A Cell Controller Architecture Solution: Description and Analysis of Performance and Costs

António Quintas [1] , Paulo Leitão [2]
[1]Professor Associado do DEEC da Faculdade de Engenharia da Universidade do Porto, Rua dos Bragas, 4099 Porto Codex; Tel: 351.2.2041844; email: aquintas@fe.up.pt
[2]Assistente da ESTG do Instituto Politécnico de Bragança/CCP, Quinta S[ta] Apolónia, Apartado 134, 5300 Bragança; Tel:351.073.3303083; email: pjl@ccp.up.pt

Abstract

Nowadays, with the globalization of the markets, the companies must to implement new manufacturing technologies and concepts for the improvement of manufacturing systems productivity. The integration is one of the keys to solve the problems in manufacturing systems. The arrangement of the resources in cells and the implementation of control systems allows the improvement of productivity, flexibility, quality and the reduction of production costs.

This paper describes a Cell Controller architecture solution, implemented at the flexible manufacturing cell integrated into the demonstration pilot of an ESPRIT Program Project; special attention to the control structure and the system communication is taken. The last part of the paper focuses the costs involved in this solution and presents the alternative platforms and technologies to implement the Cell Controller.

Keywords

Cell Controller Architectures, Production Technologies, Industrial Communications, MMS.

1. INTRODUCTION

A Flexible Manufacturing Cell is a group of processing resources (PLC, NC machines), interconnected by means of an automated material handling (AGV, robots) and storage system, and controlled by a control system [Groover]. This control system is called Cell Controller, and it's a module of software that performs the integration of the several cell devices, allowing more flexibility and the improvement of production. The Cell Controller should be able to perform the following functions [Rembold, Groover]:

- **production control** - management of the cell production;
- **real time scheduling of production** - reaction in real time to the cell capacities;
- **resource management** - optimal use of materials and devices;
- **NC and RC programs management** - storage and download of NC and RC programs;
- **device monitoring** - visualization of cell and devices status;

- **error monitoring** - reaction to fault conditions.

The design and the implementation of a Cell Controller is a complex task, involving real time control restrictions and the manipulation of different operating systems, different communication protocols and machines supplied from different vendors.
The specification of a Cell Controller architecture results into the definition of three structures: control structure, communication system and information system.

The control system is very important for the final performance of the Cell Controller and it's structure can be performed with some basic control architectures: centralized, hierarchical and heterarchical [Diltis]. With the improvement of the performance/price relation for computing processing power, the architectures evolued from centralized architecture to the heterarchical architecture, allowing the distribution of the cell controllers functionalities in several hierarchical control levels.

The communication system is crucial to implement the integration of the cell resources, as to transfer the NC and RC programs to the cell resources and also to control these devices. There are several protocols available to implement the communication system: MAP (Manufacturing Automation Protocol)/MMS (Manufacturing Message Specification), FieldBus, serial link, TCP/IP. The choice of a communication protocol depends upon the costs, the standardization, and the control degree.

Following the paper, a Cell Controller architecture solution is described, with the analysis of his performance, and then, other platforms for the Cell Controller are discussed and a comparative performance evaluation is presented.

2. FLEXIBLE MANUFACTURING CELL DESCRIPTION

The manufacturing cell has two CNC machines and an anthropomorphic robot for the load/unload of the machines. One of these machines is a turning center *Lealde TCN10*, with a SIEMENS *Sinumerik 880T* controller; the other machine is a milling center *Kondia B500* model, with a FANUC *16MA* numerical control.

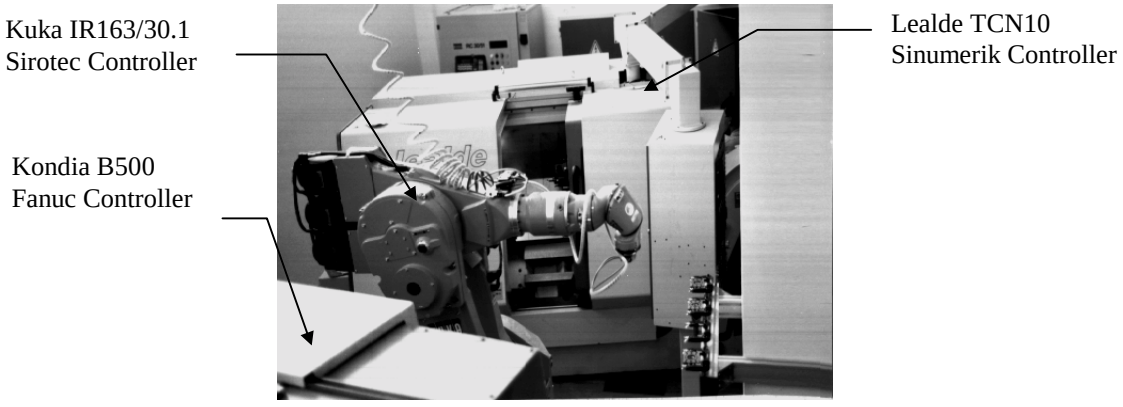

Figure 1 - Manufacturing Cell Layout

The robot is a KUKA *IR163/30.1* with a SIEMENS *RC3051* controller. The manufacturing cell has two transfer tables for the containers loading and unloading. These containers bring the material to be operated into the cell and take away the pieces produced.

