An Approach for Distributed Objects Applied to Satellite Simulator

Burgareli L. A.; Ferreira M.G.V.; Nakanishi T. (INPE)

1 – Introduction

The purpose of this paper is to present a study of a distributed object application to the Satellites Simulator, present in the National Institute for Space Research – (INPE), Satellite Control Center, in an attempt of adjusting it, to the trends of technological evolution. With the intention of improving this application, techniques for a better modeling of the distribution of these objects are also studied.

The Satellites Simulator is a software with the purpose to train operators in its control and tracing, and to provide a real environment to perform the tests of acceptance of the software of satellites system control. Since it operates in a centered system, the Satellites Simulator presents some limitations, related to availability, fault tolerance, interoperability, flexibility in taking care of the different situations of control, that turn out to be obstacles to the performance of the simulation process. It is observed that many of this limitations can be solved or minimized through the existing features in the distributed object paradigm.

Distributed object represents the evolution of the conventional object. It has an environment controlled by middleware, providing advantageous properties as distribution, transparency, fault tolerance, availability, concurrency, and better performance. To conserve these same features, in a client/ server environment, without distribution of objects, significant efforts would be demanded to implement and control the many necessary interfaces for the interconnection of these systems.

In the last years, companies of software are offering in the market, some new standards to be applied to the distributed object systems. These standards become, each year, more enclosing and efficient. However it is observed a deficiency in the aspect of how to model correctly this distribution, in other words, how to distribute the objects according to the application features and the user necessities, exploring more actively this technology advantages in order to reach a better quality.

Besides exploring the features of the objects distribution applied to the Satellites Simulator, it is intended to construct an analysis, through the modeling of distribution of objects, observing the necessities and aspects presented by the application. Then, some techniques are proposed, for example: models based in Use Cases, fault Tolerance and random form, that establish criteria and different forms of how to distribute better involved objects in the Satellites Simulator. The initiative does not have as an objective to point what is the best technique to be used. The efforts will be concentrate in raising the advantages and disadvantages of each technique, exploring qualitative and quantitative factors involving performance, availability, data traffic, etc.

So, the foreseeing of the behavior of the objects in the application, examining important individual factors or when related to the other objects and in different aspects, can bring a contribution for a better organization of them in the Satellite Simulator. It is expected that this analysis points to the necessity of modeling correctly the objects distributed, so that it gets important profits regarding the efficiency of the application.

2- The SICSD Architecture

The SICSD architecture purpose consists in presenting a distributed application, applied to the INPE satellite control software, where the objects are presented in a dynamic way, that is, they can migrate from a *node* to another, adapting themselves to the service solicitations, and improving a number of characteristics, such as: performance, flexibility, reliability and the use of available computer resources.

The SICSD architecture presents the application objects, for example: Telemetry, Remote Control, Station, Ranging and Equipment.

The next figure presents an overview of the SICSD architecture and its available services:

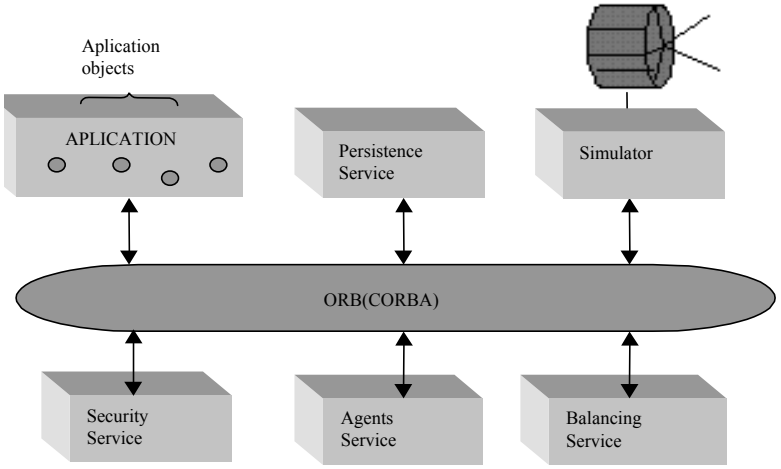


Figure 2.1 – An overview of the SICSD services.
Source: Ferreira

- *Agents Service*: The instanced object physical localization, the availability or the non-availability status of the object in order to receive the system service solicitations and the CPU availability of a determined node are some of the information generated by the service agents in order to monitor the object status and the network status as well, keeping the necessary information distributed and updated in order to develop a dynamic and flexible application for satellite control.
- *Balancing Service*: This service is capable of detecting the configured environment critical points for the satellite control, that is, they identify the nodes that have a work overload, making decisions that may result in migration or in the replication of the application objects for the satellite control. Therefore, it is the responsibility of this service to provide mobility, flexibility and dynamism of the proposed architecture. For example, if there are a large number of instanced objects at a determined node, the balancing service is responsible for the object migration of saturated machines to free machines; or even, if instanced objects at determined

nodes are receiving several solicitations from other network nodes, the balancing service should replicate the saturated node objects into free nodes.

