Building Simulation 2003

Eighth International IBPSA Conference
Eindhoven, Netherlands
August 11-14, 2003

INTEGRATION OF CONTROL AND BUILDING PERFORMANCE SIMULATION SOFTWARE BY RUN-TIME COUPLING

Azzedine Yahiaoui[1], Jan Hensen[1], and Luc Soethout[2]
[1]Center for Building & Systems TNO-TU/e
P.O. Box 513, 5600 MB Eindhoven, Netherlands
[2]TNO Bouw, Afdeling Gezonde Gebouwen en Installaties
Postbus 49, 2600 AA DELFT, Netherlands

ABSTRACT

This paper presents the background, approach and initial results of a project, which aims to achieve better integrated building and systems control modeling in building performance simulation by run-time coupling of distributed computer programs. This paper focuses on one of the essential steps towards this goal, namely the communication mechanism between two or more different software tools and/or platforms. Various options for this inter-process communication are described and compared. The paper finishes with indicating future work which includes a suggestion for establishing mappings between data structures in the different data-parallel programs.

INTRODUCTION

The design of modern building services systems, including the control thereof, requires tradeoff studies of various configuration options. The importance of performing modeling and simulation studies early, and throughout the design development, has been shown many times. The major benefit of integrating modeling and simulation into the building development process is a significant reduction in total cost of ownership.

While recognizing that there are many other aspects involved in achieving lower energy consumption, greater satisfaction and higher productivity of the occupants, the work described in this paper, focuses on modeling and simulation for better control of the indoor environment.

Most current building performance simulation software does not have a flexible way of dealing with control strategies.

The current situation is that there exists software that is very advanced in control modeling and simulation features, but limited in building performance simulation concepts. On the other hand, domain specific building performance simulation software is usually relatively basic in terms of control modeling and simulation capabilities. Marrying the two approaches by run-time coupling building simulation environments and control modeling software would potentially enable integrated performance assessment by predicting the overall effect of innovative control strategies for integrated building systems.

The first part of this paper presents a brief description of advanced control strategies. The next part elaborates the reasoning behind our hypothesis that run-time coupling will facilitate integrated performance assessment by predicting the overall effect of innovative control strategies for integrated building systems. The major part of this paper is a discussion resulting in a selection of an inter-process communication (IPC) mechanism. The paper finishes by indicating our current plans for future work.

ADVANCED CONTROL STRATEGIES

With regard to control of indoor environment processes, there is no single control method or conventional control application that can solve all the challenges encountered. An integral approach is needed to be able to simulate the comfort and energetic aspects of building design. This needs to be combined with advanced process control strategies and modern model-based control methods, which are fast, accurate and robust. Being able to study the relation between design and control will eventually improve the overall quality of the built environment. So, there is a need to improve building performance simulation in this respect.

Different advanced control techniques have been applied to a wide variety of systems. In buildings they have been applied to separate processes; e.g. heating, ventilation, lighting, power generation and so on. Many of the applications reported in the literature describe the use of single methods (e.g. Zajac 1997). A more comprehensive view of advanced control is depicted in Figure 1.

Advanced control in buildings will require implementation of appropriate controllers in order to keep the relevant processes operating at the desired conditions. Here, the type of controller employed depends on the task at hand. Although it is relatively easy to tune and maintain simple controllers, several building performance related processes require

control by more sophisticated algorithms. Currently, the higher level tasks of optimisation and supervision are typically still carried out by hum operators. Due to the advent of modern technology, and advances in the field of artificial intelligent (AI), these higher level tasks can now also be automated (e.g. Meystel et al. 2002). In particular, the installation, operation and integrity of modern controllers can be supervised by higher level automated systems. The challenge here is to integrate the various components in an efficient and manageable fashion.

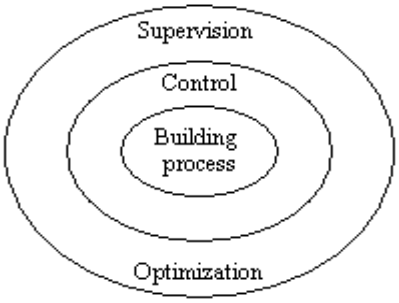

Figure 1. Hierarchical layers in integrated modern control analysis

The literature on relevant control, monitoring, supervision and optimisation techniques is extensive, with many papers each exhorting a certain solution to a particular problem. However, it is generally acknowledged that there is currently not one technique that will solve all the control problems that can manifest in modern practice. So we need to find a flexible approach which will allow easy incorporation of alternative solution strategies.