The Cell Controller was developed and implemented in a Sun SparcStation 10 workstation with Solaris 2.4 Operating System. This workstation has a network card with two stacks:

- **TCP/IP stack**, for the communication with the Shop Floor Controller and the Project Department. This network allows the transmission of NC and RC programs from fileserver to the industrial machines;
- **OSI MAP stack**, for the communication with industrial machines, like NC machines and robots.

The manufacturing cell is connected to the controlling room by a LAN with a bus structure topology, based on a base band transfer media (10Mb/s). The LLC protocol used is 802.3 (Ethernet CSMA/CD). All the machines have MAP interface boards. These interfaces are: **CP 1476 MAP** for Siemens Sinumerik 880T machine controller, **CP 1475 MAP** for Siemens Sirotec robot controller and **GE FANUC OSI-Ethernet Interface** for GE Fanuc 16MA numerical controller.

Flexibility and open systems concept lead us to the application of an open systems communication standard protocol at the manufacturing process control level. MMS is a standardized message system for exchanging real-time data and supervisory control information between networked devices and/or computer applications in such a manner that it is independent from the application function to be performed and from the developer of the device or application. MMS is the international standard ISO 9506 [ISO/IEC 9506-1], based upon the Open Systems Interconnection (OSI) networking model - it's a protocol of the application layer of the OSI model. It runs on the top of an OSI stack providing a set of 80 services distributed for ten MMS functional classes.

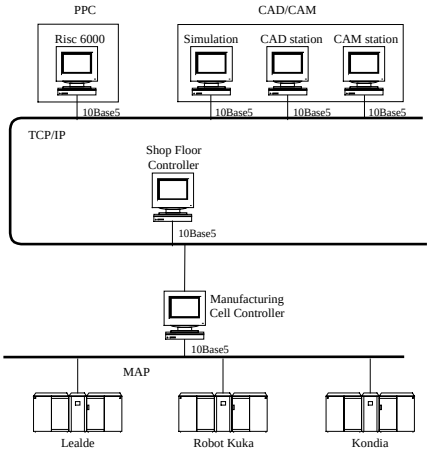

Figure 2 - Shop Floor Communication Infrastruture

3. MANUFACTURING CELL CONTROLLER ARCHITECTURE

The definition of control system structure for Cell Controller was done keeping in mind two important aspects:

- **modular structure**, which allows the future expansion of the control system, for instance, if the number of machines grows up;
- **real time requirements**, to guarantee that control system is able to execute all required functions performing the time restrictions, for example the cooperation between machines.

The specified control system for this Cell Controller is a blend of centralized and hierarchical architecture. It's a hierarchical architecture because the functions are distributed by several control levels; however, it's not completely hierarchical because the lowest levels may execute jobs and services but they are not able to execute the complete manufacturing orders.

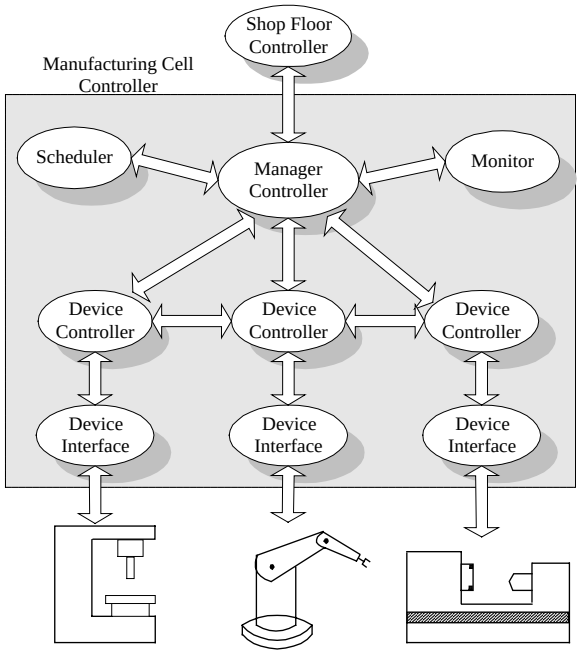

Figure 3 - Cell Controller Structure

This structure has three hierarchical levels, being each one responsible for the execution of control functions. The first level is designated by Manager Module, and it's responsible for the planning and the control of the cell production. In the second level, there are several modules to control industrial machines. Each of these modules, designated by Device Controller, is customized to the industrial machine and it has the responsibility for the execution of the jobs in the machines. Finally, the last level, the Device Interface, contains the interface between the Cell Controller and each of the industrial machines. The communication protocol implemented in these three modules is the MMS protocol.

