OBJECT ORIENTED TIMED MESSAGING SERVICE FOR INDUSTRIAL ETHERNET: A FIELDBUS LIKE ARCHITECTURE FOR POWER PLANT CONTROL AND FACTORY AUTOMATION

Erwan Becquet, becquet@cnam.fr [**]
Mazen Abdallah, abdallah@cnam.fr [**]
Eric Gressier-Soudan, gressier@cnam.fr [**]
François Horn, francois.horn@kelua.com [***]
Laurent Bacon, laurent.bacon@edf.fr [*]

[*] *EDF DRD/EP/CCC/GAS,6 Quai Watier, 78401 Chatou Cedex France*
[**] *Laboratoire CEDRIC-CNAM, 292 rue St Martin, 75141 Paris Cedex 03 France*
[***] *Kelua Software, 9 chemin de la Brocardière, 69570 Dardilly France*

Abstract:
This paper describes an object oriented timed industrial messaging service suited to industrial Ethernet. We use an object oriented middleware technology able to run over Ethernet and TCP/IP. This framework addresses fieldbus architectures for power plant control and factory automation. The services provided by our prototype for data access are similar to the one offered by TASE.2. TASE.2 is a companion standard of the ISO Manufacturing Message Specification, dedicated to the utility domain. More precisely our work maps a subset of the TASE.2 specification to an ORB environment. We are interested to TASE.2 block 1 (periodic data exchange), block 2 (event based data exchange), and block 5 (device control). Our services are specified with CORBA IDL. Our result has a general scope and can be used for any factory network built on top of industrial Ethernet.

Keywords: TASE.2, real-time, object oriented middleware, fieldbus, industrial Ethernet, process control, factory automation

1. INTRODUCTION

The present economical and technical environment for power plant control and factory automation applications is rapidly moving: users ask for new services (access to monitoring information via the web, via Personal Digital Assistants, via mobile devices), as well as the enterprise (tools for e-business, for rapid prototyping). In this context, integration and adaptation capabilities are major factors of success. Industrial Ethernet (TCP/IP over switched full-duplex Ethernet networks) (GGH, 2001) is an emerging technology seen as an open and unifying solution for this problem. This paper goes a step further and proposes a general object oriented industrial messaging service for power plant control and factory automation that can be efficiently implemented on top of industrial Ethernet. This framework based on the TASE.2 IEC specification (Telecontrol Application Service Element version 2) (UCS, 1996*b*) (UCS, 1996*a*) also known as Inter-Control Center Communications Protocol (ICCP) introduces a generic industrial messaging service that can be

used at different levels in industrial application architectures. The preliminary work presented in this paper only considers the most important functions dealing respectively with periodic data exchange, event based data exchange and device control. A first implementation of this limited framework is in progress using an open flexible ORB (Object Request Broker) called Jonathan (Dumant *et al.*, 1998). This ORB written in Java is able to support various personalities (CORBA 2.0, RMI, RMI-IIOP) and to run over diverse communication protocols including all TCP/IP based protocols and thus industrial Ethernet. At the same time, we also develop a C++ version on top of MICO ORB, which is a free implementation of CORBA 2.3.As far as we know, the one equivalent work we know is (Oquendo and Attaoui, 2001). Their solution is built using the FIP fieldbus protocol. They also uses Jonathan for the implementation of their protocol.

This paper is organized as follows. Section 2 rapidly presents the industrial context of this work i.e. the EDF's power plant control applications. Section 3 motivates our proposal. Section 4 and 5 give an overview of the two foundations of the described framework: the TASE.2 standard and the RM-ODP/ReTINA model used to map the TASE.2 standard onto the object oriented paradigms of distributed ORBs (Object Request Brokers). Section 6 concludes and presents future works.

2. CONTEXT: POWER PRODUCTION ARCHITECTURE

The overall context of this study is the control of the thermic power plants owned by EDF (Electricité de France), the french main energy producer.

2.1 *Thermic power Process*

The classical thermic power plant technology is based on a water/steam cycle as depicted in figure 1. Water, when presented to a hot source, is transformed to a steam which is injected in a turbine coupled with an alternator. This last element produces the electric power. Since the used hot steam has to be freshened into water, thus a new cycle can be started again. This cycle involves a large number of different physical elements in the plant: pumps (especially water pumps for water transportation in the circuit), tanks (water, fuel), valves, transformers, circuit breakers, etc. The process is too complex to be fully automated and plants are thus conducted manually. The conducting teams however are assisted by systems offering an increasing number of control command capabilities: functions for acquiring data about plant elements (such as valves states), for presenting these data in a graphical and friendly way, for transmitting commands directly to raw devices, for doing the auto-regulation of subsystems, etc.. These actions need reliability, which is generally achieved by redundant systems.

