Method for the Integration of Applications based on Enterprise Service Bus Technologies

JOSÉ VICENTE BERNÁ MARTÍNEZ[1], CLAUDIA IVETTE CASTRO ZAMORA[2], FRANCISCO MACIÁ PERÉZ[3], CARLOS RAMÓN LÓPEZ PAZ[4]
[1,3]Higher Polytechnic School, Universidad de Alicante (University of Alicante), San Vicente del Raspeig, Spain
[2,4]Faculty of Computer Engineering, Universidad Tecnológica de La Habana (Havana University of Technologies José Antonio Echeverría), Havana, Cuba
[1]jvberna@ua.es, [2]ccastro@ceis.cujae.edu.cu, [3]pmacia@ua.es, [4]carlosr@ceis.cujae.edu.cu

Abstract: - Companies have developed and acquired multiple business software applications to respond to new organizational requirements. In many cases, the applications have logical dependencies or are used in isolation. This practice increases the possibility of errors, produces inconsistencies and duplications of data, and hinders management of the organization's processes. To take into account the complexities of an integration initiative in terms of integration technologies, staff capabilities and the business applications themselves, organizations that are new to Enterprise Application Integration should follow guidelines and best practices to implement their integration solutions in an integrated manner. The present work conceptualizes a method for developing an integration solution based on Enterprise Service Bus technologies. To achieve this objective, a case study methodology is used for the conception and evaluation of the proposal in terms of systematizing the lessons learned in the development of projects to integrate applications in multiple Cuban organizational sectors in the 2013-2016 period. The proposed method is independent of technology and is modelled as a process in terms of stages, activities and artefacts. The results obtained offer a mechanism that guides and favours the development of a service-based flexible integration solution using integration technologies such as the Enterprise Service Bus.

Key-Words: - Application Integration, Enterprise Service Bus, web services

1 Introduction

It is common for the information technology areas of organizations to identify situations such as duplicate information, growing availability of business software applications with a diversity of information technologies, out-dated information and the performance of unnecessary manual activities in their information solutions [1]. These situations are due in part to the implementation by companies of new initiatives to automate processes without adequately reviewing existing technologies, logic and information systems deployed in the organization. There are also cases where intercommunication of applications is essential due to changes in the structure of the organization or its business model, such as those that occur in the processes of the merger and acquisition of companies. One solution to these problems is Enterprise Application Integration (EAI).

Linthicum [2] defines EAI as the unrestricted exchange of information between two or more business applications, in which a group of technologies allow the exchange and flow of information between different applications and business processes in the same company or between different companies. EAI can also be viewed as an approach that provides methodologies, technologies and tools that help reduce the cost of integration [3,4].

To satisfy this need, different software providers offer a set of proprietary EAI tools that simplify the different integration tasks and eliminate point-to-point connections between applications (messaging broker). Recently, the adoption of web services technology and Service Oriented Architecture has changed the paradigms of both application development and integration [5]. Thus, web services, while not an alternative to EAI technologies, are a significant part of them [6].

EAI is more than a set of tools such as a messaging broker and technologies such as web services. EAI also involves plans and methods that are applied in a structured way to integrate business processes into IT infrastructure [6]. Lam and Shankararaman [7] define three types of integration components: the legacy systems that need to be integrated, the manual processes that require the development of new IT systems for automation, and

the automation of manual processes that extend legacy systems.

The object of integration is business applications. Business applications are defined as systems that coordinate activities, decisions and knowledge through different functional levels in an organization in a specific business sector [8], whether for a general or specific purpose. This type of system is classified as any business software comprising sales systems, material management systems, project management systems, production planning systems, human resources systems, supply chain management systems, and general resource planning systems such as Enterprise Resource Planning (ERP) systems.

The implementation of an application integration solution is facilitated by procedures that organize the integration process. In addition, these procedures should focus on formalizing good practices as a means to reuse previous experiences and adapt to new additions of customers and suppliers [9]. A formalized integration process is even more justified when a company does not have specialized personnel in the area of integration.

The levels of formalization of good integration practices can be conceptualized with the case study method [10]. This is an observational research method based on the projection and collection of multiple sources of evidence that rigorously allows both the evaluation and construction of theories from the exploration, description and explanation of phenomena in their actual context. In this work, the phenomenon under investigation is the diversity of scenarios of EAI.