This control structure is modular and flexible, allowing easily the expansion of the Cell Controller if the number of machines in the cell grows up. When a new machine is added to the manufacturing cell, it's only necessary to add a Device Controller module and a Device Interface module, customized to the new machine.

All Cell Controller code is written in C language and the communication between modules, inside the Cell Controller, is implemented with a Unix Operating System functionality, called pipes, which is made up of files to exchange the messages.

3.1 Manager Module

This module is the brain of the manufacturing cell controller, and it´s responsible for the control and the supervision of the production process of the manufacturing cell and also for the management of cell resources. The main functions of this module are:

- Start the Cell Controller, verifying if it´s a normal or abnormal start;
- Receive and process the messages from the Shop Floor Controller;
- Receive and process the results of services executed by the Device Controllers;
- Determine for each order, the next operation to be executed;
- Dispatch the orders to the Device Controllers;
- Notify the Shop Floor about the evolution of the orders and whenever an alarm occurs;

The management of the manufacturing cell is based upon information stored in the cell database. This database contains information about orders, resources, cell buffers and tools stored in the industrial machines of the cell.

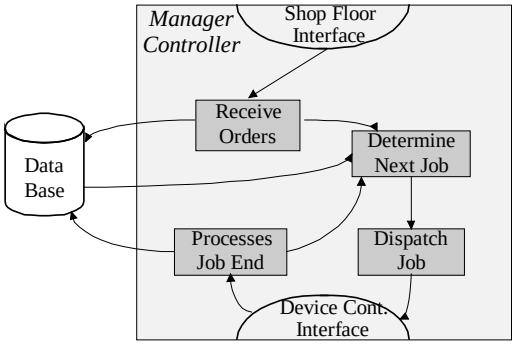

Figure- 4 Manager Module Structure

The Manager Module is a set of several small modules, each one responsible for the execution of different functions. The Shop Floor Interface and the Device Controller Interface send and receive messages to the Shop Floor Controller and from the Device Controller, respectively. The other four small modules execute the management of the manufacturing orders.

3.2 Device Controller Module

The Device Controller is responsible for the local control of the machine, and for the execution of the jobs requested by the Manager Module. This module receives jobs from the Manager Module and separates them into elementary services, to be executed by the Device Interface module in the industrial devices. For example, the execution of a machining program requires the execution of two services: the transmission of a NC program from fileserver to the NC machine, and the start of a NC program.

When all services are completed, the required job is concluded and this module sends a message to the Manager Module indicating the end of the job. If an error occurs in the industrial device, this module sends also an error message to the Manager Module.

3.3 Device Interface Module

The MMS protocol implements the interface communication between the Cell Controller and the industrial devices. The key features of MMS are:

- **the "Virtual Manufacturing Device" (VMD) model**. The VMD model specifies the objects contained in the server, and with MMS services it´s possible to access and manipulate these objects;
- **the client / server model**. The client is a device or application that requests data from the server; the server responds with data requested by the client.

There is a distinction between a real device (e.g. PLC, CNC, Robot) and the real objects contained on it (e.g. variables, programs, etc.), and the virtual device and objects defined by the VMD model. Each developer of a MMS server device or MMS server application is responsible for "hiding" the details of their real devices and objects. The executive function translates the real devices and objects into the virtual ones defined by the VMD model.

In many cases, the relationship between the real objects and the virtual ones can be standardized for a particular class of device or application (PLC, CNC, Robots, PCs). Developers and users of these real devices may decide to define more precisely how MMS is applied to a particular class of device or application - the result is a Companion Standard.

The Device Interface module is responsable for the execution of services, required by the Device Controller, on the industrial devices. The MMS objects implemented in the server application (industrial device) and accessed by the client module (Device Interface module) are:

- **MMS Domains**, used for the standardized transmission of data or programs;
- **MMS Program Invocation**, used to manipulate NC/RC programs;
- **MMS Variables**, used to map typed data (e.g. Tool offsets, NC data, PLC data, etc.);
- **MMS Events**, reporting the end of program execution.

When the service is finished, this module sends a message to the Device Controller indicating the end of service execution. An error message is sent if a fault occurs in the machine, for example, a tool collision.