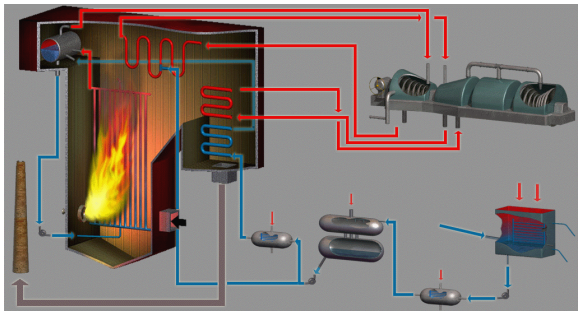

Fig. 1. Thermal Power Production Process (Thermo-electricity)

2.2 *Plant Distributed Architecture*

A thermic power plant control application is naturally distributed and must run in a heterogeneous environment. Control devices are physically distributed along the process and associated control functions are mapped onto computers with replication schemes for fault tolerance purpose. Operating systems and hardware are fully heterogeneous. The architecture of the corresponding power plant control application is layered following the classical CIM (Computer Integrated Manufacturing) (Valenzano *et al.*, 1992) model. Level 0 deals with captors and actuators. Level 1 (L1) supports acquisition and control. Level 2 (L2) performs supervision functions (operator interface, process regulation, alarms and states recording, printing, etc.). Level 3 (L3) deals with plant management (production and stock management). Communication relies on Ethernet LAN (Local Area Network) everywhere except for Level 0. Communication with the outside world involves the exchange of production data with an energy transporter or a production planning center. Usually, these exchanges use the TASE.2 standard. Solutions are based on dedicated architectures and highly depend on specific message oriented services built on top of the OSI protocol stack or proprietary communication systems like fieldbus (MODBUS for example).

3. PROOF OF CONCEPTS

The usual architecture of power plant control and factory automation applications is layered following the CIM architecture. Many usual implementations are using different dedicated technologies

and models for each level, which raises very difficult integration and adaptation problems. In this context, Industrial Ethernet (GGH, 2001) is an open technology addressing the low levels of the CIM architecture. Today, most fieldbus system providers are porting their protocols over TCP/IP in the context of industrial Ethernet (GGH, 2001). However we believe that this technology oriented solution is not general enough to cope with the wide integration and adaptation problems raised by present and future power plant control and factory automation applications. We thus suggest to use a generic timed industrial messaging service built on top of an open flexible ORB (Object Request Broker) middleware.

The ORB approach, widely used in the domain of non industrial distributed applications, consists in providing the application designers and programmers with a distributed object oriented programming model, which offers a unified programming framework that hide the heterogeneity of the underlying technologies on one hand and on the other hand, flexible middlewares that realize the adaptation layer to the various protocols and technological standards.

Such a proposal has been suggested by several observations:

People from the manufacturing and process control community (and in particular people from the fieldbus systems community) have been familiar with objects for a long time before the advent of object oriented technology. For example, they use the abstraction of VMD (Virtual Manufacturing Device) from ISO MMS (Manufacturing Message Specification) (Valenzano *et al.*, 1992) which is a kind of raw object. Most of these objects cooperate in a distributed way.

Many fieldbus capabilities are similar to the ones provided by a generic timed industrial messaging service that can be identified and isolated in the TASE.2 standards (UCS, 1996*b*)(UCS, 1996*a*). Additional communi-cation services supported by fieldbuses (peer-to-many communication) can be implemented with a multicast protocol without changing the semantic of TASE.2 exchanges.

TASE.2 is a standard, functionally suited to power plant control and factory automation, but commonly recognized as flawed by an old message oriented architecture with too many layers. Previous studies concerning MMS and TASE.2 have presented strong evidences that these protocols could be easily and efficiently implemented on top of a lightweight CORBA based middleware (Guyonnet *et al.*, 1997),(Gressier-Soudan *et al.*, 1999),(Seinturier *et al.*, 1999), (Gressier-Soudan, 2000),(Gressier-Soudan and Becquet, 2001).