STATE OF THE ART

For more than a quarter of a century, building performance simulation programs have been developed to undertake non-trivial building (design) analysis and appraisals. In general these programs deal only with a small sub-set of the overall problem. However, advanced architectural developments require an integrated approach to design. The domains of building physics, heating, ventilation, air-conditioning (HVAC), lighting, and thermal storage systems, for example, are often closely related and it is only by taking into account their dynamic interactions, as indicated in Figure 2, that a complete understanding of building behavior can be obtained. Buildings and systems, including the control thereof, need to optimize in their entirety, they should not be designed as the sum of a number of separately designed and optimized sub-systems or components.

One of the issues facing us when we want to simulate a building plus environmental control systems is that frequently certain system components and/or control features can be modeled in one simulation environment while models for other components and/or control features are only available in other simulation software.

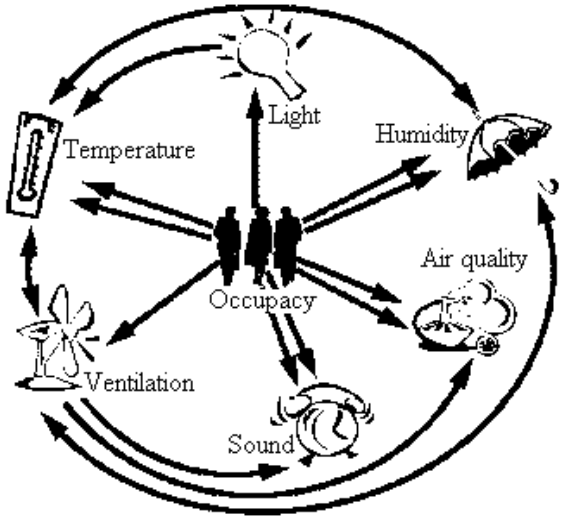

Figure 2. Dynamic interaction between physical effects and comfort aspects in a building context

As suggested earlier (Hensen 2002), one approach to alleviate this would be a semi-open simulation environment as shown in Figure 3, with the following main advantages

- Enable shared developments.
- Make it easier to consider different performance aspects (comfort, health, productivity, energy. etc.) at different levels of resolution in terms of time and space (region, town, district, building, construction element, etc). This would realize the building modeling and simulation laboratory metaphor.
- Allow components, features and models to be provided by other stakeholders (producers, re-sellers, etc who could provide models as additional product documentation) as opposed to only by software developers and researchers.

There are several strategies to enable sharing of distributed developments. In these studies, one application controls the simulation and calls the other application(s) when necessary. For more details see (Augenbroe G.L.M. 1994, Clarke J.A. 1986a, Clarke J.A. 1986b, Janak 1994, Mahdavi et al. 1999 and Papamichael et al. 1997).

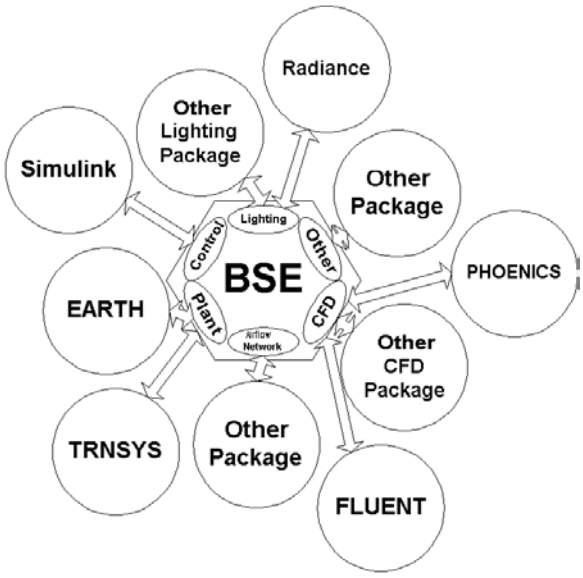

Figure 3. Semi-open simulation environment based on run-time coupled domain specific and domain independent software

The aim of the current work is to establish better control modeling in building performance simulation by integrating distributed computer programs as well as by applying advanced control analysis suites and rigorous process models based on fundamental physical principles.

The current situation is that there exists domain independent software that is very advanced in control modeling and simulation features, but doesn't have any building performance simulation concepts (e.g. Matlab / Simulink). On the other hand, domain specific building performance simulation software (ESP-r, TRNSYS, etc) is usually relatively basic in terms of control modeling and simulation capabilities.

