DISTRIBUTED SOFTWARE APPLICATIONS ARCHITECTURE FOR INDUSTRIAL CONTROL SYSTEMS

Adrian PELCZ
Francisc SISAK
Sorin MORARU
Delia UNGUREANU

Automatics Department, "Transilvania" University of Brasov, M.Viteazu Street, no.5, 500174, Brasov, Romania, phone/fax: +40 0268 418836,
apelcz@vision-systems.ro, sisak@unitbv.ro,
smoraru@vision-systems.ro, delia@deltanet.ro

In this paper we present the architecture of a software system designated to manage industrial control systems. In industrial enterprises, the variety of control systems is wide, consisting of many distinct systems, produced/maintained by different vendors, each having it's own architecture and management problems. The solution to the problem is to have a single software system, which provides functionality for all the enterprise's resources, into a unified manner. The difficulty in implementing such a solution is the adaptation for all the existing applications to work with a new system and how should this new system be built to be able to absorb any kind of problem. The new system should provide accessibility, security and mobility as main features, and thus it should be designed as a web-oriented application.

Keywords: industrial, software, architecture, system, integration

1. INTRODUCTION

Today many industrial facilities have a lot of industrial control software systems used in various ways. These systems work using various architectures, starting from the simple "*one application on one computer attached to a process*" to more complex systems such as "*client-server-database*" or "*multi-tier*" applications. Such systems are usually provided by different vendors, this leading to the problem of managing a lot of applications, developed using various technologies and running on different environments. There are employees specialized to work with each software system, but from an administrator's point of view, having to manage many different software systems, each with it's own problems, could be a stressful task. Just managing the data backup problem for all these systems is a very complex task. Having many different applications does not imply just administration problems. There are also a lot of security aspects, as each software system has it's own security policies. Also, the information is spread into many places, this making it difficult to have a global view on the enterprise's resources.

We offer an architectural solution for this problem, which comes to meet the new demands from the informational systems of today. The trend in the industrial IT domain is guided by the emergence of more and more integrated and embedded systems, with broad access from any place (wired networks, mobile devices through wireless networks or mobile internet). Such systems offer as the most important advantage the very fast response from the people in charge. A worker having mobile

control for a system can respond almost instantly to problems, as it is in permanent contact with the system.

2. THE DISTRIBUTED ARCHITECTURE OF THE SYSTEM

2.1. Software Systems Integration

The solution to the problem presented in the introduction is to have a single software system which to provide functionality for all the enterprise, into a unified manner. The solution may sound simple, but the problem is how to convert, adapt or connect all the existing applications to work with a new system and how should this new system be built to be able to absorb any kind of problem. Our distributed software applications architecture comes to deal with these problems.

One important problem to address is how to unify the existing applications into one unitary system. Some industrial applications have the possibility to export their measured data or to import commands from external sources (like files, databases or even connectors as public interface functions). These applications are easy to integrate with other systems because their creators thought methods of external connectivity with other systems. Other applications or systems are self-contained and provide no connectivity possibilities with other third party systems. These systems are the hardest to integrate because the only possible solutions are (i) developing special connectors or (ii) completely re-implementing similar applications, which to provide the additional needed connectivity capabilities. In this case, the new developed applications can raise dramatically the costs of the integration at enterprise level, because the controlled processes can need complex software applications, and these applications can be very expensive to develop. The preferred solution is to deal with their creators to create connectors or export/import features for interaction with the new system.

Another aspect is the security. Many distributed applications have complex security policies. Integrating them into a unitary system can require the elimination or avoidance of the existing security modules, this task falling into the responsibility of the main system. As the system to be developed will need to offer wide accessibility, the embedded systems attached/contained into it will have to be accessible only through the security mechanism of the main system, and thus, direct access to them should be blocked. This problem has a complicated answer from the practical point of view, because most secured systems are based on the security module for each operation, and thus, this module can't be eliminated without major changes in the system's architecture. The problem here, just as with data exporting and commands importing, is *change*. Any change into the functionality of a system implies financial costs, which raise the total costs of the integration process, and which costs we try to avoid. So, instead of changing the security mechanisms of the existing systems, it is much more efficient to create *human simulator functions*. Such functions exchange

information with the "targeted" system just as a human operator would do. For example, the target system will not be able to distinguish between a real operator and an automated main system function when doing a login procedure.