4. TASE.2 STANDARD: KEY POINTS

4.1 *TASE.2 Overview*

The TASE.2 protocol is a companion standard of the popular MMS designed to specify the exchange of data between utility control centers and production units (UCS, 1996*b*)(UCS, 1996*a*) (KEMA-ECC, 1996). Although TASE.2 builds many abstractions on top of MMS ones, it can be seen as a standalone standard defining a general-purpose "timed" application protocol suited to factory automation in many domains. This is especially true in the current context where industrial Ethernet (Full Duplex Switched Ethernet plus TCP/IP) brings new opportunities for fieldbus systems (GGH, 2001).

4.2 *Interaction model*

TASE.2 is explicitly described as Client/Server based. It is a little bit more complex than usual. Two types of interactions are defined: "operations" are initiated by clients and correspond to a classical reliable method invocation (they usually return a result), and, "actions" are initiated by servers and correspond to an unreliable notification (they don't return a result). Four data transfer semantics are provided: "once" (classical client/server request), "periodic" (periodic transfer), "exception" (state change based transfer), "event" (event condition based transfer). An important characteristic of the TASE.2 model is that data exchanges between a client and a server are only possible through an "association" that must be explicitly created during an initialization phase. This association is a kind of "communication object" used to encapsulate security and temporal Quality of Service parameters. It is coupled with a Bilateral Agreement that formally defines the set of data that can be exchanged through an association established between a client and a server.

4.3 *Conformance Blocks*

TASE.2 functions are separated in nine conformance blocks. Block 1 defines a minimal set of services related to data management and periodic data exchange. Block 2 extends block 1, it provides exception semantics often referred as Report By Exception semantics. Block 3 considers blocked transfers, a performance issue. Block 4 is dedicated to text based information exchanges. Block 5 deals with devices control and block 6 with program control. Block 7 handles events and event conditions. Block 8 deals with accounting

and block 9 with time abstractions and condition monitoring.

In the following text only describes key features of blocks 1, 2, and 5.

4.4 *Variable Management*

TASE.2 defines "Data Value" objects and "Data Set" objects (supported by MMS named variables or list of named variables) managed by the server. Data Value management and Data Set management functions (KEMA-ECC, 1996) deal with the look up of existing Data Values and Data Sets, their creation, their destruction, etc. Data values reference in practice specific information such as "Indication Points" (which can contain status information, analog values, attributes, etc.).

Data Set Transfer Sets describe the way Transfer Reports must be pushed toward the client and contain for each case, parameters defining under which conditions data values related to data sets are transmitted.). The meaning of the relevant parameters is given in Table 1. Figure 2 gives an overview of variable management in TASE.2.

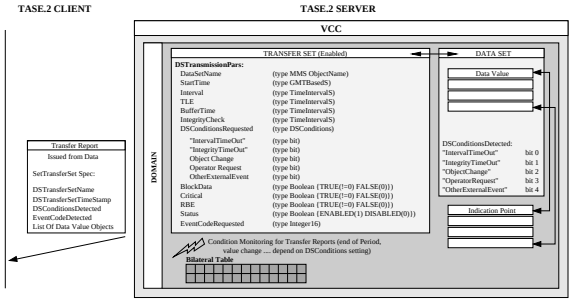

Fig. 2. Overview of Variable Management

4.5 *Device Management in TASE.2*

The Device object is an abstraction for a physical raw device, which can be controlled by a TASE.2 client. The technical implementation of how the real device is controlled and managed is outside the scope of the TASE.2 standard. Two types of devices are defined in the block 5: devices subject to inter-locking (they require a serialized access) and devices which can be freely accessed concurrently.

5. TASE.2 OBJECT ORIENTED MODEL

RM-ODP is an object framework aimed to provide a common unified foundation to the various distributed object technologies. The different standards (such as CORBA) just derive from the RM-ODP model by a specialization of its general concepts. ReTINA is such a specialization for systems requiring Quality of Service guarantees (such as timeliness properties, security, etc.)

