# SpaceESB - A Proposal of an Enterprise Service Bus for Spacecraft Conceptual Design

2 authors:

Ariana Cristina Caetano Souza
National Institute for Space Research, Brazil
**3** PUBLICATIONS   **1** CITATION

SEE PROFILE

Walter A. Dos Santos
National Institute for Space Research, Brazil
**45** PUBLICATIONS   **66** CITATIONS

SEE PROFILE

Some of the authors of this publication are also working on these related projects:

Project  CTEE - Capacitação Técnica em Engenharia Espacial View project

Project  UbatubaSat View project

# SpaceESB – A Proposal of an Enterprise Service Bus for Spacecraft Conceptual Design

Ariana C. Caetano De Souza, Walter A. Dos Santos
INPE – National Space Research Institute, São José dos Campos, Brazil
+55 12 3208 6622
acaetano@img.com.br, walter@dss.inpe.br

**Abstract.** The conceptual design of space systems may become extremely complex and usually demand a multidisciplinary team working together in order to effectively coalesce into a successful architectural design. However, due to pressure on cost and time, many space projects nowadays demand a cloud enabled environment as they undergo their development divided among several project partners, sometimes temporally and geographically separated hampering the information exchange and causing adding risks to its conception. This paper proposes an enterprise bus service (ESB), named SpaceESB, to realize this distributed environment by exposing the various processes of space mission conceptual phase using a Service-Oriented Architecture (SOA) paradigm. As an example, a simple service related to propellant mass estimation is characterized by using BPMN (Business Process Modeling Notation) and Business Process Execution Language (BPEL). Its SOA materialization is implemented via a Java Web Service and its WSDL (Web Service Definition Language) provided as an interface to the SpaceESB.

## Introduction

Recent interest in collaborative environments for enabling cloud enabled systems engineering has raised in the literature [8]. Space systems engineering demand this environment as it requires essentially multidisciplinary expertise ranging from onboard computing to launcher vehicle interfacing which sometimes are not temporally and even geographically co-located.

Factors such as time zone differences, language barriers, numbering and units of measurement conventions, and different IT platforms can all have an adverse effect on the project timeline and budget.

In parallel, the growing demand generated by clients on services provided by space systems increases expectations for good space system engineering on delivery time reduction, higher quality and performance as well as cost reduction [1].

As space system designs are growing more complex, technical issues related to concepts development, also referred as Phase A, become more important and difficult. Conceptual design maps client needs to product use functions and is where functional architecture (and sometimes the physical architecture) is decided upon.

Typically, the design specifications and constraints impose a heavy burden on systems-of-systems engineering and particularly on requirements engineering which drives the whole system's life cycle. Henceforth, taking suitable decisions at this project phase ends up paying dividends on schedule, performance, cost and risk. Therefore agility and flexibility in the execution of intrinsic processes are necessary.

This highly-coupled and distributed scenario can be tackled thanks to the availability of open

standards to reduce barriers between different platforms and infrastructure to support the creation of services. This abstraction is possible via SOA and web services [3] which are being adopted to make business processes more efficient and effective. These technologies contribute to shape business processes, create solutions, design, develop and deliver services.

This work aims at creating an enterprise service bus [9], named SpaceESB that will provide a set of services to support the conceptual design phase of a satellite project allowing systems engineering partners to access them regardless of platforms. An example of a simple SpaceESB service, a propellant mass budgeting operation, is presented from its service modeling through its WSDL [9] interface exposure to the bus.

This paper is organized into the following sessions: (1) A brief introduction to SOA, ESB and web services; (2) The concept of SpaceESB; (3) The creation of a propellant mass budgeting service; (4) Future work and; (5) Conclusions.

## Background on SOA, ESB and, Web Services

As organizations grow, they acquire an ever-increasing number of applications distributed across a number of departments and sites. At the same time, it is becoming increasingly important for organizations to efficiently share information between these applications. SOA arose out of this need to allow intercommunication between applications [8]. SOA entails developing loosely coupled interfaces to applications (services), which can be used to access the applications and exchange information. By combining these services, it is possible to develop adhoc applications (mash-ups) as required.