The two aspects presented above, are a major concern when integrating software systems: *How to transfer information between the systems* and *how to get access to that information*. Basically it means how to make the two systems compatible and communicative.

2.2. The Architectural Perspective of the Solution

The solutions presented above are converging towards unifying the existing systems into one level of application. We can look at them as *data sources* or *interfaces* between the controlled processes and the new integrator system. Such an interface can provide the main system with information about the processes and also can send commands for controlling the processes, coming from the main system.

As presented in fig. 1, the architecture has three levels of application:

- The data sources;
- The server together with the database system;
- The client applications level.

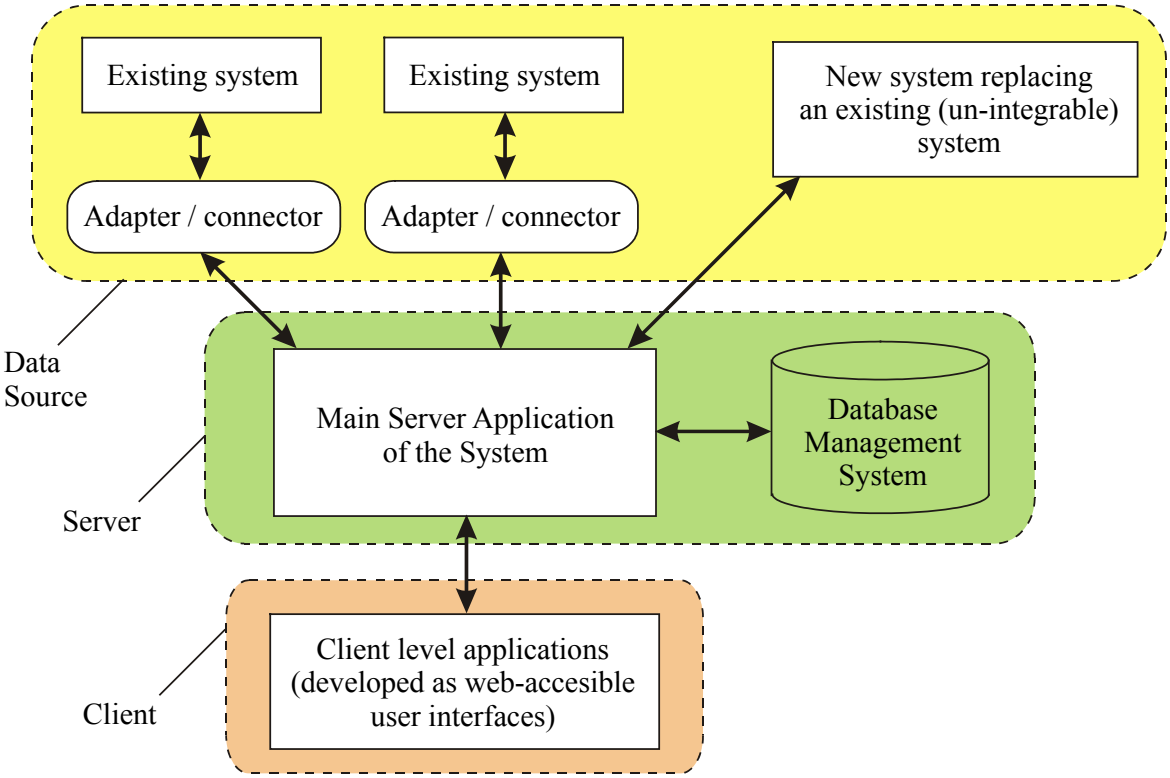

Fig. 1. Distributed Software Applications Architecture for Industrial Control Systems

The data source has two main parts. One is comprised of the existing software applications that communicate with the equipment; the other is a connector to the system. This connector has the same type of interface on the part that communicates

with the main server, but has communication methods specifically designed for each existing system on the other side. The new developed system communicates directly with the main server in the same way as the connectors do, and, on the other side, communicates with the process itself.

The server gathers all the data from the data sources, processes and stores it into the database. It is also responsible for the security of the system together with the client level. The server provides the middleware for the system, having most of the responsibilities for the functional procedures. Using the attached database management system, the server is able to provide the data processing and management needed for the entire enterprise.