Attributes			*Meaning*
EventCodeRequested			Application event specification
DS Tranmission Pars	**Interval** ①		Time interval between reports
	Start Time ①		Start Time for periodic reporting
	RBE, Report By Exception ②		false: periodic reporting, true: event base reporting
	Integrity Check ②		Time value defining interval for integrity check, if Integrity Time Out condition is used.
	TLE, Time Limit for Execution ②		Deadline for reporting
	Buffer Time ②		Time interval for buffering the Object Change condition before reporting.
	Critical ②		The client needs to ack reports
	DS Condition Requested (status allowing reporting)	**Interval Time Out** ①	Indicates whether or not the server shall send a report when the Interval time arrives
		Operator Request	Indicates if the server shall send a report when an operator requests it.
		Object Change ②	Indicates if the server shall send a report when an object in a Data Set changes
		Integrity Time Out ②	Indicates if the server shall send a report of the entire Data Set when Integrity Check expires
		Other External Event	Indicates if the server shall send a report when an appli-cation event condition becomes true

Table 1. Data Set Transfer Set Condition Monitoring Attributes

①block 1, RBE set to false, ②block 2, RBE set to true

5.1 *RM-ODP/ReTINA model*

The ReTINA architecture (Blair and Stefani, 1997) is based on the following general concepts:

- An **object** is an entity containing (encapsulating) information and offering services;

objects may be of arbitrary granularity (from one byte to a telephone network...).
- An object can only interact at **interfaces**: informally, the interfaces of an object are its access points, which means that all the interactions of an object with its environment must occur at one (and only one) of its interfaces. Interfaces are manipulated by reference.
- Two objects – say O_1 and O_2 – may interact in two different ways: either object O_1directly invokes an operation on O_2 (if it belongs to the same address space), or it invokes the same operation on a **binding object** whose role is to transmit the invocation to $O_2$and to return a result, if necessary.

The prime characteristic of the ReTINA architecture is to provide an explicit abstraction for the communication mechanisms. Binding objects are fundamental in many application environments for at least 2 reasons:

(1) They offer a natural and adequate abstraction for the communication semantics and properties, such as multicast, timeliness, security properties.
(2) They encapsulate the mechanisms used both by the networking and local execution infrastructures to engineer the communication: for example, the encoding/decoding algorithms, the buffer and thread management policies, the scheduling policies, etc. Binding objects are usually composite objects, distributed over several address spaces.

This makes possible to plug arbitrary forms of binding policies between objects beyond the implicit binding model for clientserver interactions assumed by standard architectures like CORBA or Java Remote Method Invocation (RMI).

The second characteristic of the ReTINA architecture is that interface naming does not depend on the ability to actually access it. This property makes the architecture really flexible since it allows:

- the transmission of an interface reference via a given binding even if the reference cannot participate in bindings of the same type,
- the participation of a given interface in several bindings of different types,
- the designation of an interface even if it is not accessible (mobility, failure...)

The ReTINA model thus defines a minimal kernel whose role is to provide a generic environment that can be used by any binding factory to create and to manage specific bindings. With this architecture, it is possible to introduce new binding factories written by application developers and possibly discovered and installed at runtime.

It also provides an explicit abstraction for interface references that may be accessed and modified by binding factories. Different binding factories may thus coexist in the same address space. Each of them brings in the functions needed for specific binding types, and safely manipulates the same interface references. Binding factories essentially provide two kinds of methods:

(1) export methods, that let interfaces be registered by the target binding factory, so that it may create bindings to access them,
(2) bind methods, that require the establishment of a binding between a set of exported interfaces.

In this approach, a CORBA or a RMI compliant ORB can be built as a particular "personality" (i.e. a set of APIs and language mappings), thus decoupling the specifics of the CORBA or RMI API from the personalityindependent kernel interface.

In our work, we use the CORBA personality of Jonathan.

5.2 *TASE.2 Object binding Model*

As we already mentioned, interactions between TASE.2 clients and servers (VCC) are quite complex: all data exchanges necessarily use an explicit association created before use in response to a client request. Data exchanges either correspond to operations (originating from the client) or actions (originating from the VCC). As a consequence, all TASE.2 entities are both client and server. The so-called client object declares an "Association Management interface" (used to request the creation or the destruction of an association).