**SOA.** Over the years, companies have been increasingly investing into information and communication technologies (ICT). For this reason, companies have many legacy applications which cannot be discarded and need to be adapted and integrated into the new solutions. To meet these demands, the concept of SOA has become quite appealing.

SOA is a software architecture style whose basic principle advocates that the functionalities implemented by the applications should be delivered as services enabling greater reuse, redundancies reduction and, greater efficiency in maintenance [3]. Frequently these services are organized through a "service bus" that provides interfaces, or contracts, accessible via web services or another form of communication between applications. The SOA, shown generically in Figure 1, is based on principles of distributed computing paradigm and uses the request / reply to establish the communication between client systems and the systems that implement these services [12].
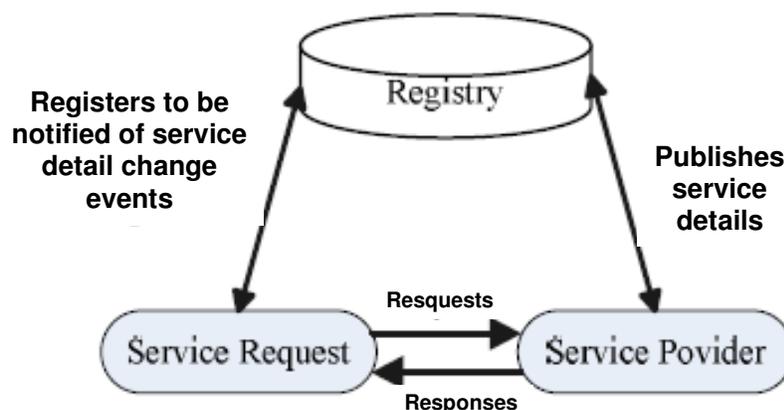


Figure 1. Basic Schematic SOA Environment

SOA is based on three fundamental elements, namely:
- Services - represent the business functions and are technology-independent;
- Interoperability - obtained through an infrastructure called Enterprise Service Bus - ESB, which combines the distributed services on different technologies and platforms;
- Loose coupling - is related to reduction of dependencies between systems, hence it is possible to achieve greater fault-tolerance and reduction on change impacts.

Therefore, SOA is not a COTS tool but rather an approach that when applied to business provides greater scalability, flexibility and growth, also serving as a bridge between business and ICT.

**ESB.** An ESB is an infrastructure that enables high interoperability between services, allowing exposed services to be called by clients. Its layout is sketched in Figure 2.
The main responsibilities of an ESB are: (1) Data transformation; (2) (Intelligent) routing; (3) Dealing with reliability; (4) Service management; (5) Monitoring and logging; (6) Providing connectivity and; (7) Dealing with security, among others.
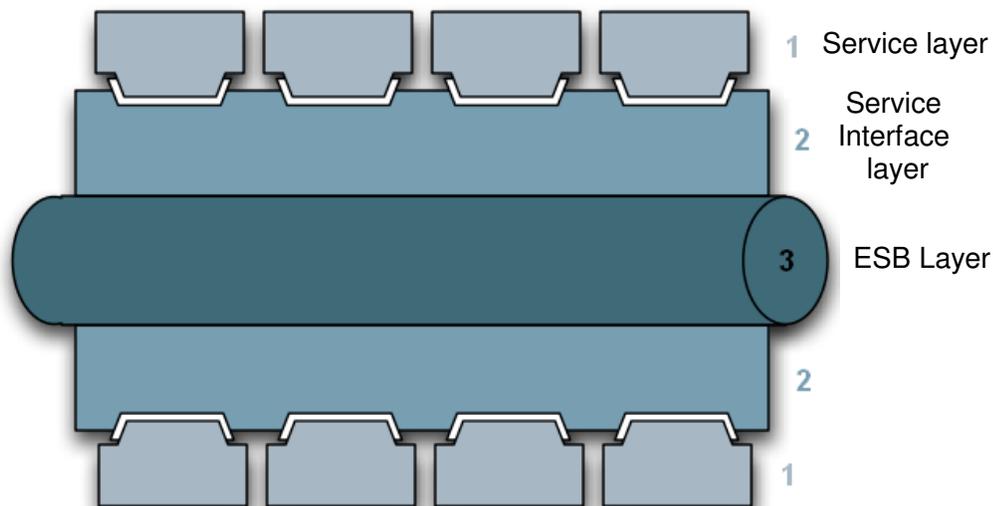


Figure 2. Layout of an Enterprise Service Bus [6].