- *Persistence Service*: the purpose of this service is to store/recover the application persisting objects. The persisting object store methods for the application do not directly access the data store system; they simply access the persistence service to do this operation.
- *Security Service*: it is responsible for guaranteeing the system access only to people previously authorized, restricting them to the previously defined functions, according to its profile. As the SICSD allows that the objects migrate from a machine to another; security mechanism should also be used to guarantee that only the authorized objects could migrate to remote machines.
- *Satellite Simulator*: object responsible for simulating the possible satellite status, that is, the possible internal conditions that it presents at a determined moment.

3- Elimination of the Satellite Simulator Limitations

The Satellite Simulator, as already mentioned, allows the training for the people working in the operation, and it acts as a base for accepting tests, operational procedure validation and satellite control [16]. However, some limitations related to the system availability, fault tolerance, inter-operation, flexibility to attend different control situations, etc. are presented by the Satellite Simulator, coming from the centralized system upon which they act.

These limitations eventually create obstacles for the performance of the whole process. These obstacles have the characteristics presented in the distributed object paradigm. And also that working in a distributed environment, which has a better result of the application performance, is now not only a forecasted tendency in the computer world but a reality [8].

Therefore, it is understandable that it is possible to eliminate the limitations described above, using the distributed computing resources, through its innovating properties. Thus, it is possible to resolve the mentioned problems through the following advantages provided by the object distribution:

- **The increase of service availability**: The object-distributed technology provides a mechanism that assures the object availability, independently from computer failure [4]. This characteristic finds the solution to the current Satellite Simulator availability, increasing its service availability.
- **Fault tolerance**: a failure of a computer or object, in a distributed environment represents only a partial system failure, which can be overcome if there is a node failure, where a determined object is instanced. A new connection can be established with another object that can work similarly. This capacity resolves the Satellite Simulator limitation of not being able to tolerate failures.
- **Interoperability**: This important characteristic allows the distributed object other methods, even if platforms, operational system and programming language are heterogeneous, which can resolve the restrictions imposed by the existing heterogeneity in the Satellite Simulator. The interoperability a significant property that distributed structures must assure, since not all the computer set need to use heterogeneous platforms [7].

- **The increase of concurrency and performance**: The capacity to instance copies of the same object in different machines, provides a better answer to the multi-user requirements. A factor which should be taken into account to exploit the concurrency resource in distributed systems, that is, two or more users can request the same service from the system, but being instanced at different nodes.
- **Flexibility to attend the different control situations**: with the application of distributed objects in the Satellite Control System, we can distribute parts of the simulating process , or even totally replicate them in distinct machines. Therefore, there is greater machine availability , increasing the limit number of users who need to use the satellite simulator concurrently.

4- Modeling for the Distribution

During the last years, developing software companies are launching into the market several patterns to be applied to the distributed object systems. With relation to the object distribution, these patterns are becoming more including and efficient.
Thus, the distributed computing is rapidly gaining more importance in the computer field, however, there are a few tools available to design and model the distributed object system.
A deficiency is noted in the process of correctly modeling this distribution, that is, how to improve the distribution according to the application characteristics and the user's necessity, exploiting more actively this technology advantages and obtaining a better quality this way.

4.1- UML

The UML (Unified Modeling Language) is a language to specify, to visualize, to construct and to document the software system devices, and represents a collection of the best engineering practices that were successfully used in the modeling of a complex system [10]. It is possible to use it to show the limits of a system and its main functions; to illustrate the realization of these use cases with interaction diagrams; to represent the system static structure using class diagrams; to model the object behavior through status transition diagrams; to reveal the physical implementation architecture with component and application diagrams; and, finally, to extend its functionality through stereotypes.

A use case is a typical interaction between a user and a system, a specific way of use from a segmented functionality point of view [2].