Thus, this paper analyses different integration solutions developed in academic-business contexts in the period 2013-2016 as case studies. The results obtained are systematized as an integration process that has value for guiding the development of an integration solution in a flexible manner.

Following this introduction, the related work is reviewed in section 2. The theoretical propositions of the case are presented in section 3. In sections 4 and 5, each of the case studies that constitute the basis of the proposed method are presented. The integration process, defined as the GeIntegrationApp method, is described in section 6 and is validated in section 7. Finally, in section 8, the main conclusions of this article and future lines of work are described.

2 Related Work

Enterprise Application Integration Methodology (EAIM) [11, 12] describes a structured approach that responds to the implementation needs of the enterprise architecture [13]. Among the recommended practices for using EAIM is Integration [14], which is used to integrate processes and applications across the company. Among the EAIMs consulted that are dedicated to the practice of Integration, most use experiences in different sectors [7, 15, 16, 17] as the basis for constructing methods. Some studies, such as [15, 17], rely on the case study method for the validation of the proposed solution, while others, such as [7, 16], do not present evidence of validation. For this article, we will use a subset of the elements proposed in [16] to address the integration of applications with a systemic approach. The selected elements are Stages, Work Units, Participants and Work Products. Table 1 is a comparative table that shows the main features of the reviewed methodologies.

Table 1 reflects different aspects of existing integration methods. The stages refer to those within the life cycle of the business architecture and are indicated as follows: I (planning), II (analysis), III (design) and IV (implementation). The table shows whether the work identified takes into account these stages (Yes or No). In addition, the absence, slight presence or presence of the work products is also analysed.

Integration methods	Stages				Roles	Construction	Validation
	1	2	3	4			
Lam, W. and V. Shankararaman [7]	Y	Y	Y	N	Absence	Experience without specifying sector	No evidence
Themistocleous, M. and Z. Irani [20]	Y	Y	N	N	Absence	Based on [7]	Case study in multiple sectors
Silveira, R., J. Pastor and E. Mayol. [16]	Y	Y	N	N	Slight presence	Experience, ISO/IEC24744 Model Other case studies Interviews with experts	No evidence
Van Den Bosch, M.A.P.M. et al. [17]	Y	Y	N	N	Presence	Based on [7, 15, 16] Design Sciences	Case studies

Table 1. Dimensions of the integration methods

3 Theoretical Proposals of Enterprise Service Bus Technologies

The theoretical propositions of the case mark the beginning of the study. Integration projects develop in a real context, and as developed in later sections, they assume that web services, although they have played a

leading role as an integration solution, still have the following limitations [18]. (a) It is not possible to have control over the information that is exchanged between the applications because each application is responsible for connecting with the rest. (b) Each client application is responsible for updating when there are changes in the interfaces of the provider applications. (c) For medium and large businesses, the increase in point-to-point links hinders maintenance tasks. (d) There are legacy business applications that do not natively support web services, and therefore, their integration occurs through messaging, which is a limitation for reuse by other business applications or services that need it. (e) Format incompatibilities in the information that is exchanged must be resolved by each application that participates in the integration. (f) The survey of the composition of the information is performed on each client machine, and its reusability for the rest of the applications is null.

An Enterprise Service Bus (ESB) solves these problems [19] by providing a framework that manages services and their integration with others through a common platform. In addition, the ESB acts as a mediator between the service provider and its associated clients. The ESB also provides the support infrastructure needed to implement message routing, protocol translation and transformation and acts as a mediating agent that supports interconnection for multiple services.

This communication is usually based on the exchange of messages between the two parties. The ESB allows the messages received from the clients to be processed and sends them to the designated service point. In addition, it provides a set of interfaces or endpoints for different channels, as well as incorporates various types of IT resources. The ESB allows the incorporation of suppliers and service applicants with different communication interfaces. During the communication, the service applicants do not need to know their identity; it is the responsibility of the ESB to manage this identity for each service applicant.

The main capabilities of an ESB model to assume the development of an integration solution are [20] service repository, policy based on secure messaging, communication protocols, service discovery and monitoring. ESB technology is an integration model that combines messaging, web services, data transformation and intelligent routing.

3.1 Mule and WSO2 technology

Mule is a free open-source software project developed by Ross Mason and belonging to the company MuleSoft [21]. Mule is a Java-based software tool with architecture that is inspired by the ESB concept that supports the design and implementation of EAI solutions based on integration patterns [22].

