architecture , CORBA) Java Database Connectivity (JDBC), Open Database , connectivity (ODBC) Object Request Broker (ORB), Simple Object Access Protocol (SOAP) JavaScript Object Notation (JSON)

Figure 3: Description of tech level

C. Operational Level

With the aim to implement a service-oriented integration, this level describes the conceptual and technology interaction through a series of steps. These stages are described below.

1. Stage I: Identification Services: At this stage, we identify functionality contained in the legacy software or business processes, to be encapsulated in the form of service. To perform this activity, we can start by answering questions such as: is the encapsulated functionality involved or may become a party to the composition of other business processes? Is it an agnostic process functionality? Is it autonomous enough functionality to allow us sufficient control over its resources to its proper functioning? If at least one of these questions is answered affirmatively, it is correct to think about turning functionality as a service. However, we must not forget the implicit requirements in the implementation of services (security, performance, etc.). In Figure 4 outlines how to identify candidate services to implement the integration.

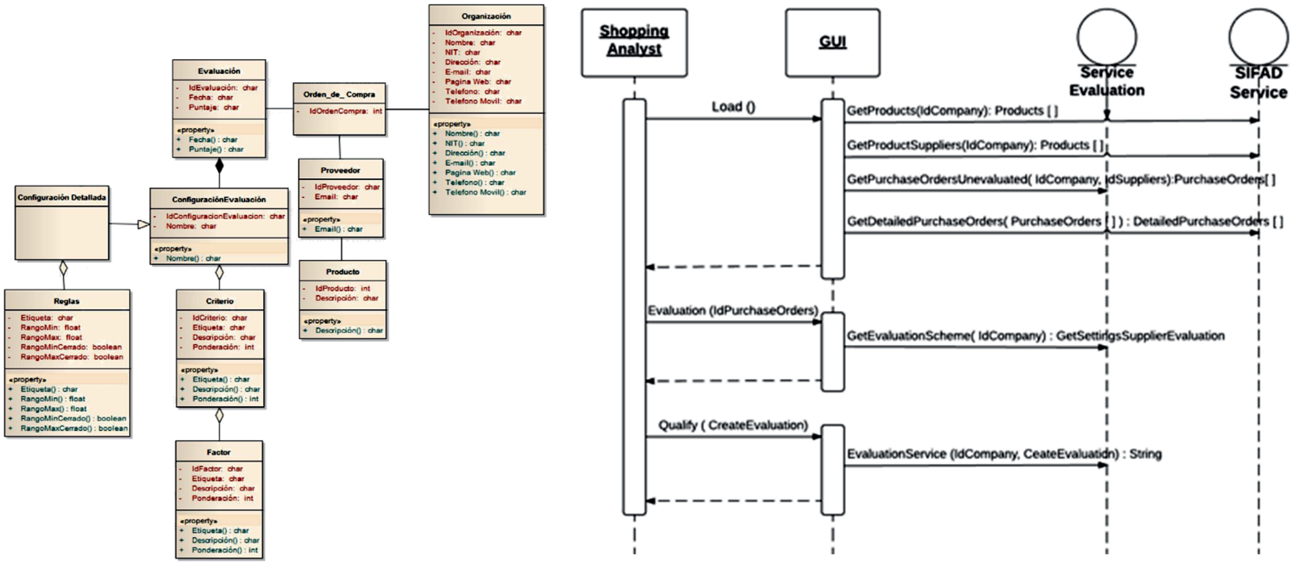


Business Services Area	Services
Business	Supplier Evaluation Selection of Supplier
Task	Authentication
Infrastructure	Data Synchronization Log
Entity	Authentication

Figure 4: Model identification service [14]

2. Stage II. Domain Model: A domain model is a visual representation of conceptual classes or real-world objects in a domain of interest. They are also called conceptual models, domain object models, and

object models analysis [15]. The stage goal is to represent the business object model. It is focused on products that are important elements in business. This model does not include the responsibilities of users in the system. A domain model is an abstraction behind what the company does and what the system does. If we start with a model for the domain, we could make it more concrete through the development to the logical model of a computer system in the domain. However, we could also make the model by develop-ing specific to the business object model of current practices and business processes of a company within the domain. While in an object-oriented designed, the responsibilities remain with the classes that implement the abstractions of domain model, in a model of business objects these responsibilities are "expelled" and assigned to the different functions of work [16]. Figure 4 (a) illustrates the abstraction by a class diagram supplier evaluation.



a) Class Diagram supplier evaluation

b) Sequence Diagram service evaluation

Figure 5: Diagrams to domain service and behaviour models

3. Stage III. Behavior Models: These models respond to the decision to implement the underlying logic of services under an object-oriented paradigm, as well as the overall solution. In any case, the implementation can be performed under any kind of paradigm or traditional technology. Sequence diagrams, activity diagram, or business model can be used for representation of the logic services behaviour. A sequence diagram is shown in Figure 4 (b).

