Proceedings of the International Multiconference on
Computer Science and Information Technology pp. 859–869
ISSN 1896-7094

Real-Time Communication for Large Scale Distributed Control Systems

Mariusz Postol

Department of Computer Network, Technical University of Lodz Poland
postol@zsk.p.lodz.pl

Abstract. Because of their scale, complexity and requirement of expandability, Large Scale Distributed Control Systems (LSDCS) are usually created in a multistep integration process. To succeed, it has to be governed by well-defined information architecture, appropriate communication infrastructure and the supervisory role of the time notion taken into consideration from the very beginning of the design stage. Mutual influence of the architecture and underlying communication is discussed in the paper and a novel systematic design methodology is proposed to greatly reduce the complexity. A dedicated communication component is proposed in this approach. Functionality and scheduling algorithms offered by these components enable to satisfy all the defined prerequisites of the real-time distributed control and design the robust system in a systematic and uniform way. The presented case study proves that the solution not only allows the real-time process requirements to be met, but also is a platform for multi-enterprise collaboration.

1 Introduction

The defining characteristic of a networked control system (NCS) is having one or more control loops closed via a serial communication channel. A distributed control system means that the feedback is provided from a supervisory control system to isolated islands of automation and all of them deal with the same real-time process.

Large scale emphasizes three main features of the discussed type of systems [1, 2, 6, 7]. The first two are especially big responsibility and intrinsic complexity. The third one is a large geographical area the system is spread over. An example is described in Sect. 6.

The first installations of LSDS appeared just after microprocessor technology had started rocketing [1]. In spite of no objections concerning possible improvements to the process management and safety, the number of this kind of applications is incomparably smaller than others. The main reason is the complexity of information architecture determined by time relations and a communication infrastructure.

The article showns that it is possible to reduce complexity and provide a systematic uniform design pattern using the proposed modified relation model where an additional component provides communication support for the implementation of association between system components. The solution enables normalization of information architecture for all applications and making it independent of the underlying communication links structure.

2 Prerequisites

A very important feature of the real-time systems is their intrinsic reliability [1, 6, 7]. Reliability is crucial because a failure of the computer system could result in an economic disaster or even loss of human lives. Unfortunately, there are no appropriate means to guarantee full reliability, but it can be significantly increased by using well-designed principles and approved common solutions during the system life cycle.

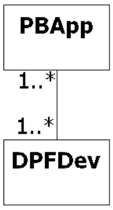

Fig. 1. Basic model

2.1 Structure

Typical simplified hierarchical information architecture [2] for the process industry involves the following levels:

1. Process and Business Management (PBM);
2. Field Management (FM);
3. Controlled real-time process;

The Field Management level consists of smart digital plant-floor devices responsible for control of real-time processes and access data from the plant floor and integrate it into management and business systems. Control and management subsystems located in the upper layer are responsible for controlling the process as a larger whole to gain additional benefits from its optimization. For further discussion it is assumed, that from the PBM layer point of view the FM layer is a sole virtual representation of the underlying layer providing additional functionality and an interface to the upper layer. It means that PBM can reach the target function and achieve full system controllability using the interface provided by the FM layer.

Both layers (1, 2) have a composite character. Therefore, while modeling, we have to distinguish association between components rather than the layers themselves. For applications (PBApp) making up subsystems on the PBM layer and for digital plant-floor devices (DPFDev) on the FM layer many to many association between them must be assigned (Fig. 1).

The previous discussion leads to a very simple design model which becomes incredibly complicated after having been transformed into the implementation diagram (Fig. 2). In this model, we can distinguish vertical and horizontal links responsible for an appropriate inter-layer and intra-layer relationship.

If there is any common part along the link between a device and an upper layer application (port, medium, repeater, etc.) additional precautions must be taken to preserve the integration of transferred data.

Basing on the above discussion, the following prerequisites can be defined to find a systematic design approach:

1. The number of links has to be reduced to keep the complication level of the architecture manageable.
2. Mutually exclusive access to common resources must be guaranteed to deploy architecture presented in the Fig. 1

2.2 Communication

In real world, we need underlying communication to instantiate association, i.e. create a link between objects (devices and applications). Although from the information architecture modeling and design point of view this communication can be considered transparent, its availability and reliability is crucial for the final result. Assuming transparency, it simplifies the problem to a great extent, provided that the assumption is valid.