Mule technology provides tools to support developers in the creation and configuration of an XML file that contains the Mule flow definitions [21], such as plugins for Eclipse, MuleStudio and AnypointStudio. Mule provides the possibility to monitor and manage the different flows of applications that are deployed in the ESB. In addition, Mule permits the visualization of the resources of the implemented applications, whether they are flows, components or messages that are exchanged between components, as well as the status of the server where it has been deployed in terms of consumption of physical resources [21].

For its part, WSO2 is an ESB tool designed for a service-oriented infrastructure under Apache Software License v2.0. The technology allows developers and designers to configure the message for the actors involved in integration through routing, mediation, and transformation [23].

With respect to the functionalities it provides to the developer, WSO2 allows the creation, administration and deployment of ESB projects through the Eclipse Developer Studio plugin, in addition to the ESB management console, which allows the creation and administration of integration projects [23].

4 Mule-ESB Case Study

During the 2013-2016 period, an application integration project was developed at the Havana University of Technologies. During 2013-2015, the project focused on the study and understanding of ESB technologies as part of an effort to validate technologies to develop IT solutions as integration services. Mule technology was selected because of the potential flexibility of its components for modelling integration flow.

4.1 Preparation of the case

The selected integration scenario was an Intranet at the faculty level and a system for the management of academic activity. The Intranet offered information related to the different processes carried out by the faculty; however, the update mechanism was based on manual updates. The academic management system, in contrast, offered

information regarding the teaching and personal data of the students through web services. The services were developed with the Java programming language and had security levels such as basic authentication and digital certification. Thus, integration of the Intranet with the academic management system was proposed to obtain the information associated with the students.

The case was designed based on the "MuleESB technological environment" context; the analysis unit was the ability to orchestrate secure, developed services with Java technology.

4.2 Evidence collection strategy

The evidence collection strategy was developed based on the units of analysis. Evidence must be obtained from multiple sources to minimize biases on the part of the researchers and thus guarantee their accuracy [24]. The most significant types of evidence are (obtained in the Mule-ESB case):

- Documents: Technical reports such as manuals or evaluation results of ESB tools. Scientific articles with evaluations of ESB. Regulations, good practices, procedures and documentation templates.
- Computational artefacts: Integration flows to be collected during the development of an integration initiative with Mule.
- Observations: artefacts defined to collect evidence of the analysis of certain aspects in the investigation. These include videos, images or spreadsheets related to the assimilation of tools and problems presented during the development of the projects. The technique used is the thinking aloud protocol [25].

In general, the documents consulted contributed to form a theoretical basis for working with the tool. However, no guidelines or recommendations were found on how to guide the development of the integration requirements, which necessitated considerable additional effort on the part of the team responsible for designing the specific flow of integration.

At the end of the work, a set of documentary evidence was obtained to generate a step-by-step guide with a specific solution complementing the collected documentation.

The integration flow included an orchestration involving the consumption of three services with different features; subsequently, all collected information was integrated and returned in JSON format.

4.3 Analysis of the evidence

In the first stage of the case, a study of the service layer provided by the academic management system was performed. The conclusion of this study was that no single operation responded to the need for integration. However, some operations were identified that, when combined, could obtain the desired result. Bearing in mind that one of the capabilities of the ESB is orchestration and reuse by other clients, the development of integration with ESB technology was considered.

It later became necessary to create a pseudo-code that would define the logic of the orchestration to solve the need. At this step, clarity of the different activities to be executed in the integration flow was very helpful prior to implementation.

In addition, during the process of implementing the Mule flow, transformations required to prepare the input objects for the different operations to be consumed were identified. The message within the flow was also modified by the features of the SOAP message that was consumed. Other specifications, such as definitions of the flow variables for storing the responses of the operations consumed and their subsequent use in the flow, were identified.

Moreover, systematizing the lesson learned in this exploratory case revealed that the three operations to be invoked within a flow designed in Mule required different levels of security, such as basic authentication and digital certification. The flow invoked two services of the academic management solution with one and two operations, respectively. This feature required a study of a global element in the HTTP_HTTPS flow that specified the physical address of the certificate, as well as its credentials and basic authentication.