The server will be responsible with data management. This is a very complex task when data amount is very large, as in the case of industrial monitoring systems. Data storage can affect adversely the performance of the system and can even cause it to crash. The large amount of data is stored in database systems, mostly for archiving purposes. Most of the time, the most recent data is actually used, the rest of it being just stored as history records. But when these history records count billions of rows in the database tables, accessing those tables for the recent measurements is strongly affected by the large number of record-sets, and the overall performance of the system is lowered. For this, the server will contain advanced data manipulation mechanisms, which will store history records in separate tables, with variable sampling times, for optimizing utility and accessibility of data (eliminating redundant data for example).

Another problem with industrial systems is error management. Errors are inherent in monitoring or commanding a process. The key element is error tolerance, which is the capability of the system to handle an error, while maintaining it is stability. To make a system fault-tolerant it is important to know the possible error sources and create algorithms for correcting or ignoring erroneous data.

The client applications are the interface of the system with the user. The client applications not only provide web access to the processes, but can also offer administrative features such as statistics, accounting figures, forecasts - all based on the massive data amounts gathered in the database from all the enterprise resources.

2. INDUSTRIAL ENTERPRISE RESOURCE PLANNING

The concept of *industrial enterprise resource planning* is starting to get shape from the ideas presented above, mostly guided by the fact that having centralized control on all the resources is a very powerful management asset. Due to evolution in technology, the industrial enterprises are forced to continuously make changes in the software-hardware tools they use in production and because the evolution comes from many sources, the technological environment is very unstable within an enterprise. To face the challenges of this technological instability, enterprises need to unify their systems for having the ability to manage the resources in an organized form.

The technology used for developing this system is web-oriented. Like this, all the data, from the entire enterprise can be viewed from any place, using the client applications, even from outside the enterprise's local area network, through the Internet. Also, with the recent advancements in mobile technology, using mobile devices connected through wireless networks will become a common practice. Mobile phones using powerful processors and bigger screens, together with operating systems and software tools, will become soon the tools used by employees to control and check the industrial processes, while not being near them.

3. CONCLUSIONS

The proposed system architecture can provide very powerful means to control the resources of an industrial enterprise, while maintaining low costs in the implementation of such a system.

Having an integrated software tool as industrial control system eliminates the problems related to administrating different solutions and reduces the effort in personnel training. The security of enterprise resources is improved by being unified into a single system, more accessible and easier to use.

The concept of industrial enterprise resource planning is getting shape, as such a system will provide the means to have all the information centralized and processed as a whole.

We are facing a new revolution in the industrial control systems, which is not guided by the technological improvements, but by the integrating power of distributed software applications.

4. REFERENCES

[1] Fielding R.T., Taylor R.N., *Principled Design of the Modern Web Architecture*, ACM Transactions on Internet Technology, Vol. 2, No. 2, pp. 115-150, 2002

[2] Bagrodia R., Phan T., Guy R., *A Scalable, Distributed Middleware Service Architecture to Support Mobile Internet Applications,* Wireless Networks 9, pp. 311-320, Kluwer Academic Publishers Netherlands, 2003

[3] Almeida J.P., Sinderen M., *Designing Interaction Systems for Distributed Applications,* IEEE Distributed Systems online vol. 6, No. 3, 2005

[4] Butterworth P., *Automating the business process of Mission-Critical Distributed Applications*, Forte Software Inc. USA, 1997.

[5] Sadowykh A., Wesner S., Bohdanowicz J.E., *A Generic Architecture for Supervision of Distributed Applications*, EuroWeb - The web and the GRID: from e-science to e-business, 2002.

[6] Bellur U., *Topology Based Automation of Distributed Applications Management,* Indian Institute of Technology, Bombay

[7] Barnett B., Kirtland M., Ganapathy M., *An Architecture for Distributed Applications on the Internet: Overview of the Microsoft .NET Framework*, Microsoft Corporation, 2003

[8] Sirer E.G., Grimm R., Bershad N., Gregory A.J., McDirmid S., *Distributed Virtual Machines: A System Architecture for Network Computing*, University of Washington Seattle, 1998.