Available online at www.sciencedirect.com

ScienceDirect

Simulation Modelling Practice and Theory 15 (2007) 716–733

SIMULATION MODELLING PRACTICE AND THEORY

www.elsevier.com/locate/simpat

Development of distributed real-time simulators based on CORBA

Manuel Díaz, Daniel Garrido [*], José M. Troya

Department of Languages and Computing Science, University of Málaga, 29071 Málaga, Spain

Received 22 December 2004; received in revised form 12 July 2006; accepted 17 March 2007
Available online 30 March 2007

Abstract

The development of complex simulators with multiple simulation models running in a distributed way is a difficult task where communication plays a main role, particularly in cases where real-time constraints exist. This paper presents distributed simulators for nuclear power plants with communication based on Real-time CORBA, a middleware for real-time systems. The simulators are used to train power plant operators safely. They are based on a complex architecture of simulation models with real-time constraints involving many different applications that allow full scope simulation of the control room of a nuclear power plant. The use of Real-time CORBA in the simulators allows us to obtain platform and language independence, reusable components and the control of real-time properties.
© 2007 Elsevier B.V. All rights reserved.

Keywords: Distributed; CORBA; RT-CORBA; Real-time; Nuclear power plant; Components

1. Introduction

The constant development of new technologies, methodologies, tools and programming languages makes companies carefully consider the adoption of some of these new elements in their projects [1].

The Common Object Request Broker Architecture (CORBA) [2,3] is the core of the Object Management Architecture described by the Object Management Group (OMG) [2] to build distributed applications. Standard CORBA Object Request Brokers (ORBs) traditionally support best-effort capacities and do not allow guarantees regarding temporal application responses [4]. So real-time applications can not benefit from CORBA features such as interoperability, transparency, platform independence, etc. The OMG Real-time CORBA specification (RT-CORBA) [5,6] defines the features needed to obtain end-to-end predictability based on fixed priorities in real-time ORBs. Nowadays, there are not many large real-time distributed applications using RT-CORBA and most of these existing systems use traditional languages like Ada or C with *ad hoc* techniques for communication.

[*] Corresponding author. Tel.: +34 952132865.
E-mail addresses: mdr@lcc.uma.es (M. Díaz), dgarrido@lcc.uma.es (D. Garrido), troya@lcc.uma.es (J.M. Troya).

1569-190X/$ - see front matter © 2007 Elsevier B.V. All rights reserved.
doi:10.1016/j.simpat.2007.03.001

This paper presents the use of RT-CORBA in full scope simulations of the control rooms of nuclear power plants. The simulators are exact replicas of the control rooms, taking care of all details, from physical artefacts like furniture, control panels, etc. to software, simulating the applications running in the control room of the power plants. The kernel of the simulators consists of simulation models with real-time constraints, which provide the values of the distinct signals and variables needed by the other hardware and software components. The use of RT-CORBA has provided a suitable communication infrastructure which allows the real-time needs of the different simulation models and tools of the simulators to be taken into account.

The main purpose of the simulators is to train the operators of the power plant, enabling them to practice different situations, ranging from standard situations like temperature monitoring, valve manipulation, etc. to the most unusual, including emergency situations. The duration of some simulation sessions can last a few days, so the software developed and the hardware platform have to provide quality, stability, robustness, etc.

The software described in this paper has been developed in different joint projects between the company Tecnatom S.A. and the department of Languages and Computing Science at the University of Málaga. The work is related to the development of new software with special emphasis on communications based on RT-CORBA and building software components [7,8] that can be reused in future projects. Specifically, the software developed is being used in full scope simulators for nuclear power plants located in Spain, Germany and Mexico.

The paper is organized as follows: the rest of this section presents the main RT-CORBA features used in the paper. The hardware architecture is presented in Section 2. Section 3 presents the software architecture of the simulators. The remaining sections are about the main components and applications of the simulators and related RT-CORBA issues. The paper finishes with some conclusions and future work.

1.1. RT-CORBA features

CORBA is a communication middleware that allows the communication of objects developed in different programming languages and running on different hosts or operating systems in a transparent way [3]. These objects (*servers*) define interfaces with operations provided to the *clients*. The *clients* only use these operations and there is no difference between invocations to local objects and invocations to remote objects because all the communication details are managed by CORBA.

Temporal predictability is a main aspect in the development of real-time applications. However, standard CORBA implementations are not suitable for real-time because they only support best-effort capacities in the communications and there are no guarantees about the temporal response on particular invocations to remote objects. So the solution is to use ORBs supporting the Real-time CORBA specification. Real-time CORBA provides mechanisms that allow configuration and control of processor resources, communication resources and memory resources. The following points show the main RT-CORBA features used in the implementation of the simulators:

- **Native and CORBA priorities:** Real-time CORBA applications can use CORBA priorities that allow hiding the heterogeneity of native priorities in the different Operating Systems of a distributed application. RT-CORBA priorities can be specified with values ranging from 0 to 32767. These priorities are used in a platform-independent way.
- **Server declared and Client propagated priorities:** Two different policies are used to transmit priorities. In the SERVER DECLARED model, the server declares the priorities at which an invocation made on an object will execute. The CLIENT_PROPAGATED model allows the propagation of the client's priorities that must be honored by servers, avoiding priority inversions problems [9,10].
- **Thread pools:** The pools allow the pre-creation of threads in such a way that a thread manages each invocation on a particular object. This way, the cost and unpredictability of dynamic threads are avoided. The thread pools can contain static and dynamic threads and can be created with lanes with different priorities, allowing the redistribution of invocations depending on client priorities.
- **Mutexes:** RT-CORBA mutexes are the standard RT-CORBA synchronization mechanism that permits priority inheritance and priority ceiling protocols [9] if the underlying operating system supports them (e.g. POSIX [11]).