Once the case was executed, the following conclusions were reached:

- Understanding the integration scenario and identifying and characterizing the data sources can lead to approach options such as web services or ESB, among others.
- Defining the logic of integration as a pseudo-code beforehand can organize and guide the process of implementation of integration in the ESB.
- Characterizing the operations to be invoked in terms of security levels and input and output data contributes to clarifying the integration solution prior to implementing it.

- The design of the service providers determines the use of the components in the implementation of integration flow.
- Providing a service layer with medium granularity, as in the case of the academic management solution, allows a combination of different operations and thus responds to the proposed need for integration.

5 Case Study Mule- WSO2-ESB

The second case corresponded to the evaluation of the WSO2-ESB tool as an alternative to ESB technologies. Consequently, a new case was designed that focused on explaining how both integration technologies responded to the same need for integration. The observers were participants without previous knowledge of either of the two technologies.

5.1 Preparation of the case

The selected integration scenario was the integration needs of a project management application implemented in .NET. The application under study was the Team Foundation Server (TFS) tool. It was decided to develop a service layer that would allow reuse by third parties. Consequently, the service layer was presented in a single integration context with two analysis units: the Mule solution and the WSO2ESB solution.

The evidence collection strategy was identical to that of the Mule case.

5.2 Analysis of the evidence

The analysis of evidence shown in Table 2 illustrates how both ESB tools respond to the main integration activities. The two tools have similarities in terms of the components and mediators that invoke, transform and route the message. ESB Mule provides more message manipulation options than WSO2ESB. This difference implied that the activity demands more knowledge from the developer since the structure of the message that travels within WSO2ESB is XML.

During the execution of the case, it was evident that identification of the operations was of great importance in the planning and creation of the integration logic. In this way, unnecessary loss of time was avoided during the implementation of the orchestration because the input and output parameters required by each operation were previously known.

Integration act.	MuleESB	WSO2ESB
Invoke operation	SOAP, HTTP	PayloadFactory, Call
Manipulate the message that travels inside the ESB	SetPayload, Expression, Java Variable	PayloadFactory, Header
Iterate a fix	Foreach	Iterate
Transf. to XML	Object to XML	PayloadFactory

Table 2. MULE - WSO2 integration activities

In addition, the characterization of the operations in terms of data complexity allowed the adoption of strategies focused on the variables to be used during the orchestration regardless of the ESB. With respect to technology, the characterization of the operations allowed the configuration of the invocation of the necessary services to be previously defined. In this way, errors and unnecessary difficulties were avoided. If the operation requires security, then components and configurations are available within the orchestration.

Regarding the development of the integration solutions with the Mule and WSO2ESB tools, it is concluded that Mule allows trace tracking during flow execution through its Debugger View. Mule allows the data of the flow message, the message processor, and their structure to be visualized by showing the values and types of data. To perform debugging, breakpoints can be defined for most components.

Mule provides a variety of transformers. However, if the actions performed in the flow are very specific and the available components cannot meet these needs, the Java component can be used. Methods, objects or variables can be created to perform the necessary transformations. When using these components, it must be borne in mind that when receiving parameters internally, the flow message must be used and allowed to install itself with all functionalities and methods it has as an object.

In addition, Mule proposes its own language, MEL, for the configuration of some components. For some developers, this may be a disadvantage because they must know another language to develop integration flows, but MEL is very useful for the flexibility of integration. MEL also provides a wide variety of output points that facilitate the consumption of information from any data source. Similarly, there a variety of points of entry, which

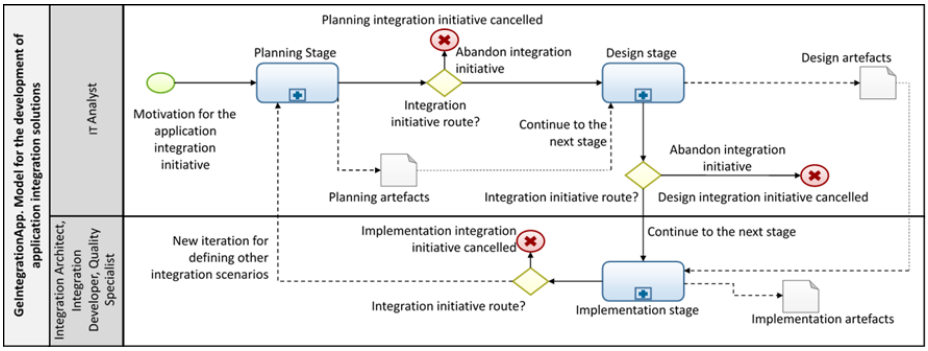

Figure 1. GeIntegrationApp as a process

exposes the flow developed through different mechanisms.

