Towards a RT-Java Based Embedded Remote Monitoring Tool for Small and Medium Power Plant Units

L. Réveilleau[**], E. Becquet[**], L. Bacon[*], J-M. Douin[**], E. Gressier-Soudan[**], F. Horn[***]

[*]EDF, DRD/EP/CCC/GAS,
6 Quai Watier,
78401 Chatou Cedex France
laurent.bacon@edf.fr

[**]CEDRIC-CNAM,
292 rue St Martin,
75141 Paris Cedex 03 France,
{becquet,gressier}@cnam.fr

[***]KELUA Software,
9 chemin de la Brocardière,
69570 Dardilly, France,
francois.horn@kelua.com

Abstract: This work describes an on-going project at CNAM-CEDRIC. Its goal is to evaluate the ability of the Real-Time Java technology for process control applications with soft real-time constraints. We are designing a lightweight embedded remote monitoring computer prototype able to support Web based Man Machine Interface and production data exchanges between power plants and control centers.

1. Introduction

This paper describes an on-going project at CNAM-CEDRIC, the computer science laboratory of CNAM, a school of engineering in Paris. Its goal is to evaluate the ability of the Real-Time Java technology for process control applications with soft real-time constraints. Users' requirements come from a parallel project with EDF (Electricité de France), one of the world's leading energy group (www.edf.fr), whose aim is to gain experience with the TASE.2 (Telecontrol Application Service Element version 2) communication protocol [4], also called ICCP (Inter-Control Center Protocol). TASE.2 is an OSI based protocol ensuring a timed industrial messaging service for utility applications [11][12].

We are designing a lightweight remote monitoring system prototype able to support Web based MMI (Man-Machine Interface). The aim of the prototype, at the end of the project, is to be fully written in Java. The overall solution features a lightweight monitor for small and medium power plant units in the context of European open market for power supply.

Confidence on Real-Time Java has been gained at CNAM-CEDRIC through the design of a QoS enabled real-time distributed object platform [13]. This prototype used an enhanced real-time micro-kernel (ChorusOS) with deadline scheduling, end-to-end deadline inheritance, zero-copy buffer management, a real-time inter-process communication facility based on an ATM 155Mb/s network, and offered a specific real-time java virtual machine to run object oriented application entities. This platform was very efficient, but really too specific [14]. We are trying to obtain an equivalent architecture with off-the-shelf components [2]: the pSOS+ real-time micro-kernel [26], the PERC Real-Time Java Virtual Machine (RT-JVM) [18][19], and Industrial Ethernet (switched full-duplex Ethernet and TCP/IP) based plant communications.

The Java hypothesis leads us to investigate an object oriented middleware solution to support distributed coordination of intelligent captors and actuators. We adopted Jonathan [6], an open source Object Request Broker (ORB) [20], to handle distributed objects interactions. Jonathan is a flexible ORB that allows pluggable communication protocols, and that offers different personalities: CORBA 2.0, RMI, and, RMI-IIOP. TASE.2 based timed object oriented industrial messaging services [3] provides facility to coordinate interactions with actuators and sensors. This framework is also intended to replace the IEC TASE.2 standard protocol for the communication outside the power plant. Section 2.2 explains why and section 3.4 gives hints on the design of the solution.

This paper describes the functional architecture of the embedded remote monitoring tool, and the main characteristics of its on going implementation. We conclude with our perspectives on the use of RT Java.

2. Overview of the current prototype

The following description is deduced from EDF's functional requirements. A representative embedded remote monitoring application for small and medium power production units is depicted in Figure 1. The monitoring tool is a real-time computer (an MBX 821, a Power PC processor from Motorola, in our case) with a real-time kernel (pSOS+ from Wind River, in our study).

pSOS+ has been evaluated in a previous work. It supports real-time computing, the instrumentation and measurement system (IMS), a real-time database in memory, and a lightweight servlet server.

The monitoring computer can be accessed remotely. Production management, control computers, operators or maintenance crew are remote users. A point-to-point link enables access to/from outside world. It can be a TCP/IP based dedicated leased line, or an Internet based virtual private network. This link supports TASE.2 communications. These communication constraints are deduced from EDF's requirements. Remote control computers implement: production management functions, advanced MMI, alarms, process variables and objects, logs and printing functions. Maintenance crew uses portable PCs with lightweight web based monitoring and control operator interface. Mobile solutions with palm and GSM could be considered too in the future.

Plant communications are based on industrial Ethernet. A fieldbus system is used to communicate with sensors and actuators. An optional PC supports input/output devices (printer, logs…) if/when needed.

The full architecture should be built with common off-the-shelf products. It should be secure, reliable, efficient and cheap. Communications with Tini boards [10] will be explained later in this paper, Tini boards feature next generation intelligent devices that should be deployed in factory plant platforms.