A typical ESB structure is composed of these 3 layers:
- Layer 1:  It is the service layer that exposes functionalities and/or business rules of existing systems;
- Layer 2: Represents the interaction of services with the ESB for collaboration with other services hooked to the bus. If applied to web services, this layer could be, as example, SOAP [10] or REST [9];
- Layer 3: It is the service bus, or ESB, which mediates the communication between services.

**Web Services.** The web services are one of the possible ways of realizing the SOA abstraction and they are used in systems integration and communication between different applications. They can integrate applications written in different languages and running on different platforms through messages based on XML (eXtensible Markup Language).
The exchange of information held within web services uses a communication channel, usually,

the HTTP (Hypertext Transfer Protocol) protocol. There are many types of web services, but the most known and well-used are: RPC, WS-* (WSDL / SOAP) and REST.

The WS-* architecture is the mostly used nowadays and is composed of more than 20 specifications. The main specifications for this architecture are SOAP (Simple Object Access Protocol), WSDL (Web Services Definition Language) e UDDI (Universal Description, Discovery and Integration) [7].

In the WS-* architecture messages exchanged between client and service must be stored in envelopes that use SOAP, responsible to define the message format between applications.

The service description and how to access it is defined by a document that uses the WSDL language. The registration of a service and its location is defined by the UDDI.

Therefore, a web service needs to be published in a place available to all clients through the UDDI. A client would seek this repository for the desired service and then obtain the necessary data to invoke the service through the WSDL document. Thence, as Figure 3 illustrates, client and service can exchange messages enveloped on the SOAP protocol and transmit data represented in XML.
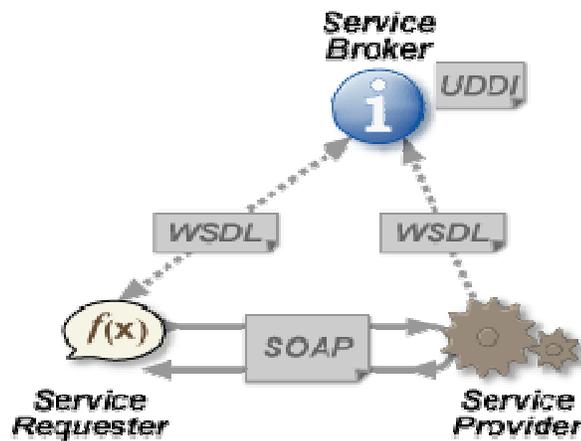


Figure 3. Basic scheme of web services and its messaging system

## The concept of SpaceESB

Some systems engineering activities for space systems are becoming more complex, decisions made in defining suitable architectural designs have a highly coupled nature. The conceptual design of a space mission requires that engineers from various disciplines to work together. However, in many cases, these engineers do not always share the same physical nor even their geographical space, hampering communication and creating delays on decision making which is crucial for achieving good results, lowering costs and meeting deadlines. This scenario indicates a need for systems engineering tools to aid in the conceptual designs trade-offs.

This work proposed the creation of an ESB dedicated for this part of the project life cycle. Based on a concurrent engineering approach the SpaceESB provides the information related to the conceptual design would be available as services which could be invoked from anywhere and regardless the environment. This promotes team communication as impacts of the decisions taken for one team are rapidly evaluated and all of the subsystem requirements and constraints are considered simultaneously by the other teams involved precluding misconceptions.

## Process Modelling for the SpaceESB

One of the first steps for SOA deployment is investigate the underlying business processes and perform a consistent modeling in order to ensure that the business rules are really being satisfied.

BPMN [11] is a part of a new generation of languages and has been established as a standard for handling business processes [4] as it provides the necessary notation to express the business processes by employing a BPD (Business Process Diagram).

As BPMN is a graphical language, it allows the many stakeholders to easily understand the process behavior. BPMN is based on flowcharts to perform a business process representation. One business process model is thence a graphical network of objects which represents flow control defining the execution sequence of each activity model [4].