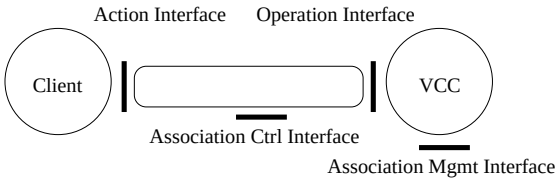

Fig. 3. The TASE.2 binding object

The creation of an association results in the creation of three interfaces:

(1) an **Operation interface** on server side,
(2) an **Association Control interface** on the server side,
(3) an **Action interface** on the client side,
(4) and an **Association Management interface** on the server side.

```
typedef string DataNameType;
typedef sequence<DataNameType>
        DataNameSeqType;
exception UnknownDataNameType {};

interface DataValueManagement {
DataType getDataValue(in DataNameType name)
         raises (UnknownDataNameType);
void setDataValue(in DataNameType name,
                  in DataType val)
     raises (UnknownDataNameType);
DataNameSeqType getDataValueNames();
DataType getValueType(in DataNameType name)
         raises (UnknownDataNameType);
};
```

Fig. 4. Data Value Objects Interface

6. TASE.2-BASED TIMED OBJECT ORIENTED MESSAGING SERVICES OVER AN ORB

6.1 *IDL specification*

A part of the distributed object oriented technology relies on mapping rules between remote objects method invocations onto messages exchanged on the network (and converse). These mapping rules are embodied in an IDL (Interface Definition Language), which is technology neutral in order to ensure interoperability between objects running on different hardwares and operating systems and developed in different programming languages.

We therefore suggest to specify a generic timed messaging service in the CORBA IDL (which is the most popular one). The result is then a technology neutral specification of a part of the TASE.2 standard via a set of interfaces and data types.

Figure 4 presents a short piece of the IDL specification. It deals with the Data Value object method interface.

6.2 *The Jonathan flexible ORB*

Jonathan (Dumant *et al.*, 1998) is a Java implementation of the ReTINA specifications: it offers a communication framework allowing the modular construction of binding factories (suited for RPC or stream styles of interactions), and the reuse of protocol components between binding implementations. In this framework, a binding object is composed of protocol sessions ("Protocol" should be understood here in a loose way: it effectively represents a protocol like TCP or GIOP, but it may also represent a service, like a compression service) and inner binding objects.

To bind the client and server, the outer binding factory creates a stub and a skeleton, and asks the GIOP protocol to bind them. In turn, the GIOP protocol creates sessions on the client and server sides, and asks the TCP/IP protocol to bind them. Protocols appear as specific binding factories, possibly using lower level protocols. Protocol sessions implement the protocol semantics, receiving messages and transforming them before sending them.

6.3 *Object Oriented Timed Industrial Messaging Services*

With previous prototyping we learnt how to design an object oriented industrial messaging services over CORBA (Gressier-Soudan *et al.*, 1999) (Gressier-Soudan, 2000). The same design patterns can be used. The TASE2 client and server contain a CORBA server object each. Clients and servers are multi-threaded if possible.

(Gressier-Soudan and Becquet, 2001) discussed different ways to implement a TASE.2 service on top of CORBA. We decided to eliminate the use of our Java based object oriented service (Gressier-Soudan *et al.*, 1999). Our timed object oriented messaging services are built directly over the middleware.

The CORBA server object on the TASE.2 server side supports the VCC interface described before, it implements operations. The corresponding methods are classical invocation methods, thus they return results. The other TASE.2 objects are supported by the VCC as objects implemented with the programming language, Java in our case. The VCC interface inherits from all basic objects interface. Local communications between the VCC and the physical resources manager use a specific adapter. They are implemented as efficiently as possible. For experimentation purpose the ORB supports local interactions, it is enough convenient. An alternate solution could use Java Native Interface.

The CORBA server object on the Client side supports the Transfer Report Services. The corresponding methods are classical invocation methods without result parameters.

Figure 5, hereafter gives an overview of the design of our prototype.

The previous time based features could have led us to say that the service we are providing is real-time. We are able to address such environments but we prefer to introduce the term "timed". It is more realistic with the technology we address. The main reasons why we don't argue that we are doing native real-time are the following:

The use of Ethernet, either a switched full-duplex Ethernet, avoids being real-time! Ethernet can

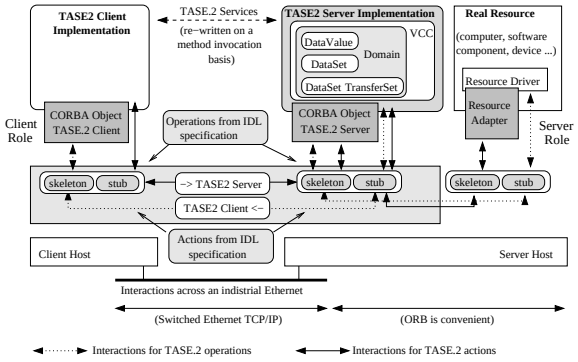

Fig. 5. Object oriented TASE.2 design over an ORB

be very fast (10Gb/s max throughput currently) but fastness is not punctuality. Switching allows bandwidth guaranties for links but determinism depends on switch behavior (cut through vs store and forward or mixed). 802.1Q/p helps, it supports basic priorities. Ethernet becomes better but it will not be as deterministic as ATM or FDDI.