We need a memory/processing engine unit to instantiate an object and we need a medium to instantiate a link between nodes. But it is insufficient. To transfer data over the medium, we have to use selected protocols controlling access to the medium and data transfer. The protocol is implemented as a component which needs processing power. Medium control and processing consume power. Additionally, protocol and medium often limit bandwidth and medium access. Any of these requirements can cause that the above assumption and, in consequence, this approach becomes unreal. Therefore, we need to look for a compromise between an unacceptable complexity and unreal assumption.

Because of the scale and area of the LSDCS, we cannot even make an assumption that the system is to be designed on a platform of one medium [10]. To transfer the data, we need a medium, but to use the medium, we need to engage an infrastructure: a *technology* (GSM, satellite, ISDN, etc.) governed by technical standards and an *organization* governed by regulations, procedures, practice, etc. A platform optimal today may be useless for future because technology is progressing rapidly and economical standing of organizations may fluctuate.

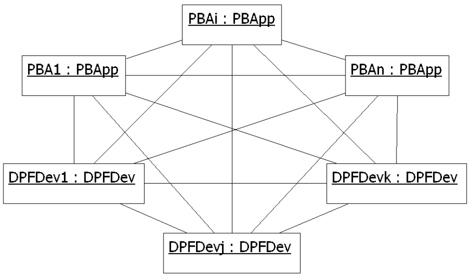

Fig. 2. Basic model implementation

Lack of common medium coverage of the whole area requires engaging simultaneously many communication infrastructures, and dealing with a multidimensional communication network—the next point of freedom. The main advantage of using many infrastructures is the possibility of improving robustness of the system by providing communication redundancy [11] in overlapping areas.

We need to provide vertical (interlayer) and horizontal (intralayer) communication to follow two-layer information architecture as described above. To make two devices interoperable, both have to use the same (a vendor specific or standard compliant) protocol. Relying on vendor specific solutions limits future system expandability and, therefore, generally it is not recommended and vendors usually offer a standard protocol for plant floor devices. Unfortunately, there are hundreds of "open standards" defined in the automation marketplace. Most of them are dedicated to FM intralayer communication, but they can be also successfully used for interlayer communication.

Traditionally, whenever a software package needed to access data from a device, a custom interface or driver had to be written. To overcome this disadvantage the OPC [3, 8] was designed to bridge applications based on general purpose operating systems and process control hardware and software applications.

The following prerequisites for a systematic approach to deployment of the communication platform for the Large Scale Distributed Control Systems (LS-DCS) can be defined:

1. It has to allow to engage multidimensional networks,
2. It has to provide communication redundancy,
3. It has to conform with a widely accepted standard.

3 Modified architecture

As it was stated in Sect. 2, the main problem we are faced with is the rapidly increasing complexity with the number of elements in the system (Fig. 2). A novel architecture is proposed in Fig. 3 to reduce this complexity. An additional element (PO) was introduced between FM and PBM layer in this architecture. The main function of this element, called *process observer*, is to represent the FM layer as a whole at the boundary of the PBM layer. Two main functions can be distinguished for this element: data and state observation. *Data observer* provides an information mirror of the current values of physical signals gathered from the plant-floor devices and integrated into one homogenous address space of process variables. *State observer* is responsible for computing current values of unavailable signals using available data and a dynamic simulation model. It also integrates obtained results into the same address space. Using the state observer it is possible to improve control quality and build fault-tolerant systems.

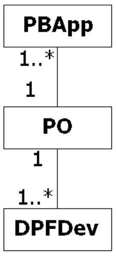

Fig. 3. Modified architecture

Additionally, it will be possible, but not necessary, to reduce the complexity once more if direct intralayer communication is replaced by indirect one through the intermediary PO element.

The approach presented above allows to virtually replace k elements by only one at the PBM layer boundary, and, in consequence, to change the many to many association to two associations of one to many. Finally, we can reduce the number of possible links from (k+n)(n+k-1)/2 to k+n and obtain homogenous architecture.

In the proposed architecture, the PO element is like a bridge connection between the process plant-floor control side and the process and business management side. For the sake of expandability and before deploying architecture presented above, it should be emphasized that the data of plant-floor devices governed by the PO should be communicated to any upper layer client application in a standard way. To provide a universal error free solution, data access has to be defined on an appropriate abstraction level as a set of well defined strongly-typed data structures and method prototypes.

4 Lower layer communication

Employing of an intermediate component as a driver for plant-floor devices is a middleware archetype used worldwide in thousands of applications. A novel lower layer communication model is proposed (Fig. 4) to use it as a consistent sole representation of a distributed real-time process at the PBM layer boundary.