For its part, the WSO2ESB analysis unit does not incorporate the option of debugging the sequences, which is advantageous for developers to determine which values that reach the mediators, particularly when an XML message travels between components. As an alternative, a mediator property can be added to the sequence to store the value, and a mediator log can be used to show the value in the console. In this way, the data obtained can be visualized, and any failure can be corrected if necessary.

Based on the experience acquired with this case and the previous case, a set of steps are proposed to guide the development of integration solutions, and artefacts are proposed for the documentation and organization of the solution. The steps are: identify integration needs; identify and characterize the possible service providers; identify services and operations of suppliers; characterize the operations to be consumed; perform a pseudocode with the logic of the integration solution, identify the possible components or mediators in the ESB tool that respond to integration activities; and implement the integration.

6 The Application Integration Method as a Process

The development of both case studies is conceived under the research paradigm of Design Science Research (DSR) [26]. In this context, case studies are appropriate to investigate in depth a context of real and contemporary integration, which is a typical scenario in these types of initiatives.

In addition, the perspective of a case study under DSR gives this research process a reflective approach since during the execution of the projects with research integration initiatives as their goal, the general types of problems related to technologies and integration methods that must be conceptualized and systematized will be reflected.

The method is conceived independent of technology, although its conceptualization is based on lessons learned from specific technologies. Therefore, the integration process to be modelled combines both theoretical propositions and practical experiences of systematized integration. The collection of evidence based on the units of analysis identified enables a revealing analysis of possible scenarios in which the findings can be replicated in similar contexts.

Consequently, the application integration method is conceived as generic in terms of stages, activities, roles and artefacts. It is called GeIntegrationApp and will guide the development of application integration solutions with a focus on web services. The bases of GeIntegrationApp are:

- The method focuses on an application integration oriented to web services.
- The design principles proposed by SOA for the development of interoperable services are incorporated.
- ESB is the integration technology. In this sense, the selection of this type of technology is facilitated.
- It is mainly aimed at business units that provide IT solutions and are in the initial stages of implementation of application integration initiatives.
- The method conceptualizes the principle of design first and then installs the selected integration technology.
- The activities are defined in the methods in a general way as a guide. Each integration team can complement them with other resources, techniques, methods or specific procedures as required.

The defined method follows three stages: Planning, Design and Implementation, as shown in Figure 1. The premise of the method is based on the activities pursued in the systematization of related work that address invariants to implement an integration initiative. In addition, the method takes as resources the results of the study of the executed

cases. The phases of analysis of these cases are more detailed, and the findings are formalized in terms of activities to be replicated in analogous contexts.

There is precedence between each of the stages, which guarantees that the previous stages have been completely executed. Next, each of the activities are specified in stages. Due to space issues, the process models of each subprocess of the method and the specifications of each template that are used in the activities by phase are not included in this work.

6.1 Planning stage

In the Planning stage, as the main objective, the necessary aspects to start an EAI initiative are defined. In this stage, the structure of the company is learned to understand the processes and the different business applications that support these processes. The integration needs are also defined, as well as the design of the integration scenarios. This stage is divided into three activities: understand the structure of the company, specify the integration needs and define the integration scenario.

6.1.1 Understand the structure of the company

In this activity, each of the processes and business operations are analysed. The portfolio of business applications that the business unit of an organization has among its IT service providers is identified and characterized.

In this activity, an integration map is designed to reflect the dependencies between the different applications in terms of the information required by the business that is shared. The information resulting from integration has strategic value for the technology areas because it streamlines the operations in the business that are supported by these applications to be potentially integrated. For this purpose, the integration workflow of GeIntegrationApp is complemented with technical documentation specific to each application object of integration.

The formalization of this activity is complemented by the artefacts Diagnostic Template of the IT structure and Integration Map. The Diagnostic Template of the IT structure presents a characterization of the company's technological situation, the description of its own systems and legacies, and the service layers. The Integration Map indicates the dependencies between the applications, where the degree of dependency is the integration flow itself.