The Figure 4 shows the high level BPD of a satellite conceptual design which emphasizes the flow elements, their inter-dependencies and how they are executed. The whole process is divided into 3 domains: (1) Definition of mission objectives; (2) Mission characterization and; (3) Mission concept evaluation and selection. Each domain is further detailed into a set of processes that compose and interlinks them.
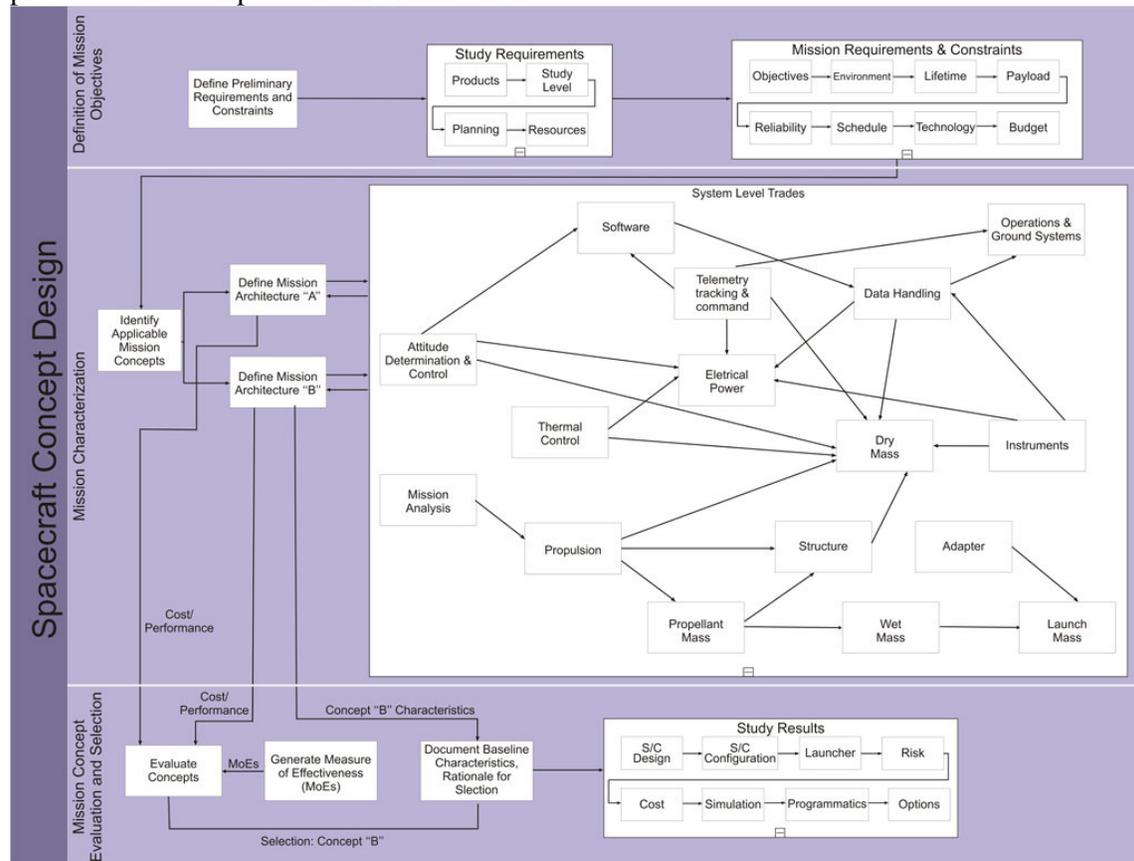


Figure 4. The Business process diagram of a satellite conceptual design

For development purposes, all processes will have to be mapped into services via modeling up to its WSDL interface exposure to the bus. This is detailed in next subsection.

## Preparing for process automation

After the diagram construction, its flow bas be prepared in order to map each flow activity into a correspondent service, in other words, identify what activities will be provided as services

which may be independently executed, generating well defined results.

In this paper, just a few business model activities will be approached and will be part of the enterprise service bus. Only the business model activities dedicated for the mission characterization domain are depicted in high-level at Figure 5 as a portion of the SpaceESB. From these activities, only the propellant mass activity will be further detailed until service implementation level later on.