Because it is so important to provide process representation at the PBM layer boundary in a standard way, it is proposed to use a widely accepted OPC [3, 8] interface implemented by *OPC_API*. *Cache* is an intermediate storage of real-time process data. The *Cache* reflects the latest value of the data (subject to update rate and deadband optimizations) as well as the quality and timestamp. *Controller* holds the plant-floor device data description.

Channel is used to represent independent communication threads conducted simultaneously to each other. Usually, lower layer communication requires multidimensional networks to transfer the real-time process data to/from plant-floor devices. In some cases, even with homogeneous infrastructure, like IP network, we can establish independent simultaneous connections using selected transport protocols (e.g. TCP/UDP). *Channel* concept allows defining as many simultaneous communication paths as it is necessary to meet current needs.

Segment represents a single communication path and is responsible for managing communication resources and data transfer from a group of devices that is to be accessed using the same transport connection. To assure mutually exclusive access to common resources, the *Channel* activates only one *Segment* at any time.

DataProvider is responsible for providing a stream of data to the *Segment.* To provide a consistent process data representation at boundary of the PBM layer, the process observer has to deal with a multidimensional network environment and vendor specific languages (protocols) used by the devices for data access. To meet requirements of this heterogeneous environment, the proposed solution enables to instantiate many custom *DataProviders* by a *Channel* and use them by a uniform data transfer algorithm realized by the *Segment.* Each segment can use only one *DataProvider*, but one *DataProvider* can be used by many *Segments* associated with the same channel. Because *Segments* are activated sequentially by the corresponding *Channel,* using of *DataProvider*s is safe for common resources.

To provide polymorphism for the environment specific needs, the *DataProvider* is located outside the main software package and inherits an interface ensuring flexible management of the communication medium and transfer of the process data. This solution makes it possible to keep the core software unchanged in all applications and flexibly adapt a skeleton to the vendor or application specific needs.

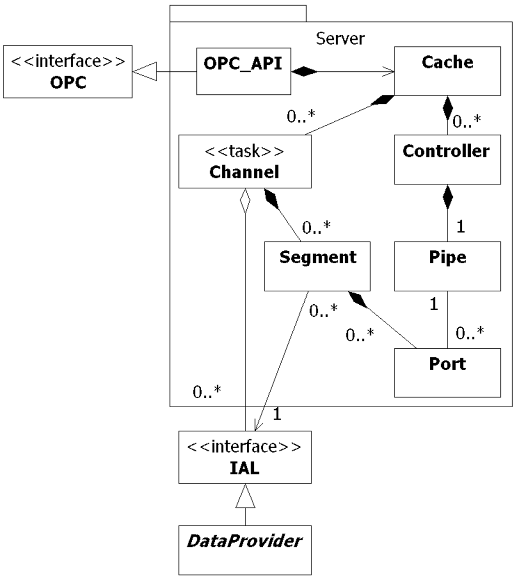

Fig. 4. Lower layer communication model

In a real environment, apart from managing communication networks and accessing plant-floor devices, monitoring and management of the recourses that make up information processing and communication infrastructure are often of the same importance as access to the real-time data. To provide a state observer or commence factory tests of any system, we need to build a simulation environment. With the data provider concept, it is possible to publish all of the mentioned types of information in the same way using the defined interface.

It is generally accepted, that redundancy is indispensable for network control systems. The redundancy of communication is the possibility of defining the excessive communication paths, which can be used to access plant-floor devices. Duplication of the communication paths can be costly, because data transfer over distributed networks is usually not for free. The crucial feature of paths redundancy is the provision of path multiplication without any necessity of transferring the same data over the network many times and controlling the backup path availability at the same time.

In the proposed model (Fig. 4), the *Pipe* concept is used to assure redundancy. *Pipe* is a collection of *Ports*, where only one is active at any time. After detecting a failure of the active path, another *Port* belonging to the same pipe is activated immediately. It represents a bidirectional device data stream. *Segment* creates *Ports* to get access to the data model description held by the *Controller* and after acquiring the process data to deposit it in the *Cache* repository. *Segment* using all associated active *Ports* builds up a data model description to schedule the sampling operations. *Segment* uses only active *Ports* and, therefore, it is assured that the process data is transferred over the underlying network once only in spite of communication paths multiplication. Behind the scene, the *Pipe* checks availability of non-active paths periodically by changing the state of *Ports*. Using paths redundancy additional spare plant-floor devices can be used seamlessly as the next level of redundancy.

5 Optimization of communication paths utilization

The main job of the communication software is to make ‘best effort’ to keep the process data fresh and allow clients to access this data randomly. From the communication point of view, two independent communication environments can be distinguished (Fig. 3):

- Connecting FM level plant-floor devices to the intermediate component (PO);
- Connecting the intermediate component (PO) to PBM level clients.