6.1.2 Specify the integration needs

This activity begins with a detailed understanding of the application integration map and proposes to identify the types of needs that are present in the organization at a global level or locally between two or more applications. For each need, the level of priority for its solution must be established (low, medium, high).

The formalization of this activity is complemented with the artefact Specification Template of Integration Needs. It establishes and documents the integration needs at a business or private level between two or more systems.

6.1.3 Define the integration scenario

This activity specifies the scenarios that will solve the identified integration needs. Providers and customers are specified, taking into account that a provider is the application that provides the information required by another client application.

The formalization of this activity is complemented by the artefact Description of the integration Scenario Template. It presents a global view of the integration scenario, where customers, suppliers and integration requirements are involved.

6.2 Design stage

In the Design stage, the objective is to characterize the data sources and define the logic of the integration. This stage addresses the IT Analyst profiling the integration before deciding what technology will be used for its solution. To accomplish this objective, two activities are proposed: characterize the provider sources and define the logic of the integration scenario.

6.2.1 Characterize the provider sources

This activity can concern the database type or service layer of the provider systems. For a service layer, the services and operations to be invoked in the integration must be identified. In addition, the operations are characterized in terms of entry and output parameters and security requirements. For the database case, a physical model of the database must be built or consulted.

The formalization of this activity is complemented by the artefact Characterization of the Web Services Template. This artefact

characterizes the operations involved in the integration scenario.

6.2.2 Define the logic of the integration scenario

In this activity, a pseudo-code is proposed that allows the logic of the integration to be defined and guided. Flow diagrams can be used for this activity.

6.3 Implementation stage

In the Implementation stage, the integration technology to be used in the integration is selected: web service or ESB. If ESB is used, the ESB technological product that matches the needs and characteristics of the company is selected. In addition, the activities related to the implementation of the solution are performed considering the design principles proposed by SOA, and the relevant tests are conducted to guarantee the quality of the solution and its subsequent deployment. This stage is divided into three activities as shown in Figure 2.

6.3.1 Select technology for integration

In the select technology for integration activity, the architecture model that best fits the need for integration is selected. For this, the design of the integration scenario and the diagnosis of the IT structure of the organization are taken into account.

6.3.2 Select the ESB technology

In the select the ESB technology activity, a guide for the selection of an ESB product is proposed by identifying elements that fit the needs and characteristics of the company. The formalization of this activity is complemented by the artefact Evaluation of the ESB Technology Template. This artefact provides a set of features that support the evaluation of the ESB technologies identified for integration.

6.3.3 Implement the integration scenario

This activity considers the SOA principles for the design and implementation of interoperable web services. In addition, it proposes to document the errors found during the implementation of the integration scenario. The formalization of this activity is complemented by the artefact Error Specification Template.

6.3.4 Test the integration scenario

In the test the integration scenario activity, relevant tests are performed to guarantee the quality of the integration solution. The defects identified are corrected, and a cycle of testing and corrections is established. In addition, the integration scenario solution is deployed in a development environment to perform quality of service (QoS) tests. The formalization of this activity is complemented with the artefact Specification of QoS Indicators Template. It documents the values associated with the measured QoS indicators for further analysis.

6.3.5 Deploy the integration scenario

The activity deploy the integration scenario has the objective of installing and configuring the necessary infrastructure for the consumption of the integration solution by the clients, namely, the Application server or ESB server, service registration client, and the tool for monitoring the services exposed for the deployment of the integration. The formalization of this activity is complemented by the artefact UML Deployment Diagram.

6.4 Roles

Table 3 shows the roles involved and their responsibilities for executing the activities proposed in the different stages.

The IT Analyst is responsible for diagnosing the technological situation of the company. The IT analyst identifies the integration needs, designs the integration scenarios, and selects the best-fitting integration model. In addition, the IT analyst designs the logic of integration. All of this workflow occurs in the Planning and Design stages of the integration.