4. Stage IV. Design Services: This activity includes the description of the operations and messages that make up the public service interface. This is the starting point for a correct application of the principles of design, emphasizing a differentiation of a style called remote procedure (RPC) and message-oriented style. At this point, it is possible to identify patterns of message exchange required to perform system functionality as well as the creation of centralized systems for the exchange of messages.

5. Stage V. Coding and Deployment Services: The objective of this stage is to code and develop services using last generation programming languages. It is recommended to use languages or frameworks that support a service-oriented approach as Java Development Kit (JDK) and. NET Framework, and also use the standards and technologies defined by the World Wide Web Consortium (W3C) [17].

6. Stage VI. Testing Services: Integration based on a service-oriented approach has greater challenges and requirements that development under a traditional scheme. Thus, this stage proposes performing a verification of the following aspects:
   6.1. Check that logic is dependent on the contract and that the contract is not dependent logic.
   6.2. Check the autonomy of each service.
   6.3. Check state management between services.
   6.4. Check that services can interoperate with other services. Functional testing.
   6.5. Transactional evidence.
   6.6. Test failures in communication channels.
   6.7. Validation access to Web services meta information Usability testing.

4. CASE STUDY

To validate our integration pattern, we study a case that is part of a critical business process of any organization, the process of vendor selection. The target in this case is the Universidad Tecnológica de Bolívar (UTB) at Cartagena, Colombia. The selection process of providers is not a set of activities undertaken within this organization. In order to give transparency to the selection, the UTB has implemented a clear, specific, and well-defined process to implement a formal selection method supplier based on the information thereof through history. The scope of the case is delimited to answer the requirements described in “Designing a service-oriented computational tool for decision making in the selection of suppliers”. This case has been study before by [18]. The pattern validation proposed in this article was developed in [14], [19]. The following describes how the pattern is applied in the case study above.

A. Conceptual Level

The software and hardware devices identified in the case study of the UTB are: Legacy system called SIFAD, this system manages the database of purchasing and procurement process. Data is stored in a database systems legacy Informix 7.3. The application is hosted in a stovepipe operating system Unix SCO (Santa Cruz Operation). Supplier evaluation system, this system manages and administers the selection and evaluation of suppliers. It is identified as a service-oriented system. The supplier evaluation application is hosted on Web Application Server (Windows Server) and deployed in Internet Information Server 7.0. Data is stored Supplier evaluation system in SQL Server database.

B. Tech Level

The process of extraction and processing of information described below. SIFAD (the information providers) exports data in a text file periodically to a repository. The supplier evaluation system imports the data by a module performing the transformation of the data. In many cases, there are inconsistencies in this process. To overcome this issue, we propose to develop web service data synchronization in order to perform the transformation of data effectively. The implementation of service was done using an ODBC by Informix.

C. Operational Level

Below it is described as implementing the operational level is performed in the case of raised study.

1. Stage I. Identification Services: Identifying the service was developed, implemented the principle of abstraction, mediate the Figure 3. Can be identified, what are the possible services to implement.
2. Stage II. Domain Model: At this stage, the underlying logic of services under an object-oriented paradigm is implemented. We use class diagrams to reflect the logic of each service as well as the overall solution. The classes are part of the conceptual solution evaluation service providers are illustrated in Figure 5 (a).
3. Stage III. Behavior Models: To understand the business logic of the system, we use a sequence diagram as shows Figure 5 (b). In this process, managers and administrators are the main actors that help us to represent the logic business.
4. Stage IV. Design Services: The purpose of this step is to design the service under the principle (standardization of contracts) in order to define the operations, the method of discovery, implemented requirements, domain, data formats input and output. an example of how services should be designed is illustrated in Table I.
5. Stage V. Coding and Deployment Services: Coding and deployment was implemented as follows. The Data Synchronization Service and Service Log was developed in JDK 7 and deployed to GlassFish 2. Services Service supplier evaluation and supplier selection were coded deployed C#. NET Framework 3.0 and Internet Information Server 7. Developed services use a Web interface, contracts are described in Web Service Description Language (WSDL) and the communication protocol used between services is Simple Object Access protocol (SOAP).
6. Stage VI. Testing Services: With the aim of verifying the correct operation of the system and its components, we do functional testing, transactional testing, testing communication channels, and usability testing.