The SpaceESB implementation as well as all the modeling tools is based on open software solutions.
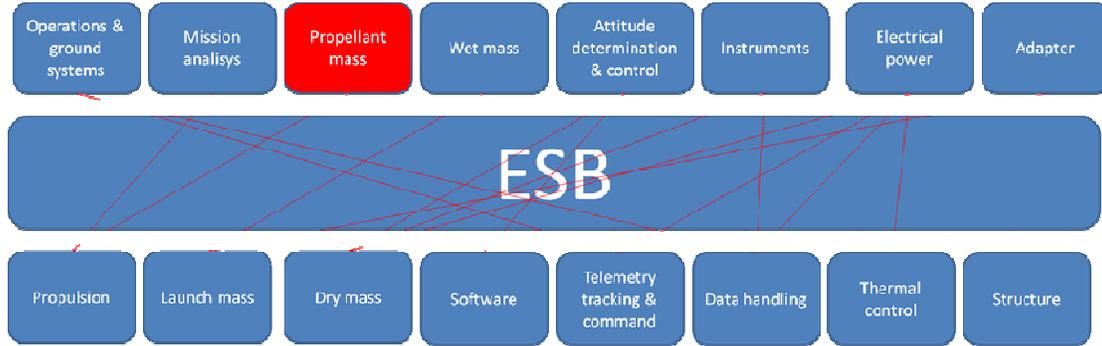


Figure 5. Portion of the proposed SpaceESB in high-level and services dependencies

Each one of these business model activities have an algorithm of their own which encapsulates the underlying business rules of the company developing their products. This is a key artifact in the Knowledge Management policy of companies and considered a real asset.

## Creating a Propellant Mass Service Plug-In to SpaceESB

Business functionalities are mapped into services and they are composed by parts, named operations, which encapsulates the existing business rules complexity. The service operations, as well as the business process, need also be modeled with BPMN.

Typically, in the conceptual design phase of satellites suitable architectures are sought that matches successfully mission objectives [5]. One of the aspects to be dealt with is the propulsion subsystems which derives its requirements from a mission analysis service and drives the next service which is estimating propellant mass. The enchaining of information in the propellant mass service is implemented via exchanges of envelopes in a SOA abstraction as shown in Figure 6 with its existing operations and flows in high-level.
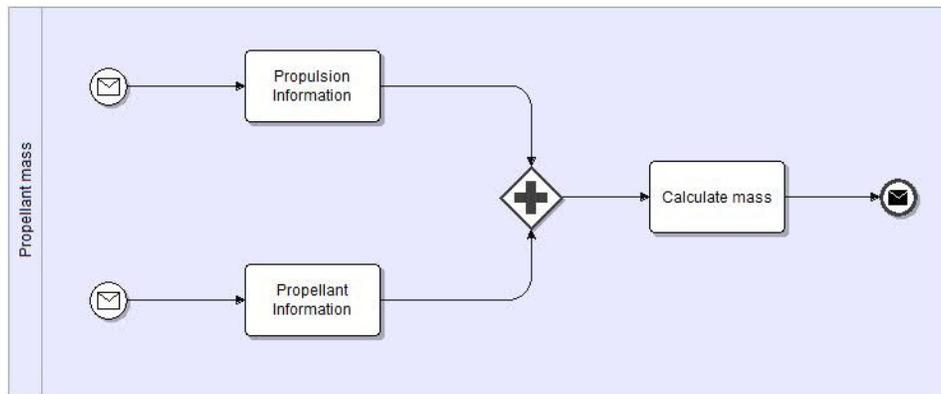


Figure 6. High-level flow diagram of propellant mass service operations

From the diagram construction is one may realize the difference between services and operations as the later are parts of a work provided by a service which by itself does not generate any much useful information.

## *The Propellant Mass Service Implementation*

The realization of a SOA abstraction, as like any software implementation, needs a set of tools and a development environment in order to create its business models, its services and, its overall derived elements.

After the service definition and operations models, one has to perform the transposition of the models to computer systems, in this case to a web service. For this work, the programming language chosen is Java and the development environment chosen is Netbeans, a free integrated development environment (IDE) which has SOA plug-ins resources for service creation and orchestration.