It is possible, for example, to segment a satellite simulator subsystem in which the user, according to the UML concept, actor, interact in a specific way with the activities described in the figure 4.1:

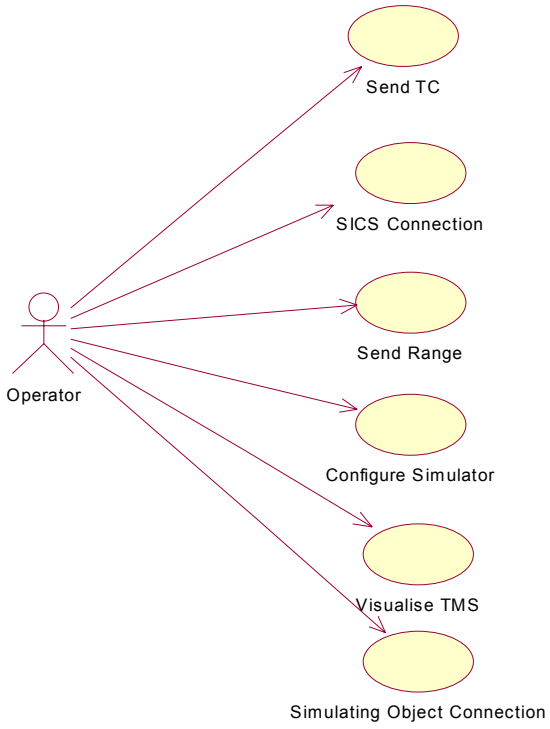


Figure 4.1 – The Use Case Diagram – Subsystem Simulator

4.2- Distribution Modeling Based on Objects that Collaborate to Accomplish a Use Case

In many applications, at determined moments, we can note an interdependency that exists among the objects that should not be broken. That is what occurs with some remote control and telemetry objects.

The Telecommand object represents messages that can be sent to the satellite in order to correct or change switch positions, turn on or off sensors, the telemetry object shows the satellite internal status, voltage, temperature, that is, the satellite equipment conditions. A telemetry shows an execute or not of telecommand sent. We can observe then that frequently a telecommand object acts and affects the telemetry object, thus, if we let them work near each other or even at the same machine, it can bring some advantages, if they obey a determined criterion.

This criterion for the object distribution, can be obtained analyzing the Satellite Simulator Subsystems and making the objects that communicate most frequently among themselves participate in a realization of the same Use Case. In doing so, we decrease the relationships between the Use Cases and consequently among the machines, providing advantages, reducing the network traffic and obtaining a greater availability, like in the case

if, for example, a failure at a determined object of a Use Case occurs, this one will not damage a second case.

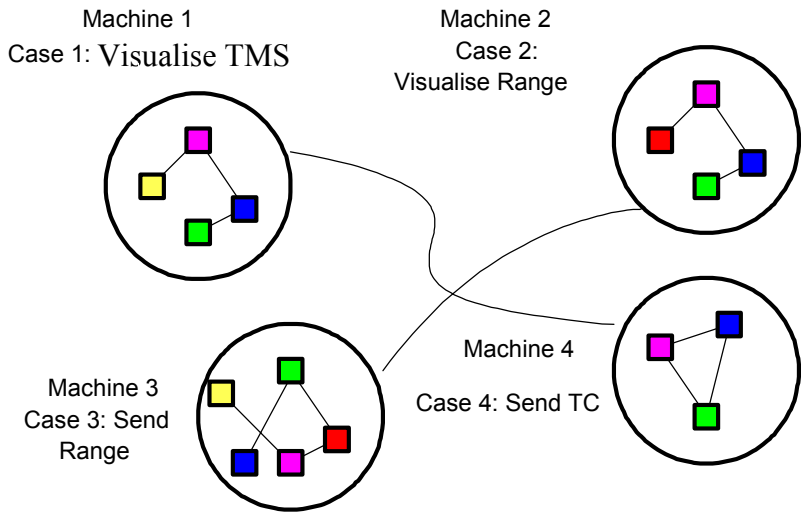
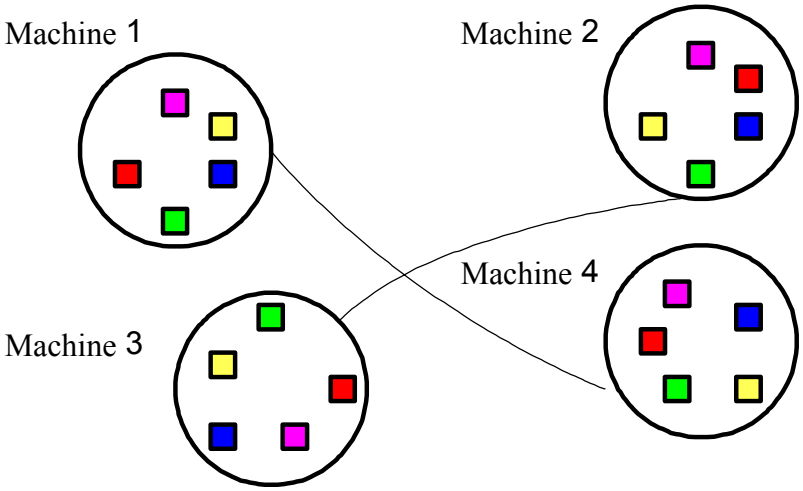


Figure 4.2 Distributed Objects by modeling based on objects that collaborate to accomplish a Use Case

There is no impediment for having a double Use Case at distinct machines, since it is a copy of a Use Case, the object set will provide the same characteristics as the one from the origin Use Case. In the figure 4.2, we observe the objects that collaborate to accomplish the Use Case. The circumference is delimiting this collaboration.