5. CONCLUSION

Actually, the companies today a cannot abandon their legacy software. In order to continue in business, any enterprise needs to apply integration techniques that not imply starting over from scratch. Our service-oriented business integration model allows enterprises to keep using legacy systems at the same time they incorporate new software. Our integration model relays on the identification of five artifacts that play an important during the process (conceptual model). After, we need to find the mechanism to interchange information between the components (tech level). And finally, we star the operational level that drives into a series of steps that goes from the identification to testing services. Our service-oriented business integration model helps the enterprise to improve the relation customer-enterprise, improve the efficiency of the internal business process, improve the business environment by allowing to continue operating their legacy software at the time new services are added to the system.

REFERENCES

[1] A. Puder, K. Romer,¨ and F. Pilhofer, Distributed systems architecture: a middleware approach. Elsevier, 2006.

[2] D. Liebhart, G. Schmutz, and P. Welkenbach, “Service-oriented archi-tecturean integration blueprint,” 2010.

[3] W. Roshen, SOA-Based Enterprise Integration: A Step-by-Step Guide to Services-based Application. McGraw-Hill, Inc., 2009.

[4] M. P. Papazoglou, “Service-oriented computing: Concepts, characteristics and directions,” in Web Information Systems Engineering, 2003. WISE 2003. Proceedings of the Fourth International Conference on. IEEE, 2003, pp. 3–12.

[5] G. Hohpe and B. Woolf, Enterprise integration patterns: Designing, building, and deploying messaging solutions. Addison-Wesley Professional, 2004.

[6] W.-T. Tsai, B. Xiao, R. Paul, Y. Chen et al., “Consumer-centric service-oriented architecture: A new approach,” in Software Technologies for Future Embedded and Ubiquitous Systems, 2006 and the 2006 Second International Workshop on Collaborative Computing, Integration, and Assurance. SEUS 2006/WCCIA 2006. The Fourth IEEE Workshop on. IEEE, 2006, pp. 6–pp.

[7] J. He, M. Chang, and E. Castro-Leon, “Evolution of the intel’s e-business data center toward a service-oriented infrastructure,” in e-Business Engineering, 2005. ICEBE 2005. IEEE International Conference on. IEEE, 2005, pp. 2–9.

[8] X. He, H. Li, Q. Ding, and Z. Wu, “The soa-based solution for distributed enterprise application integration,” in Computer Science-Technology and Applications, 2009. IFCSTA’09. International Forum on, Vol. 3. IEEE, 2009, pp. 330–336.

[9] W.-T. Tsai, “Service-oriented system engineering: a new paradigm,” in Service-oriented system engineering, 2005. sose 2005. IEEE Interna-tional Workshop. IEEE, 2005, pp. 3–6.

[10] C. Koch, “The truth about soa,” CIO Insider, 2006.

[11] M. Chang, J. He, W. Tsai, and Y. Chen, “User-centric service-oriented architecture,” in IEEE International Workshop on Service-Oriented Sys-tem Engineering (SOSE), Shanghai October, 2006.

[12] E. Castro-Leon, J. He, and M. Chang, “Scaling down soa to small businesses,” in Service-Oriented Computing and Applications, 2007. SOCA’07. IEEE International Conference on. IEEE, 2007, pp. 99– 106.

[13] Ronald. Zamora Musa. 2011. Analisis de requerimiento para la implementación de laboratorios remotos. Technical Report. Educosta. ISBN: 978-958-8710-78-5.

[14] E. F. Castro and N. Z. Alarcon,´ ”Validacion´ de la computacion´ orientada a servicios como nuevo paradigma para la construccion´ de software empresarial.” 2011.

[15] D. Georgakopoulos and M. P. Papazoglou, Service-oriented computing. The MIT Press, 2008.

[16] M. Turner, D. Budgen, and P. Brereton, “Turning software into a service.” Computer., Vol. 36, No. 10, pp. 38–44, 2003.

[17] D. Brickley and R. V. Guha, “Resource description framework (rdf) schema specification 1.0: W3c candidate recommendation 27 march 2000,” 2000.

[18] G. V. Mendoza and Y. P. Cruz, “Supplier selection service-case study utb,” in Computing Congress (CCC), 2012 7th Colombian. IEEE, 2012, pp. 1–6.

[19] H. de la Hoz Escorcia and E. R. Barragan,´ “Guía de buenas prácticas´ para la correcta integración´ de aplicaciones empresariales en un ambiente heterogéneo´ y orientado a servicios, a través´ de la tecnología esb.” 2012.