3. TASE.2 based communications

Formerly, the TASE.2 protocol [11][12] allows production data exchanges between utility control centers and production centers. It may be applied in any domain with the same requirements. Factory automation is another possible area of application. The support of TASE.2 services with applications outside the power plant is mandatory in our project.

TASE.2 is an IEC protocol built on top of the Manufacturing Message Specification (MMS) [24]. TASE.2 is client/server based, and provides its own object model. TASE.2 objects and services are mapped onto MMS abstractions and services. As a server, a center exposes its real resources as a subset of available data. Different clients can access these data (one or more control centers), each client can have different views, and clients can be servers themselves.

Two types of interactions are provided. Client initiated interactions are called "operations". Server initiated interactions are called "actions". Access control is handled at the organization level between centers through a Bilateral Agreement. A bilateral agreement defines formally the set of data that are exchanged between control centers. It is the responsibility of servers to ensure access control. Data structures are represented through Data Objects, and Data Value Objects provides their values. Data Value Objects can be measurements (real or integer) or status (bit strings). Data Value Objects can have attributes (freshness, timestamp, way the data is provided…). Data Objects can be gathered in Data Set Objects. Four semantics are provided to exchange data between control and production centers: "once" (immediate client/server request), "periodic" (periodic transfer), "exception"(state change based transfer), "event" (event condition based transfer).

The TASE.2 standard is split onto nine conformance blocks to provide some modularity. A join project between EDF's and CNAM-CEDRIC investigated blocks 1,2 and 5 functions, through the port of a TASE.2 server over pSOS+ [4]. Block 1 corresponds to mandatory services. It defines immediate client/server interactions and periodic reports. Block 2 is related to condition based reports. Block 5 deals with remote control of devices.

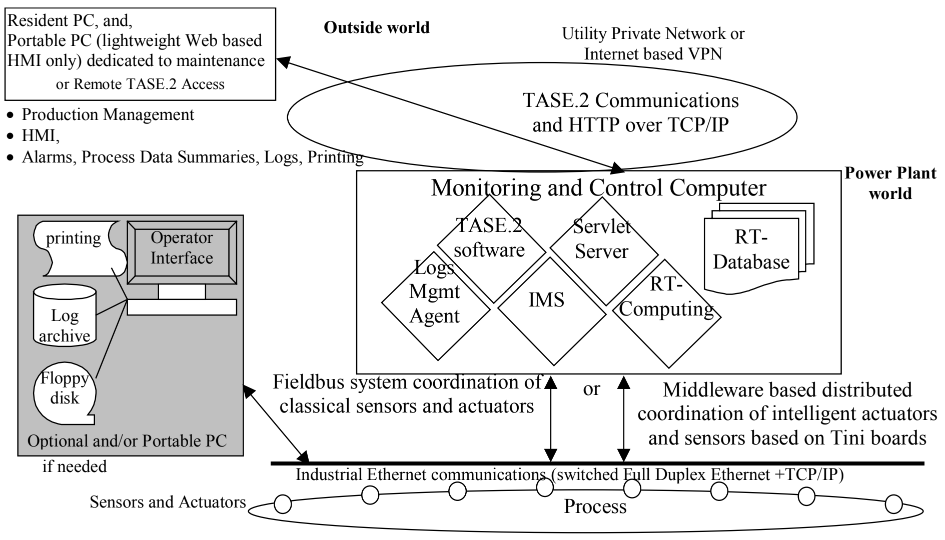
Resident PC, and,
Portable PC (lightweight Web based HMI only) dedicated to maintenance or Remote TASE.2 Access
Outside world
Utility Private Network or Internet based VPN
TASE.2 Communications and HTTP over TCP/IP
• Production Management
• HMI,
• Alarms, Process Data Summaries, Logs, Printing
Power Plant world
Monitoring and Control Computer
TASE.2 software
Servlet Server
RT-Database
Logs Mgmt Agent
IMS
RT-Computing
printing
Operator Interface
Log archive
Floppy disk
Optional and/or Portable PC if needed
Fieldbus system coordination of classical sensors and actuators
or
Middleware based distributed coordination of intelligent actuators and sensors based on Tini boards
Industrial Ethernet communications (switched Full Duplex Ethernet +TCP/IP)
Sensors and Actuators
Process

Figure 1. Architecture of our Embedded Remote Monitoring Tool

The TASE.2 server that we experimented is proprietary. Also, TASE.2 is complex, built with obsolete layers. To circumvent these problems, we are going to adapt the TASE.2 services over an ORB [3]. The key elements of this design are described in section 3. This framework offers objectified TASE.2 services that defines timed object-oriented industrial messaging services and that can also be used to support distributed interactions with intelligent captors and sensors.