Because both are used to transfer the process data, we have to address the question how these data transfer processes are related to each other. In other words, what a relation should be applied between sampling rates of individual process variables and update rate of clients connected to the server. To answer this question and finally design an appropriate sampling scheduling mechanism on the process side, we need to take the following into consideration:

- Current communication path load and its throughput,
- Current real-time process state,
- PBM level clients needs.

All of them can change in time and, therefore, it is proposed to implement the following two unique closely coupled algorithms allowing to build cost saving, flexible solutions providing process data just in time and preserving communication bandwidth:

- *Optimal Transfer Algorithm* (OTA): responsible for minimizing the difference between requirements of client individual process data update rate and current sampling rate of a process control unit.
- *Adaptive Sampling Algorithm* (ASA): responsible for adjusting the plant-floor devices sampling rate according to the current process state.

The *Segment* is responsible for keeping the process data located in the cache fresh. The scanning schedule managed by the *Segment* can be interrupted by a direct device read/write requests and disturbed by communication errors. Additionally, communication paths redundancy and adaptive sampling algorithm cause that the load of individual communication routes changes in time. Furthermore, we need to deal with “the stupid client” syndrome, where a client unconscious of the limitations requests too high update rate. Having detailed knowledge of the communication limitations the *Segment* is able to prevent monopolization of the resources by any malicious client.

To address all these issues, the *Segment* implements a weighted fair queuing algorithm. The main job of this algorithm is to harmonize individual sampling rates with clients update rate requests to minimize the difference between both of them.

In the proposed algorithm, the individual process data descriptors are aggregated in groups. Any group has been assigned a quasi static requested sampling rate and is scheduled in the sampling queue as one item. This sampling rate is a function of client requests and limitations set in the configuration. An overload coefficient is calculated for each group, as the ratio of current to requested sampling period. To make the scanning process fair, the scheduling algorithm serves the group with the greatest overload coefficient (provided it is greater then 1) first. As the result, it makes the coefficient approximately equal for all groups assigned to the same *Segment*, i.e. the same communication path, in the steady state. Because the coefficient is monotonically increasing between the sampling instants, it prevents from causing a "starvation" disease, i.e. keeping important data fresh causes that update of less important data is not possible at all. In other words, the algorithm distributes communication bandwidth proportionally to requested sampling rate of the groups.

In some applications we can have another requirement, i.e. a request to keep bandwidth utilization below a preset limit. For example, that is the case when we use toll network. For those systems this algorithm provides additionally the possibility of controlling overload coefficient at the requested level and, in consequence, guarantees bandwidth utilization at the preset level.

Segment is responsible for keeping the cached data fresh, but it is worth stressing that "freshness" of the data depends on the process state and can drastically change in time. Therefore, to minimize the data transfer costs, the sampling rate should be adapted to the current process control needs. It is proposed to use adaptive sampling algorithm to achieve this. The *Port* is responsible for adjusting the requested sampling period of the group while updating current process values associated therewith.

6 Supervisory control of a municipal heating system

In this section, a case study of supervisory control of a metropolitan heating system is provided [4, 5, 9, 10]. The system is located in the city of Lodz – Poland and consists of heat and power plants, backbone pumping stations, backbone heat chambers and local distribution points. The heat distribution network of Lodz (750k citizens) is supplied from three plants with total thermal output of 2560MW. Their optimal utilization requires a control system to allow working on common supplying areas. As the system is distributed geographically (about 800km of pipes), safe communication between nodes (automation islands) is critical. Additionally, a great number of nodes (~ 8000) requires that the development of an appropriate system structure is preceded by a detailed analysis of the availability, throughput, and connectivity standardization.

To meet the requirements, the architecture presented in Sect. 3 is used, where the *process observer* is implemented as an OPC server [4]. It has to manage data transfer using some independent networks: VHF, GPRS, leased and dedicated fiber optic and corporate multipurpose field segments. Additionally, because of the propagation limits several transmitting on the same bandwidth locations have to use. The biggest challenge is to synchronize transmission on one common frequency and provide an appropriate level of redundancy and throughput.

The introduction of local weather automation on a large scale has caused very great daily fluctuations of the demand for heat energy. The predominant function of the application is to control the distribution of energy streams produced by the plants (hot water with flow of up to 7000 m3/h each) to allow optimization of the resources utilization and provide an interface between the operators and the heating system. To control the heat stream distribution, selected nodes on the network are used to unmanned remote control of the chamber fittings in order to switch over the supply areas from one to another power plant and/or control pressure in selected points on the network. When controlling, all components of the system must be monitored in real time. Each of the software components provides diagnostics and statistics information via OPC tags, therefore it can be monitored by HMI and SCADA stations.