The roles involved in the Implementation phase

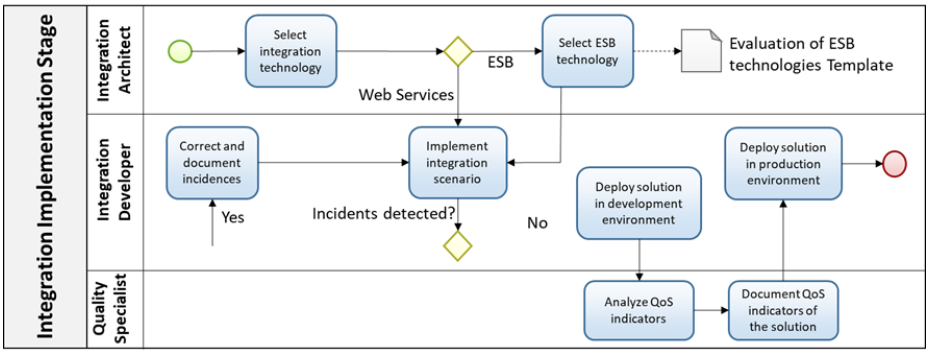

Figure 2. GeIntegrationApp implementation stage

of the integration are the Integration Architect, the Integration Developer and the Quality Specialist.

Each of these roles fulfils certain functions. The Integration Architect is responsible for selecting the integration model that fits the integration scenario. In addition, when choosing the ESB model, the Integration Architect is responsible for selecting the ESB technology that adapts to the specific characteristics of the company. The Integration Developer is responsible for configuring and implementing the ESB technologies or the service layer with the necessary functionalities to meet the integration needs. Finally, the Quality Specialist is responsible for executing the tests to evaluate the quality of the integration.

Table 3 specifies how each of the theoretical propositions were systematized in each stage and activity of GeIntegrationApp.

	Theor. Propo	Case study
Stage Planning		
Understand the structure of the company	[7,15]	
Specify the integration needs		
Define the integration scenario	[15]	X
Stage Desing		
Characterize the provider sources		X
Define the logic of the integration scenario		X
Stage Implementation		
Select the technology for integration		
Select the ESB technology		X
Implement the integration scenario		X
Test the integration scenario	[15]	
Deploy the integration scenario		

Table 3. Theoretical propositions systematized in GeIntegrationApp

7 Validation

During the course of 2016, IT consultancy services were provided to a company that provides IT services in the Cuban pharmaceutical and biotechnological sector. The integration solution was executed in the form of consultancy as a process (or coaching-type IT consultancy). In this case, the client of the consultancy was not the final client but an intermediate client who was assisted to implement integration strategies for subsequent application in certain pharmaceutical companies and biotechnology research centres in Cuba.

In a coaching-type consultancy, it is the client who defines the workflow to perform the integration solution based on a work system defined by the client's rhythm. In addition, this intervention modality is a process consultancy that performs interventions as an expert, where the consultant completes the integration solution that he will then make available to the clients of the IT service provider. This real context is based on both types of client intervention for GeIntegrationApp validation.

Under a working system of coaching-type consultancy, the client is assisted with developing an integration solution for an inventory management system module deployed in pharmaceutical and biotechnological companies in Cuba. In this project, the method adjusted the integration requirements, and it was not necessary to modify or adjust any of the definitions.

However, important elements were added to ensure greater ease and guidance in the development of integration solutions. For example, the identify restrictions activity was included in the Design stage of the integration scenario. It was noted that the restrictions of the provider applications could determine the integration scenario. Some of these restrictions could be availability of source code, security levels, and support for the development of web services, among others.

In addition, the activity define the logic of the integration scenario must be complemented in the Design stage of the integration scenario. The utility of this addition is justified by the need to define SQL queries for each requirement of the integration scenario when the source provider is a database.

8. Conclusions

From the results obtained in this work, it can be concluded that the SOA approach offers a set of technologies, such as web services and ESB, that can reuse the business logic and use potential information that can be shared by several applications. In addition, the SOA approach offers a set of guidelines that allow the development of reusable and interoperable integration solutions.

The works identified as procedures, guides and methodologies for the integration of applications did not provide comprehensive proposals regarding the different components that must be established to integrate applications in different contexts.

The design of the web services providers was a determining factor for minimizing problems during

the development of an integration solution with ESB technology.

The case study method was used as the methodological tool for the construction and evaluation of the proposed method in multiple contexts of application integration and in different organizational sectors.

References:

[1] Curl, A. and K. Fertalj. A review of enterprise IT integration methods. in *Proceedings of the International Conference on Information Technology Interfaces*, ITI. 2009.

[2] Linthicum, D.S., ed. *Enterprise Aplication Integration*. First Edition ed. 2000, Addison Wesley. 400.