Our hypothesis is that marrying the two approaches by run-time coupling will enable integrated performance assessment by predicting the overall effect of innovative control strategies for integrated building systems.

One of the essential steps in this work consists of developing system-level software for inter-process communication.

INTER-PROCESS COMMUNICATION

Inter-process Communication (IPC) is the capability that allows one process to communicate with another process (e.g. Yahiaoui 2002, Leffrer 1993 and Sechrest 1993).

Since we envisage that the different software modules may run simultaneous on a heterogeneous network (e.g. Unix, Windows) and we, potentially, also would like to be able to run-time couple with e.g. a building energy management system, a test-rig for a controller (hard-ware testing in the loop), or even a building emulator, the inter-process communication needs to be platform independent.

We start from the common data exchange method TCP/IP. This is a widely used method, available on most platforms and allowing inter-platform data exchange, which fits within the OSI -Open Systems Interconnection- model as illustrated in figure 4.

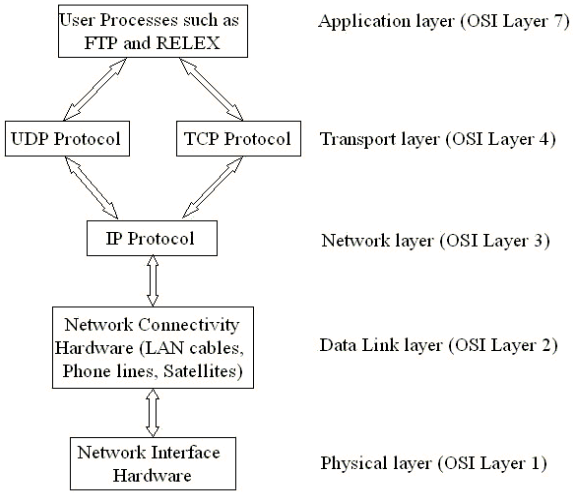

Figure 4. TCP/IP within the OSI Model

In our case, the run-time coupling should support asynchronous events that occur when the time step of one process is different from another process and synchronous events that occur when the two processes are synchronizing with the same defined time in execution. The exchange can be either unidirectional (data send in one-way) or bi-directional (the data alter from the first process to the second and from the second require passing back to the first, using client/server technique). The data structure can be transferred in the form of data files or data types, having a common format so that processes running in two different operating systems can translate them as well as the underlying communication protocols.

Data Model Interoperation

The data to be exchanged should – as much as possible – represent physical quantities as they would be measured in the real world, and should be accurate and fast with important frequencies; i.e. as opposed to derived or abstract variables. The main advantages of this approach are that: (1) since the data represents physical quantities, it should be readily available in different software programs; and (2) since such data can also be readily obtained from building energy management systems and other data-acquisition systems, it would be relatively easy to enable run-time coupling of the simulation

environment with a real building (e.g. for control purposes) or with system components in a test-rig.

Furthermore, it is important that the data exchange mechanism should support large transmitting capacities for communicating multivariable system, when an application in building control represents several physical quantities in the same time as that system can represent heating, cooling, light, … etc.

We have considered the use of XML (eXtensible Markup Language). XML is originally designed for occasional exchange of product model data over the world wide web. Product model data exchange simplifies model construction (Xml 2003). However, in the current project, we would have to implement XML interfaces on both sides of the two different processes. We would also have to define a language for supporting the interprocess communication to manage any type of data protocol. This would make the mechanism very slow because of the two step process (between process and XML interface and between this interface and both XML formatting data and its semantics). At this time we think that XML is not really suited for intensive run-time coupling.

IPC Mechanisms for Run-Time Coupling

In principle, there exist a wide range of IPC methods and techniques applicable to many different platforms. For our research needs we have researched literature (e.g. ASHRAE 1997) and evaluated the most commonly used techniques by implementing them in a test bed with communication between distributed processes (Yahiaoui 2002). Each of the IPC mechanisms discussed below has its own advantages and disadvantages; each is the optimal solution for a particular problem that is not necessarily identical to ours. What follows is a brief overview.