Automation islands—PLC's local controllers (network nodes)—equipped with radio transceivers are connected to the OPC server – CommServer [4]. This fully configurable OPC server implements architecture and algorithms (Sect. 3, 4, 5) so it provides a multi-protocol, multi-medium and multi-channel redundant access to the data of the plant-floor devices. To ensure short response time and effective utilization of the communication channels throughput, unique optimal scheduling algorithms have been implemented.

Such system cannot work alone. Thanks to the open and coherent infrastructure it has seamlessly been integrated with the following systems: power consumption prediction, geographical information, remote control of water main pumping stations, power plant monitoring systems. Using OPC as the core technology we have proven that the obtained structure is open, robust (because of built-in redundancy) and allows a straightforward integration process.

7 Conclusion

To satisfy the prerequisites defined in Sect. 2 a novel architecture is proposed. In this architecture, an additional element called *process observer* has been introduced between FM and PBM layer (Sect. 3, 4). The main function of that element is to represent the FM layer as a whole at the boundary of the PBM layer. Additionally, two unique closely coupled scheduling algorithms have been implemented in the proposed model (Sect. 5), namely *optimal transfer algorithm* and *adaptive sampling algorithm* to build cost saving, flexible solution providing process data just in time and preserving communication bandwidth. The presented methodology and solutions were used as a platform to provide supervisory control of a metropolitan heating system supplied from three plants. Their optimal utilization requires controlling the load and pressure distribution. As the system is distributed geographically, safe communication between nodes is critical. Additionally, a great number of nodes requires an appropriate system structure.

The presented approach not only allows to meet the real-time distributed control requirements (Sect. 2), but also is a platform for multi-enterprise collaboration. It has been implemented using the OPC standard [3], [8] standard that permits a consistent method of accessing field data from plant-floor devices. Originally based on Microsoft's DCOM technology, currently new harmonized standard Unified Architecture (UA) has been just proposed. Because UA uses the concept of Web Services based on top of a proved industry chosen standard, such as XML technology it can be considered as vendor and platform independent. Using of this standard provides the prospect of seamless, truly open and easy enterprise-wide communications between devices and systems, from plant floor to MIS (Management Information System) and beyond. Feature research and implementation of the presented approach should be focused on migration towards UA standard.

References

1. Arendt D. Postol M.: *Real-time Multiprogramming System for Mine Control Center,* Microprocessors and Microsystems, 14 (1990) 39–46.
2. Bayne J. S.: *Automation and control in large-scale interactive systems,* Object-Oriented Real-Time Distributed Computing, (ISORC 2002). Proceedings. Fifth IEEE International Symposium on, (2002) 3–10.
3. Box D.: *Essential COM,* Addison Wesley, 1998.
4. *CAS: CommServer—Redundant, Multi-Protocol, Multi-Channel OPC Server For Large Scale Distributed Systems,* doc: PR214001BEN, (2005), `http://www.cas.eu`
5. *Remote Control of Lodz Agglomeration Heat Distribution System,* CAS (2004) `http://www.cas.eu`
6. Hayashi H.; Takabayashi Y.; Tsuji H.; Oka M.: *Rapidly increasing application of Intranet technologies for SCADA (supervisory control and data acquisition system),* Transmission and Distribution Conference and Exhibition 2002: Asia Pacific. IEEE/PES Volume 1, (2002) 6-10.
7. Nof S. Y., Morel G., Monostori L., Molina A., Filip F.: *From plant and logistics control to multi-enterprise collaboration,* Annual Reviews in Control 30 (2006) 55–68.
8. OPC Foundation: *OPC Overview,* Industry Standard Specification, (1998).
9. Postol M., Lipowska-Nadolska E.: *Using Monitor Concept to Maintain Concurrent Processes in Some Industrial Applications. Part II—Applications,* Using computers in elctrotechice (ZKwE 2002) Proceedings. VII Symposium on, Pozna, (2002) 429-432 (in polish).
10. Postol M.: *Remote Control Of Lodz Agglomeration Heat Distribution System,* CAS, Doc: O-03300101, (2003) `http://www.cas.eu`

11. Shono T.; Sekiguchi K.; Tanaka T.; Katayarna S.: *A remote supervisory system for a power system protection and control unit applying mobile agent technology,* Transmission and Distribution Conference and Exhibition: Asia Pacific. IEEE/PES Volume 1 (2002) 148–153.