4.3-Distribution Modeling Based on Fault Tolerance

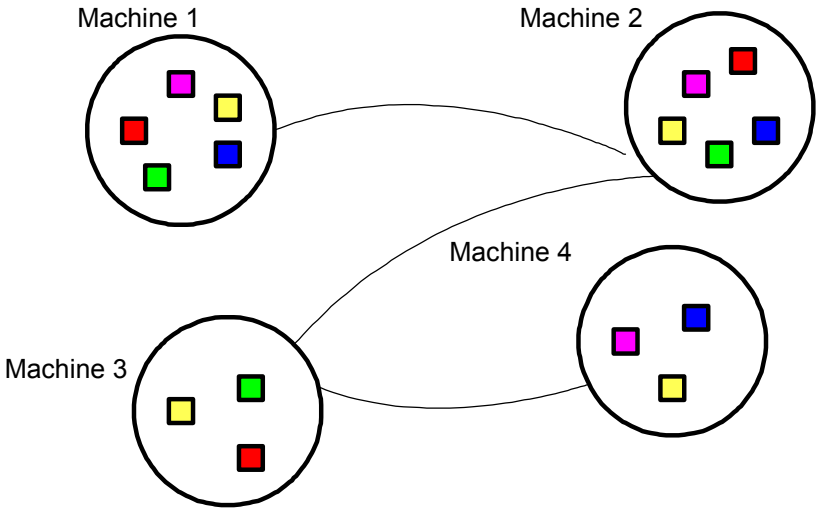


The figure 4.3 - Distribution Modeling Based on Fault Tolerance

A second aspect to be dealt with, when we try to find modeling options for a better distribution, is to provide object fault tolerance. Focusing on this property, the new established criterion is to replicate the objects independently from the service being executed at all the machines that exist on the system. In this case, the machine overload will be unavoidable, however we strongly assure the availability of a determined object. The figure 4.3 shows how this system functions.

4.4- Distribution modeling base on a random way

The third way presented consists on a random modeling. This technique is equivalent to the traditional method, excepting the fact that it obeys a single criterion imposed by the modeling, which establish the existence of at least one copy of the same object on the system. The following figure shows this distribution:



The Figure 4.4 – Distributed objects by random modeling

4.5- Modeling based on a dynamic distribution

Another modeling presented is based on a dynamic distribution. In this distribution the objects can migrate or replicate automatically from a machine to another. The Balancing Service, already mentioned, executes this activity, which is proposed by the SICSD. The SICSD does it according to the CPU availability or the number of connections between the objects and the machines.

It is this service responsibility to guarantee the mobility resources, the flexibility and the dynamism of the proposed architecture. For example, if there is a large number of instanced objects at a determined node, the balancing service is responsible for the object migration from saturated machines to free ones; or even, if instanced objects at determined node are receiving several service solicitations from other network nodes, the balancing service should replicate the objects from saturated nodes to free nodes.

The objective of this modeling is concentrated in the observation of how the dynamic distributed objects behave.

We also intend to develop a comparative analysis of the performance of the modeling processes presented.

5- Prototype Implementation

A Satellite Simulator prototype is already being implemented, using for the object distribution, The Java RMI (Remote Method Invocation) standard, from the SunMicrosystem. With the first Satellite Simulator Subsystems developed, it is possible to observe the object relationships, as it happens with the activities that transmit the Remote Control and the ones that receive the Telemetry.

The environment used is composed of 5 computers, at which the objects are distributed with the help of a scenario manager. This scenario manager consists of a software also designed for this environment, where the objects are created or destroyed at the available machines according to the established modeling type (failure tolerance, random, by use case, etc). Through it, we can also manage the CPU availability and the number of connections between the machine and the objects.

In a dynamic environment, activities like the migration and replication of objects have already been developed. These activities, being part of the Balance Service, proposed by the SICSD architecture, make possible the observation of the objects behavior in this environment.