- **Anonymous pipes** provide an efficient way to redirect standard input or output to child processes on the same computer. **Named pipes** provide a simple programming interface for transferring data between two processes, whether they reside on the same computer or over a network. Pipes are often referred to as a FIFO (old Unix IPC), and are very limited in data transmitting speed.
- **Messages** offer an easy way for applications to send and receive short messages. They also provide the ability to broadcast messages across all computers in a network domain. Since datagrams are used, they should not be considered a reliable means of IPC. The buffer must be synchronised so that processes can not send message at the same time.
- **RPC** (**R**emote **P**rocedure **C**alls) is a mechanism that calls a procedure from an application running on one machine to execute an another. But, RPC supports only the "procedural" integration of applications and provides neither the communication by asynchronous messages, nor the dynamic invocation.
- **Shared memory** is an efficient way for two or more processes on the same computer to share data, but these techniques require the programmer to provide synchronization between the processes. It ‘s important that the processes do not manipulate the shared memory at the same time.
- **Semaphores** are a technique for synchronising activities and also use the evolved method of shared memory. Cross-platform semaphores are incorporated to ensure that a process does not step on shared memory used by another process.
- **Sockets** are a protocol-independent interface capable of supporting current and emerging networking capabilities, such as quality of service monitoring, robust asynchronous communication, I/O completion ports for superior performance, and protocol-specific network features. In most cases, applications currently using data files for communication can be easily adapted to use sockets (e.g. Quinton 1997).
- **DCOM** (**D**istributed **C**omponent **O**bject **M**odel) is a protocol that enables software components to communicate directly over a network in a reliable, secure, and efficient manner (Microsoft 2003a). **.net** is an XML Web services platform that will enable developers to create programs that transcend device boundaries and fully harness the connectivity of the Internet(Microsoft 2003b). COM+ in general is delivered on the Microsoft Windows platform and traditionally limited to the office automation applications. DCOM does not manage the distribution of heterogeneous applications.
- **Java RMI** (**R**emote **M**ethod **I**nvocation) provides for remote communication between programs written in Java only. Can be extended to other languages (ex: C or C++) by using footbridges through JNI (Java Native Interface), or RMI/IIOP (Internet Inter-ORB Protocol), but present sometimes the intercompatibily in extension. Adapted well to all the Java applications because of its coupled integration included in Java Virtual Machine (Sun 2003).
- **CORBA** (**C**ommon **O**bject **R**equest **B**roker **A**rchitecture) - OMG (Object Management Group) develops OMA specification - provides a higher level integration than the sending traditional of raw data and not typified “TCP byte-streams”, benefits for distributed objects

similar to O-O languages for non-distributed programming and others motivations. A CORBA ORB is a federated set of services that handle a variety of run-time activities (locating, coupling, …etc). The main advantage is that there is a prevailing mindset that the insertion of CORBA into a system consists of adding another "layer". In fact, CORBA adds minimal overhead to the system, and the amount of overhead will vary between different CORBA implementations. Figure 5 shows one of the CORBA system configurations (Corba 2003).

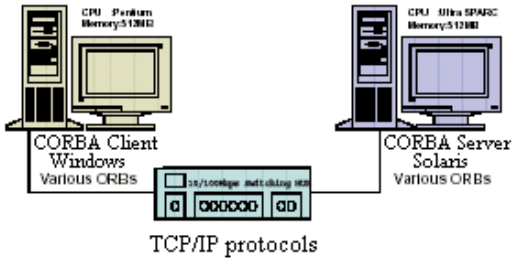

Figure 5. CORBA system configuration

Selecting an IPC Mechanism for Run-Time Coupling

After the above evaluation of the different IPC mechanisms, it may be concluded that that the most promising IPCs for the current research are sockets and CORBA.

Both sockets and CORBA use the internal operating system interface to the network (TCP/IP socket) and introduce a certain level of overhead into the method invocation. Therefore one of goals of the performance comparison was to investigate this overhead. In this test, for the client, Windows 2000 is used as the OS and Visual C++6.0 as the compiler. For the server, Solaris with Sun's native compiler are employed. For CORBA, VisiBroker-C++ (free software) is used for both the client and the server. The obtained results are discussed in the following areas.

Sockets performance results

The client makes a socket connection to the server. The threads or file transfer can handle the message transportation between client and server. The server socket writes the message in the file and the client socket reads the file created on the server-side. This file is delivered with FTP (e.g. Beveridge et al. 1997, Ralph 1996 and Coulouris et al. 1994).