IP is a datagram oriented protocol, it is a best effort approach, nothing to do with time determinism. Integrated Services based on the Resource reSerVation Protocol (RSVP)(Gaines and Festa, 1998) or Differenciated Services based on priority and efficient queuing in routers (Blake and al., 1998) should help to achieve real-time network services. The future of real-time networks depends on QoS (Quality of Service) or CoS (Classification of Services) based architectures.

TCP has some deep mechanisms that works against real-time requirements. Generally, acknowledgements and flow control (sliding window) don't help. Also, congestion avoidance with slow start introduces unpredictable jitter. The TCP Vegas proposal (VEGAS, 2001) should be better but not a sufficient choice to address communications with real-time constraints.

The ORB we use is implemented in Java. The JVM (Java Virtual Machine) does not necessarily run on top of a real-time kernel. We could use a real-time JVM like PERC (Nilsen, 1996) that we experimented on top of the real-time kernel pSOS+ from Wind River (formerly ISI). In our context, PERC/pSOS+ showed a predictable behaviour (Réveilleau, 2001). But we want open and general implementations. What we need is an efficient time management facility in the kernel and the JVM that we could use.

To address the problem of Java ORB, we develop in parallel a C++ solution, with same IDL files and same classes hierarchy. We will be able to compare the two solutions and to perform interoperability tests between the two implementations, the C++ one and the Java one.

7. CONCLUSION AND FUTURE WORKS

The TASE.2 services and protocol, conformance blocks 1 2 and 5, defines interesting capabilities to provide time constrained exchanges of process variables. This lead this TASE.2 framework to a similar level of functionalities as the one offered by fielbus systems. The difference is: fieldbus systems support one-to-many communications whilst TASE.2 offer peer-to-peer relationships. But the TASE.2 interaction model can be extended at the price of a reliable multicast protocol, Lightweight Reliable Multicast Protocol could be a solution, a Java code is freely available on the web (Liao, 1998).

Due to the nature of CORBA, our solution can be built over industrial Ethernet. Our mapping of TASE.2 functions over a CORBA ORB brings a general powerful timed object oriented messaging service able to address power control and factory automation applications. This solution answers low level communication requirements. Jonathan is a flexible ORB that implements a RMI-IIOP personality, so our proposal could be implemented in Java RMI based environments. We developp at the same time a C++-based solution.

Our proposal can be extended with recent OMG's real-time related extensions to the CORBA platform. Our work fits in the service interface requirements described in the smart transducers RFP (OMG, 2000*b*). Also, our prototype can be enhanced with real-time features without any changes, due to the Jonathan ORB. Jonathan can address real-time environments and comply to Minimum CORBA (for embedded systems), CORBA Messaging (for asynchronous method invocations) and real-time CORBA (for handling end-to-end priorities) (OMG, 2000*a*). We already know that Jonathan can be extended with real-time communication protocols like ATM (Seinturier, 1999). To solve distributed real-time constraints the distributed scheduling framework (OMG, 2001) could be added to Jonathan through RAPIDSched (Tri-Pacific, 2001*b*). The advantage of this approach is that distributed scheduling analysis tools like RAPIDRMA (Tri-Pacific, 2001*a*) exist. But we know that the integration of real-time properties of different platform components such as operating system, protocols and ORB can be a hard task (Bacon *et al.*, 2000) (Lizzi *et al.*, 2000).

8. REFERENCES

Bacon, Laurent, Erwan Becquet, Eric Gressier-Soudan, Christophe Lizzi, Christophe Logé and Laurent Réveilleau (2000). Provisioning qos in real-time distributed object architectures for power plant control applica-

tions. In: *2nd IEEE International Symposium on Distributed Objects and Applications (DOA'2000)*. IEEE. Antwerp, Belgium.

Blair, G. and J-B. Stefani (1997). *Open Distributed and Multimedia*. Addison-Wesley.

Blake, S. and al. (1998). An architecture for differentiated services. Technical report. IETF DiffServ Working Group. RFC 2475.