The first implementation step is the creation of the Propellant Mass web service, containing the previously modeled operations in BPMN. At the end of the web service creation, a WSDL file is also generated, illustrated in high level at Figure 7.

```xml
<?xml version="1.0" encoding="UTF-8" ?>
- <definitions
  - <portType name="PropellantMass">
    - <operation name="PropulsionInformation">
        <input message="tns:PropulsionInformation" />
        <output message="tns:PropulsionInformationResponse" />
      </operation>
    - <operation name="PropellantInformation">
        <input message="tns:PropellantInformation" />
        <output message="tns:PropellantInformationResponse" />
      </operation>
    - <operation name="CalculateMass">
        <input message="tns:CalculateMass" />
        <output message="tns:CalculateMassResponse" />
      </operation>
    </portType>
- <service name="PropellantMassService">
  - <port name="PropellantMassPort" binding="tns:PropellantMassPortBinding">
      <soap:address location="http://localhost:8080/PropellantMass/PropellantMassService" />
    </port>
  </service>
</definitions>
```

Figure 7. High level WSDL file for the Propellant Mass service.

The out coming WSDL file contains details the service description, its operations and the access conditions. Any further implementation details were omitted from Figure 7.

The second implementation step is realizing the underlying service orchestration. The web service creation only implements the service operations, but is essential implement also the operation execution flow.

The orchestration is responsible for defining the services invocation sequence and conditions. BPEL (Business Process Execution Language) is the chosen strategy for service orchestration [2]. BPEL is an XML dialect that defines services interactions. Figure 8 shows the BPEL diagram for the Propellant Mass service.
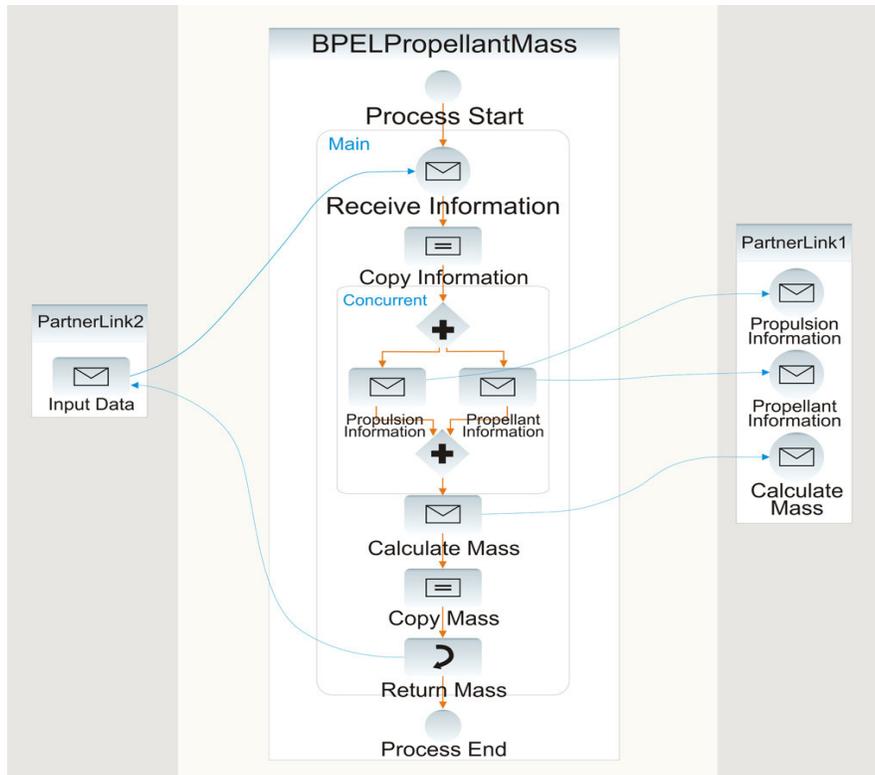
Figure 8. Propellant Mass BPEL diagram.

After this step, service creation is complete and it can be coupled to the SpaceESB which makes it available for all project partners who need that information.

The creation methodology used here can be extended to any other services component of the ESB.