It is clear that socket communication is slow. Figure 6 shows the measured time for different data sizes. It appears that some response fluctuation for socket communication is due to the use of the socket library (winsock.dll). Some events not related to socket communication itself might occur during transmissions, thus influencing the results. The transmission speeds will obviously also be affected by the actual TCP/IP implementations on the platforms involved. Finally it should be noted that some data could be lost in socket communication (pour out of the pipe).

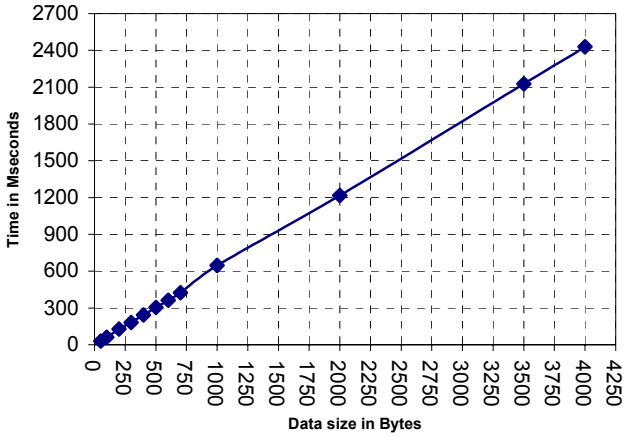

Figure 6. Response time for socket communication

CORBA performance results

This message transportation technique was implemented in CORBA for which the ORBs were on different machines. The broker is used to evaluate the interoperability among different ORBs. So one the ways to get the object reference across different ORBs is as follows: (1) Interoperable Object Reference (IOR) is changed into a string, and a server-side writes it in a file. (2) the client-side reads the file created on the server-side, and returns a string to the IOR. This file is transported with FTP (Omg, 2003 -Développez, 2002).

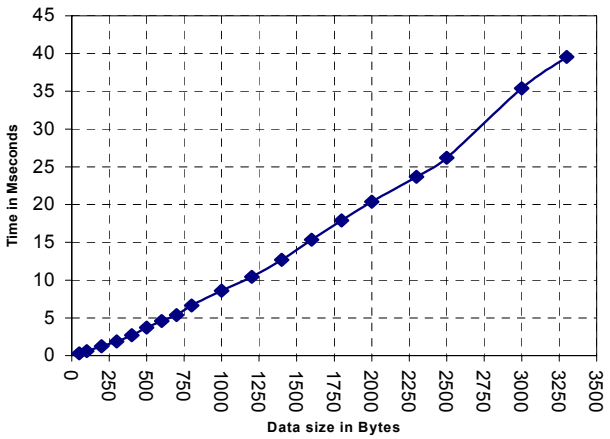

Figure 7. Response time for ORB communication

Figure 7 shows the response time for different data sizes using ORB communication. Comparison with Figure 6 shows that ORB communication is much

faster for any size of data. It should be noted that the network overhead depends on the data size. However, an optimal buffering architecture of ORBs could reduce the communication overhead considerably.

As shown, ORB communication is very fast. CORBA works well for exchange of data between different computing platforms in terms of response time and transmission capacity. ORB can even support multiple transports concurrently.

CONCLUSIONS AND FUTURE WORK

The performance comparison shows that CORBA performs better in several aspects. Our intention is to implement an IPC mechanism using CORBA by developing interfaces in ESP-r under Unix and in Matlab/Simulink under Windows. This prototype will then allow us to address important research issues such as (a) overall supervision and control of the separate evolution of each coupled application; and (b) quality assurance in terms of consistency and integrity of the overall model.

Our research will focus on the domain aspects of run-time coupling issues such as which data or information needs to be exchanged at which frequency between the separate simulation environments, and issues of data mapping between the coupled simulation environments.

In principle, data mapping can be specified in two ways. For static coupling, in which the two processes start executing at the same time and interaction between applications do not changes throughout the execution, the mappings can be specified in a configuration databases that can be read by all applications as a part of their initialization. For dynamic coupling, in which some processes may start executing later than others or the interactions between the processes change during execution, the mappings can be created and deleted as the processes are in execution. This is dynamic mapping (see e.g. Ranganathan et al., 1996).

At this stage in our research, we would like to focus on maximizing flexibility and reconfigurability rather than on specification of complex data structures.

Further objectives or our research include physical verification with validation of experimental results and utilitarian verification by means of practical application in realistic design studies.

The final outcome of this research will be a prototype system based on specific building performance and control simulation software. However, and more importantly, this research will generate associated knowledge for general and wider applicability.