Dumant, B., F. Horn, F. Dang Tran and J-B. Stefani (1998). Jonathan : an open distributed processing environment in java. In: *Middleware'98 : IFIP International Conference on Distributed Systems Platforms and Open Distributed Processing* (K. Raymond N. Davies and J. Seitz, Eds.). The Lake District, United Kingdom.

Gaines, G. and M. Festa (1998). A survey of rsvp/qos implementations. Technical report. RSVP Working Group. update 2.

GGH (2001). The online industrial ethernet book. http://ethernet.industrial-networking.com/. URL.

Gressier-Soudan, E., M. Epivent, A. Laurent, R. Boissier, D. Razafindramary and M. Raddadi (1999). Component oriented control architecture, the coca project. *Special Issue on Manufacturing, Microprocessors and Microsystems Journal* **23**(2), 95–102. Elsevier Science.

Gressier-Soudan, Eric (2000). Prototyping a corba based mms -industrial communications with corba. In: *OMG Technical Meeting*. OMG. Burlingame, California USA. ftp://ftp.omg.org/pub/doc/mfg/00-09-16.pdf.

Gressier-Soudan, Eric and Erwan Becquet (2001). Real-time corba-mms for embedded systems. In: *Workshop OMG*. OMG. San Diego, CA, USA.

Guyonnet, G., E. Gressier-Soudan and F. Weis (1997). Cool-mms: a corba approach to iso-mms. In: *ECOOP'97 Workshop : CORBA : Implementation, Use and Evaluation*. Jyvaskyla, Finland.

KEMA-ECC (1996). *ICCP User Guide*. final draft ed.. Mineapolis, USA.

Liao, T. (1998). Light-weight reliable multicast protocol specification. http://webcanal.inria.fr/lrmp/draft-liao-lrmp-00.txt. INRIA.

Lizzi, C., L. Bacon, E. Becquet and E. Gressier-Soudan (2000). Prototyping qos based architecture for power plant control applications. In: *IEEE Workshop on Factory Communication Systems (WFCS'2000)*. IEEE. Porto, Portugal.

Nilsen, K. (1996). Issues in the design and implementation of real-time java. Technical report. Newmonics, Inc.

OMG (2000*a*). The common object request broker : Architecture and specification. revision 2.4. Technical report. Object Management Group.

OMG (2000*b*). Smart tranducers interface. Request for proposal. Object Management Group. orbos/2000-12-13.

OMG (2001). Dynamic scheduling real-time corba 2.0. Joined final submission. Object Management Group. orbos/2001-04-01.

Oquendo, L. and A. Attaoui (2001). Deterministic orb (dorb) for distributed real-time applications. In: *IASTED'01 : International Journal of Computers and Applications*. Calgary, Canada.

Réveilleau, Laurent (2001). Prototyping a lightweight remote monitoring tool for small and medium power plant units. Engineering degree. CNAM. Paris, France. in French.

Seinturier, L., A. Laurent, B. Dumant, E. Gressier-Soudan and F. Horn (1999). A framework for real-time communication based object oriented industrial messaging services. In: *ETFA'99*. Barcelona, Catalogna, Spain.

Seinturier, Lionel (1999). Intégration de liaisons atm dans l'orb corba jonathan. Technical Report NT/CNET/6125. France Telecom R & D.

Tri-Pacific (2001*a*). Rapidrma: The art of modeling real-time systems. Technical report. Tri-Pacific. http://www.tripac.com/html/prod-fact-rrm.html.

Tri-Pacific (2001*b*). Rapidsched: Scheduling for real-
time corba systems. Technical report. Tri-Pacific. http://www.tripac.com/html/prod-fact-rsd.html.

UCS (1996*a*). Tase.2 object models. version 1996-08. iec870-6-802. iccp inter-control centre communications protocol version 6.1.. Technical Report IEC 870-6-802. Utility Communications Specification Working Group. Version 1996-08.

UCS (1996*b*). Tase.2 services and protocol. version 1996-08. iec870-6-503. iccp inter-control centre communications protocol version 6.1.. Technical Report IEC 870-6-503. Utility Communications Specification Working Group. Version 1996-08.

Valenzano, A., C. Demartini and L. Ciminiera (1992). *MAP and TOP Communications*. number ISBN 0-201-41665-4. Addison-Wesley. Wokingham, England.

VEGAS (2001). Tcp vegas home page. http://www.cs.arizona.edu/protocols/. URL.