4. RT-JVM

We are involved in different research and teaching projects that use Java. The targeted applications are in the field of factory automation, process control and distributed multimedia. Real-Time Java (RT-Java) is an interesting proposal. PERC is Newmonics Inc's Real-Time Java Virtual Machine (RT-JVM) [18][19], and CNAM-CEDRIC is a beta tester of this product. Also, PERC was the only commercial RT-JVM available at the time we decided to investigate RT-Java.

PERC supports JDK version 1.1. PERC provides an embedded toolkit. A set of Java classes allows the development of drivers, by performing direct access to memory and interrupt handling. It introduces two proprietary extensions to perform mutual exclusion (keyword atomic) and synchronization (keyword timed). These extensions are compatible with Java. In our case, we do not use these proprietary extensions. Newmonics developed its own interface, called PNI, for accessing native functions, which replaces JNI from Sun.

Garbage Collector (GC) provided by PERC is preemptible and incremental. In fact, the work of GC over memory is divided into small bounded atomic time intervals. Thus, the RT-JVM can preempt the GC between these intervals and is not forced to wait for the GC to complete.

PERC also supports Just in Time (JIT) compilation. Therefore, it could be very useful to provide a regulation loop implemented in Java issued from a formal specification language.

Firs benchmarks give a first overview of PERC capabilities. A full study measuring the influence of native threads on RT-Java thread scheduling should be performed to evaluate if PERC fulfill application requirements.

5. Servlet server

General application requirements indicated that a lightweight web server should run on the embedded remote monitoring tool [7]. Previous experiments have shown that a web browser can be used to support the synopsis of a power process [9]: control valves, temperature controller, pressure measurement… can be refreshed using information issued by the process in real time. This framework achieves a lightweight and low cost MMI interface.

We address this feature using a Java web server on top of the PERC RT-JVM. The web server is a servlet spawn by a servlet server. The servlet server implementation uses the Acme package [1]. In the context of PERC over our MBX these servlets are pre-fetched in memory. The default servlet is the web server associated to the processor.

The Acme package is relatively old and specific; it corresponds to the definition 1.1 of the servlet API. It has been re-written to rely on the javax.servlet.* interface from Sun and to conform the current 2.0 servlet API.

Other servlets can be implemented. The servlet server is also able to spawn a fieldbus driver for industrial Ethernet written in Java. We experimented the Modbus interface using this feature [17]. This is a convenient way to interact with standard sensors and actuators from Schneider Electric. This example illustrates also a way to implement a gateway between http and fieldbus systems.

6. Real-time computing

Distributed real-time constraints are soft, about 500ms. Constraints are related to end-to-end interactions: between sensors and outputs of the regulation loop, and between the embedded control computer and remote TASE.2 clients.

Real-time computing is the heart of the monitoring function implemented in the computer. This module implements the regulation loop. Commands can come also from a remote center too.

The regulation loop is generated using the MATLAB-Simulink [15] environment. A model, written by power engineers, expresses the technical computing functions of the regulation loop. The programs generated from the model are C programs, in our case they have been delivered by EDF. As far as we know, MATLAB-Simulink programs can't be called from Java programs [16].

Principles of the regulation loop are the following. The real-time computing component takes entries produced by the physical process through the IMS module and computes simple and complex variables that can be gathered to provide process objects deduced from the application specifications. The resulting objects are stored in the real-time database, which is fully memory resident.

It is a great challenge to re-write the regulation loop in Java. Should we let the regulation loop outside our Java experimentation? If not, how can we get Java code directly issued from a computing model compatible with the technical requirements of the power control application domain?

7. Status of the prototype

The current prototype demonstrates the feasibility of the architecture depicted in figure 1. All the functions have been implemented and tested except the IMS module [22]. This part is outside the scope of our investigations; it needs specific third party software and/or hardware. It is not mandatory to prove the feasibility of our architecture. An integrator is able to handle this task easily. Also, equivalent work has been done by our lab in the context of another project dedicated to the remote control of a Numerical Controller [8].

The timed object oriented industrial messaging services for plant communications is under development in Java and C++.

8. Conclusion

The design of a RT-Java based embedded remote monitoring tool is a work in progress. Current investigations show its feasibility [22]. From a general point of view, the use of Java is a benefit for code mobility. The use of a RT-JVM brings enhancements. Our experiment shows that RT-Java did not provide any new problem during implementation: we used only the most common features of Java 1.1.x; we did not use any specific words (atomic or timed).

The key issue to determine is: should we use Java processors or RT-Java on real-time kernels to address needs of industrial applications? This paper asserts that the second approach is tractable and provides interesting results. But stringent comparisons have to be pursued. We believe that the implementation of the regulation loop will influence the choice of one of the two approaches.

References

[1] Acme Servlet Server Classes: http://www.acme.com/java/software/Acme. Serve.servlet.Servlet.html, October 6th 2000.