ACKNOWLEDGMENT

Ir.P.J.E.M Coenen and Drs.W.J.M Lemmens of the Faculteit Wiskunde en Informatica ar gratefully acknowledged for their helpful discussions in this stage of the project.

REFERENCES

Augenbroe G.L.M. 1994. “An overview of the COMBINE project,” in Proc. of R.J. Scherer, Proceedings of the first ECPPM in the Building Industry, Dresden, Germany, pp.547-554.

ASHRAE 1997. “825-RP: A Standard Simulation Testbed for the Evaluation of Control Algorithms and Strategies,” in Proc. of ASHRAE, GA, USA.

Beveridge J. and Wiener R. 1997. “Multithreading Applications in Win32,” in Proc. of Reading, Massachusetts: Addison-Wesley Developers Press, USA.

Clarke J.A. 1986a. “The energy kernel system,” in Proc. of 2nd Int. Conf. on System Simulation in Buildings, pp. 621-635, Universite de Liege.

Clarke J.A. 1986b. “An intelligent approach to building energy simulation,” in Proc. of CICA/BSRIA Conf. on Expert Systems For Construction And Services, London.

Coulouris G., Dollimore J. and Kindberg T 1994. “Distributed Systems: Concepts and Design, 2e,” in Proc. of Reading, Massachusetts: Addison Wesley, USA.

Corba 2003. “Corba’s web page,” Object Management Group, Inc. http://www.corba.org/

Développez 2002. ”Page Web de Corba,” Club d'entraide des développeurs francophones. http://www.developpez.com/corba/index.htm

Janak M. 1998. “The Run Time Coupling of Global Illumination and Building Energy Simulations - Towards an Integrated Daylight-Linked Lighting Control Simulation,” in Proc. of IDC '98, Ottawa.

Hensen J.L.M. 2002. “Simulation for performance based building and systems design: some issues and solution directions,” in Proc. of 6th Int. Conf. on Design and Decision Support Systems in Architecture and Urban Planning,TUE, NL.

Leffler J., Fabry R.S., Joy W.N., Lapsley P., Miller, S. and Torek. C. 1993. “An advanced 4.4.bsd interprocess communication tutorial,” Technical report, Computer Science Research Group, University of California, Berkeley, USA.

Mahdavi A., Ilal M.I., Mathew P., Ries R., Suter G. and Brahme R.1999. “The architecture of S2,” in Proc. of 6th IBPSA’99 in Kyoto, International Building Performance Simulation Association.

Meystel A.M. and Albus J.S. 2002. “Intelligent Systems: Architecture, Design, and Control,” John Wiley and Sons, New York, USA.

Microsoft 2003a. “Microsoft’s DCOM web page,” Microsoft COM Technologies. http://www.microsoft.com/com/tech/DCOM.asp

Microsoft 2003b. “Microsoft’s .net web page,” Microsoft COM Technologies. http://www.microsoft.com/net/

Omg 2003. “Corba’s Basics web page,” Object Management Group, Inc. http://www.omg.org/gettingstarted/corbafaq.htm

Papamichael K., LaPorta J. and Chauvet H. 1997. “Building Design Advisor: automated integration of multiple simulation tools,” Automation in Construction, Vol. 6, pp. 341-352.

Quinton R. 1997.“An Introduction to Socket Programming,” University of Western, Ontario, Canada. http://www.uwo.ca/its/doc/courses/notes/socket

Ralph D. 1996. “Win32 Network Programming,” Reading, Massachusetts: Addison-Wesley Developers Press.

Ranganathan M., Acharya A., Edjlali G., Sussman A. and Saltz J. 1996. “Runtime Coupling of Data-parallel Programs, in Proc. of ICS’96, May 25-28, Philadelphia, PA, USA.

Sechrest S. 1993. “An introductory 4.4.bsd interprocess communication tutorial,” Computer Science Research Group, University of California, Berkeley, USA.

Sun, 2003. “Sun’s RMI web page,” Sun μ systems, Inc. http://java.sun.com/docs/books/ tutorial/rmi/

Xml, 2003. “ XML’s web page,” O'Reilly & Associates, Inc. http://www.xml.com/

Yahiaoui A. 2002. “Inter-Process Communication,” Technical Report, FAGO, Faculteit Bouwkunde, Technische Universiteit Eindhoven, Netherlands.

Zajac A. J. 1997. “Building environments: HVAC systems,” Johnson Controls Institute, USA.