6 – Conclusion

By employing objects distribution to the Satellite Simulator subsystems, the purpose is to provide it with the properties that will eliminate several limitations existing in the present time, relating to availability, transparency, failure tolerance, etc.

Some situations are not so advantageous, though, like performance interference and machine overload, which are due to the distribution, and they must be anticipated. These possibilities must then be analyzed and also characterized as a contribution to the study to be carried out. Thus, by the introduction of these new concepts, the Satellite Simulator system behavior analysis will result in significant data that will help improve the study of the objects distribution in this application.

The innovative characteristics provided by the use of distributed objects will make this technology a promising one. Nevertheless, it is important to remember that, in order to use this method, knowledge is required about the technique, analysis and the use of rules so that the project will not become confused and inefficient.

Nowadays, when we decide to use distributed objects, we usually do not considerate any method concerning the best use and organization for the distribution of these objects to the machines in the application. As a whole, the companies provide only the standards for the distribution, and do not present the modeling techniques for this purpose.

A modeling technique for the better distribution of the objects is then conceived, and it uses the basic assumption that, by anticipating their behavior in the application,

examining important factors, either individually or relating to other objects and in different aspects, may bring contributions to an improved organization of these objects on the system.

Therefore, besides exploring characteristics of object distribution, the intention is to build up an analysis through the object distribution modeling, noting the user needs and aspects of the application. For this purpose, some techniques that will establish criteria and different methods of modeling will be proposed. This initiative is not intended to point the best technique to be used; the efforts will be focused in delineating the advantages and disadvantages of the techniques, exploring qualitative and quantitative aspects and classifying them according to the results obtained in a comparative study that will involve performance, availability, traffic reduction, and so forth.

The choice for the UML in this study consists in following widely known standards for working with objects. One of the benefits of the UML is to facilitate the analysis, once this language makes possible a concise study through different approaches, and in a well-designed way.

Thus, we expect this analysis to point to the necessity of correctly modeling the object distribution in order to obtain important gains in the application efficiency.

7- Bibliography

[1] - Ahuja, S.; Quintao, R.; *"Perfomance Evaluation of Java RMI: A Distributed Object Architecture for Internet Based Application"*, IEEE, 2000.

[2] – Butler, J. M. ; *"Quantum Modeling of Distributed Object Computing"*, IEEE, 1995.

[3] - Costa, S. R.; *"Objetos Distribuídos: Conceitos e Padrões",* Dissertação de Mestrado, Computação Aplicada, INPE, 2000.

[4] - Ferreira, M. G. V. *"Uma arquitetura flexível e dinâmica para objetos distribuídos aplicada ao Software de Controle de Satélites",* Tese de Doutorado, Computação Aplicada. INPE. 2001.

[5] – Fulan, J.D. – *"Modelagem de objetos através da UML"*, Makron Books, São Paulo, 1998.

[6] – Kalogeraki, V.; Moser, L. E. ; Melliar-Smith, P. M. ; *"Dynamic Modeling of Replicated Objetcs for Dependable Soft Real-Time Distributed Object System"*, IEEE, 1999.

[7] - Kono, K.; Masuda , T.; *"Efficient RMI: Dynamic of Object Serialization"*, IEEE, 2000.

[8] - Mainetti, S, Jr.; *"Objetos Distribuídos"*, Visionnaire, Curitiba, junho de 1997.

[9] – Mowbray, T. J.;Ruh, W. A., *"Inside Corba - Distributed Object Standards and Applications"*, 1997.

[10] - OMG. – *"What is OMG-UML and Why is it important ?"*, junho, 2001. http://www.omg.org/gettingstarted/what_is_uml.htm

[11] – Orfali, R.; Harquey, D.; Edwards, J.; *"The Essential Distributed Objects Survival Guide"* , 1996.

[12] - Pooley, R. ; King, P. ; *"The Unified Modeling Language and Performance Engineering"*, Department of Computing And Electrical Engineering, Heriot – Watt University, Riccarton, Scotland, 1999.

[13] - Rozenfeld, P.; Miguez, R.; Orlando, V.; *"Proposta de um Simulador para o Satélite SCD1"*, INPE, 1990.
[14]- Sessions, R.; *"Ten Rules for Distributed Object Systems"*, OS/2 Magazine, October,1996.
http://www.objectwatch.com/articles_by_staff.htm
[15] - Yamaguti, W.; Orlandi, E.F.E; Orlando, V.; *"Manual do Usuário do Software Simulador do Satélite SCD1"*, INPE, 1994.
[16] - Yamaguti, W.; Orlandi, E.F.E; Orlando, V.; *"Documento de Projeto Preliminar do Sistema Simulador do Satélite SCD1 - SIMS"*, INPE, 1994.