[3] Chalmeta, R. and V. Pazos, A stay-by-step methodology for enterprise interoperability projects. *Enterp. Inf. Syst*., 2015. 9: p. 436-464.

[4] Z. Frantz, R., R. Corchuelo, and F. Roos-Frantz, A methodology to evaluate the maintainability of enterprise application integration frameworks. *Web Engineering and Technology*, 2015. 10: p. 334–354.

[5] Silveira, R.W., J.A. Pastor, and E. Mayol, *Towards a method for enterprise information systems integration*. 2010.

[6] Singh Bhadoria, R., N. Chaudhari, and G. Singh Tomar, The Perfomance Metric for Enterprise Service Bus (ESB) in SOA System: Theoretical underpinnings and empirical illustrations for information processing. *Information Systems*, 2016. 15.

[7] Lam, W. and V. Shankararaman, An Enterprise Integration Methodology. *IT Professional*, 2004. 6(2): p. 40-48.

[8] Wu, J., Y. Zhu, and H. Zhu, The definition and implementation of flexible architecture for enterprise application, in *Advanced Materials Research*. 2011. p. 256-260.

[9] Kamal, M.M., R. Hackney, and M. Ali, Facilitating enterprise application integration adoption: An empirical analysis of UK local government authorities. *Int. J.l of Information Management*, 2013. 33(1): p. 61-75.

[10] Runeson, P., et al., *Case study research in software engineering: guidelines and examples*. 2012, Hoboken, New Jersey: Jhon Wiley & Sons,.

[11] Clark, T., B.S. Barn, and S. Oussena, A method for enterprise architecture alignment. *Practice-Driven Research on Enterprise Transformation*, Springer, 2012: p. 48-76.

[12] Darvish Rouhani, B., et al., A systematic literature review on Enterprise Arquitecture Implementation Methhodologies. *Information and Software Technology*, 2015. 62: p. 1-20.

[13] Medini, K. and J.P. Bourey, SCOR-based enterprise architecture methodology *Int. J. Comput. Integrat. Manuf*., 2012.

[14] Darvish Rouhani, B., et al., A systematic literature review on Enterprise Architecture Implementation Methodologies. *Information and Software Technology*, 2015.

[15] Themistocleous, M. and Z. Irani. Towards a methodology for the development of integrated IT infrastructures. in *P. of the Annual Hawaii Int. Conference on System Sciences.* 2006.

[16] Silveira, R., J. Pastor, and E. Mayol. Towards a method for enterprise information systems integration. in ICEIS 2008 - *Proceedings of the 10th International Conference on Enterprise Information Systems*. 2008.

[17] Van Den Bosch, M.A.P.M., et al., A selection-method for enterprise application integration solutions, in *Lecture Notes in Business Information Processing*. 2010. p. 176-187.

[18] Caselli, V., C. Binildas, and M. Barai, The Mantra of SOA. *Service Oriented Architecture with Java*. Birmingham. 2008, UK.

[19] Sharma, D. and D.K. Mishra, A role of enterprise service bus in building web services, in *Exploring Enterprise Service Bus in the Service-Oriented Architecture Paradigm*. 2017. p. 46-58.

[20] Singh Bhadoria, R., N. Chaudhari, and S.T. G. , The Perfomance Metric for Enterprise Service Bus (ESB) in *SOA System: Theoretical underpinnings and empirical illustrations for information processing. Information Systems*, 2016. 15.

[21] Dossot, D., d´Emic, J. and Romero, V. *Mule in Action.* 2º edition. Ed. Manning. 2014.

[22] Zimmermann, O., et al., A decade of enterprise integration patterns: A conversation with the authors. *IEEE Software*, 2016. 33(1): p. 13-19.

[23] García, I.F. WSO2 platform SOA OPENSOURCE. 2018; Available from: http://www.wso2.com.

[24] Yin, R.K., *Case Study Research: Design and Methods*. 5 ed. 2014: Sage.

[25] Genero Bocco, M., J.A. Cruz-Lemus, and M.G. Piattini Velthuis, *Métodos de investigación en ingeniería del software*. 2014.

[26] Hevner, A. and S. *Chatterjee, Design Science in Information Systems Research. Theory and Practice, O.S.* University, Editor. 2010, Springer-Verlag: Stillwater, USA.