[2] L. Bacon, E. Becquet, E. Gressier-Soudan, C. Lizzi, C. Logé, L. Réveilleau: *Provisioning QoS in Real-Time Distributed Object Architectures for Power Plant Control Applications*, DOA'2000. Antwerp. Belgium. September 2000.

[3] L. Bacon, E. Becquet E. Gressier-Soudan, F. Horn: *Object Oriented Timed Messaging Service for Industrial Ethernet: a fieldbus like architecture for power plant control and factory automation*, Submitted to FeT'2001, November 15-16 2001, Nancy, France.

[4] E. Becquet, E. Gressier-Soudan, G. Hellack, E. Legrand,: *Experimenting a Real-Time Industrial Messaging Service for a Power Utility: TASE.2 over the real-time micro-kernel pSOS+*, Submitted to ETFA'2001, October 15-18 2001. Antibes. France.

[5] G. Blair, J-B. Stefani: *Open Distributed Processing and Multimedia*, Addison-Wesley, 1997.

[6] B. Dumant, F. Dang Tran, F. Horn, and J.-B. Stefani: *Jonathan: an open distributed processing environment in Java*, In N. Davies, K. Raymond, and J. Seitz, editors, Middleware'98: IFIP International Conference on Distributed Systems Platforms and Open Distributed Processing, The Lake District, U.K., September 1998.

[7] E. Gressier, M. Lefevbre, S. Natkin: *TCP/IP Manufacturing Applications: an experiment with MMS over RPC*, ULPAA'95, Sidney 1995, ftp://ftp.cs.su.oz.au/bob/ULPAA/101-GRE.ps.gz, October 6th 2000.

[8] E. Gressier-Soudan, M. Epivent, A. Laurent, R. Boissier, D. Razafindramary, M. Raddadi: *Component oriented control architecture, the COCA project*, Special Issue on Manufacturing, Microprocessors and Microsystems Journal, Elsevier Science, V23N2 September 1999, p95-102.

[9] B. Guillon: *Web based lightweight HMI for Power Plant Control*, DEST final report. September 2000. (French text)

[10] iButton: *Introducing TINI: Tiny InterNet Interface*, http://www.ibutton.com/index.html, October 6th 2000.

[11] IEC Utility Communications Specification Working Group: *TASE.2 Services and Protocol*, IEC 870-6-503, Version 1996-08, August 1996.

[12] IEC Utility Communications Specification Working Group: *TASE.2 Object Models*, IEC 870-6-802, Version 1996-08, August 1996.

[13] C. Lizzi: *Design of an ATM based Real-Time Distributed System*, PhD Thesis from Conservatoire National des Arts et Métiers, 14 Décembre 1999, Paris, France, (text in french).

[14] C. Lizzi, L. Bacon, E. Becquet, E. Gressier-Soudan: *Prototyping QoS based Architecture for Power Plant Control Applications*, IEEE Workshop on Factory Communication Systems (WFCS'2000), September 2000, Portugal.

[15] MathWorks: http://www.mathworks.com, October 6th 2000.

[16] MathWorks: *Can I call MATLAB from my Java program?* http://www.mathworks.com/support/solutions/data/23385 .shtml, October 6th 2000.

[17] Modbus/TCP Homepage: *Automation Business*, http://www.modicon.com/openmbus/. May 2001.

[18] Newmonics Inc.: http://www.newmonics.com/, October 6th 2000.

[19] K. Nilsen: *Issues in the Design and Implementation of Real-Time Java*, Technical Report, NewMonics Inc., 1996.

[20] Object Web Project: http://www.objectweb.org/. March 31st.2001.

[21] OMG: *The Common Object Request Broker: Architecture and Specification. Revision 2.4, Chap 22: CORBA Messaging, Chap 23: Minimum CORBA, Chap 24: Real-Time CORBA*, October 2000, http://www.omg.org/technology/documents/formal/corba _2.htm.

[22] L. Réveilleau: *Prototyping a lightweight remote monitoring tool for small and medium power plant units*, Cnam Engineering Degree, March 22th 2001, Paris. (in French)
[23] L. Seinturier, A. Laurent, B. Dumant, E. Gressier-Soudan, F. Horn: *A Framework for Real-Time Communication Based Object Oriented Industrial Messaging Services*, Emerging Technologies and Factory Automation (ETFA'99), Barcelona, Catalogna, Spain, October 1999.
[24] Valenzano. Demartini. Ciminiera: *MAP and TOP Communications*, Addison Wesley, 1992.
[25] W3C: Simple Object Access Protocol (SOAP) 1.1 W3C Note 08 May 2000 http://www.w3.org/TR/2000/NOTE-SOAP-20000508/, October 6th 2000.
[26] Wind River: *Wind River OS/Run-Time Product pSOSystem Datasheet*; http://www.windriver.com/products/html/psosystem.html . May 2001.