# Further Work

This work just prototyped a simple service and the surrounding enchainment of information of its neighboring services on the SpaceESB. Further work will effectively add more services to the bus and field trials will be performed. Also relevant are issues related to more specific operating details such as: routing; reliability; service management; monitoring and logging; connectivity and; security.

# Conclusion

Due to its very complex and multi disciplinary nature, the flawless design, construction and launching of space systems, like a satellite, is quite demanding. Experts from many fields are required to cooperate for ensuring the project success. Sometimes these tasks are geographically and/or temporally distributed demanding a cloud enabled environment.

In order to support the conceptual design phase of a satellite project, this paper proposed the SpaceESB, a SOA solution for an enterprise bus service (ESB) that exposes as services, processes pertaining to the design of a space mission.

The mapping of a process to its Java web service which implements these services was present using as an example, a simple service related to propellant mass estimation by using BPMN and BPEL. Using open source tools for all steps, a WSDL file was generated which describes the interface to the SpaceESB.

Using this abstraction, project partners may actively participate on the decision-making

process having greater agility and flexibility and coping with pressures on costs and time.

# References

1. Bandecchi M.; Melton B.; Ongaro F. 1999. Concurrent Engineering Applied to Space Mission Assessment and Design. In *ESA Bulletin*.

2. BPEL tutorial. SearcSOA.com
   `http://searchsoa.techtarget.com/generic/0,295582,sid26_gci13 30911_mem1,00.html?ShortReg=1&mboxConv=searchSOA_RegActivate _Submit&`

3. Erl, T.; 2004. *Service Oriented Architecture: A Field Guide to Integrating XML and Web Services*, The Prentice Hall.

4. Grosskopf, A.; Decker, G.; Weske, M. 2009. *The Process: Business Process Modeling using BPMN*. Meghan Kiffer Press.

5. Larson W. J. Wertz J. R. 1991. *Space Mission Analysis and Design*, second edition.

6. Longo J. S. C. SOA com OpenESB. Sun Microsystems.
   `http://api.ning.com/files/Sn6i83r-auY8oBXNNYPn61ZsEMqGd3sU-4 21*dPFgL0a9aX*MnF0rHU0uZzcxDGOxOq8WCIDX5pk6XchShvtTKV*Y-L3lf Ec/SOAcomOpenESB.pdf`

7. Moro, T. D.; Dorneles, C. F.; Rebonatto, M. T. 2009. Web services WS-* versus Web Services
   REST.`http://www.upf.br/computacao/images/stories/TCs/arquivo s_20092/Tharcis_Dal_Moro.pdf`

8. Ross, J. and et al. 2006, *Enterprise Architecture As Strategy: Creating a Foundation for Business Execution*. Cambridge, Harvard Business School Press.

9. Shin S. SOA using Open ESB, BPEL, and NetBeans . Sun Microsystems.
   `http://docs.huihoo.com/openesb/soabpelopenesb.pdf`

10. W3C. 2000. Simple Object Access Protocol (SOAP) 1.1: W3C Note 08 May 2000.
    `http://www.w3.org/TR/2000/NOTE-SOAP-20000508.`

11. White S. A. 2004. Introduction to BPMN. IBM Corporation. BPTrends, July, 2004.
    `http://www.bptrends.com/publicationfiles/07-04%20WP%20Intro% 20to%20BPMN%20-%20White.pdf`

12. Wu Q.; Zhou C., Jing T. 2008. Research on SOA based framework of collaborative design system for complicated product. In *International Conference on Computer Science and Software Engineering*.

# Biography

**Ariana C. Caetano De Souza** is an MSc student in Space Engineering and Technology at INPE, the Brazilian Space Research Institute in Brazil. She works as a Systems Analyst for Imagem, a GIS-based enterprise. She holds a BSc. on Information Systems, 2009.

**Walter A. Dos Santos** is a System Engineer at Space System Division and a Lecturer at the Space Technology and Engineering and Applied Computing Postgrad. at INPE. He holds a PhD on Computer Science, 2008, a MSc. on Control and Systems, 1992 and a BSc. on Electronics Engineering, 1987. He is a member of the IEEE Computer Society, the ACM and of the Brazilian Computer Society.