4. SOLUTIONS FOR SOME PROBLEMS

4.1 Execution of a Machining Program

The execution of a machining program requires the execution of a set of services: decoding a NC program, download the program to the machine and start the program by the machine. When the project team designs a product and generates the NC programs for the manufacturing of the product, it's not possible to know where the tools are stored in the machine. Therefore, the NC programs must contain the indication of the tool's type. However, the CAD postprocessor can only insert a code with two digits in the number or type of tool. Thus, it's necessary an auxiliary file to associate the tool types and the tool codes (two digits) that appears in the NC machine.

Original Program

```
MPF 1000
. . .
T22 D22
. . .
T4 D4
. . .
M30
```

Auxiliary File

```
4 fresa20
. . .
22 roca20
```

Tools Table

Tool Type	Position
drill10	1
fresa20	2
drill4	6
probe	10
roca20	12

Final Program

```
MPF1000
. . .
T12 D12
. . .
T2 D2
. . .
M30
```

Figure 5 - Decoding of a Machining Program

This limitation of CAD post-processor requires a decoding of a NC program before the download of the program to the machine. This function is implemented by a filter algorithm.

The filter algorithm searches the character "T" in the NC program; the next two digits are the tool code. It's necessary to know what is the tool type for this tool code, by reading the auxiliary file. After, it's necessary to verify if there is a tool inside the machine with the type required, by seeking the tools table. If there is a tool stored in the machine with the type of tool required for the operation, the tool code of NC program is replaced by the position of the tool inside the NC machine. When the filter algorithm finishes the changes in all lines of the NC program, this NC program is ready to be downloaded to the NC machine.

4.2 Execution of Setups

The execution of a new production requires the update of machine configurations. This operation is designated by setup and can contain the following tasks:

- change the tool stored inside the machines;
- change the gripper of robot;
- execute the machine maintenance;
- test the RC programs;
- change the machine parameters.

During the setup, the Cell Controller must execute a set of functions: transfer the tools offset to the machines and update the tools table with the parameters of the new tools. The transfer process is different for each machine:

- **Lealde machine** - the Device Controller makes the upload of tools offset from machine to a file. For each position, the old parameters are replaced by the new tools offset. When the file is completed, this file is downloaded to the machine through MMS Download service.
- **Kondia machine** - it's necessary to build a data structure with the following information: offset number, geometrical dimensions, radius, wear and tool life. This structure is transferred to the machine through MMS Write service.

5. PERFORMANCE ANALYSIS

5.1 Data Transmission rates

The execution of a program in a industrial device requires the existence of this program inside the device memory. Due to the memory capacity limitation, it´s necessary to transfer the program to the device whenever it will be necessary. This operation is performed by the Cell Controller with the MMS Download service.

During the Cell Controller implementation tests, and with a network analyzer, it was possible to work out the transfer speed: 10 kbits/s. This speed is very slow and origines the Cell Controller lost of efficiency. For example, the Cell Controller spends approximately 30 seconds

to download a NC program with 50 kbytes! This time is not profitable in the processing tasks, and causes the lost of productivity in the manufacturing cell.

With this slow transfer speed, it was necessary to know where the problem was located. After several tests, it could be concluded that the problem was not associated with MMS protocol, because two MMS applications, running at SUN worksation, could communicate with 10 Mbits/s transfer speed; thus, the problem is related to the internal machine problems. In fact, when a machine receives a program, this is analised inside the machine, line by line, to detect errors inside the program. This procedure causes the reduction of data transmission from 10 Mbits/s to 10 kbits/s.

For increasing the Cell Controller efficiency, it´s necessary to do the management of the NC programs inside the machines memory and download the needed NC programs into the Cell Controller, making use of it´s dead times. This is a complex solution to be implemented, and requires the introduction of Artificial Intelligence techniques inside the system control. The use of multi-agents systems in the Cell Controller architecture can be a good solution, to solve this problem and, in addition, to help the fault recovery tasks.

5.2 Costs of the solution

The performance of an integrated solution is analyzed into two points: operationality and cost. The cost is an important factor that will be decisive for the viability of a solution. It is possible to quantify the cost associated with this Cell Controller architecture solution, by analyzing the costs of the several platforms and interface boards used to implement the Cell Controller. These costs can be divided into two main areas: the communication protocols and the developing platform.

To implement a communication system with MMS protocol, the cost is obtained by adding the cost related to the MAP/MMS interface boards and the cost of the communication software. It´s possible to calculate the cost of the communication system implemented in the Cell Controller. The cost of the MAP/MMS interface boards is around $6000 (US dolars) each one and the MMS software (MMSEASE from SISCO in this case) around $6700. In this application, the number of MAP/MMS interface boards is 4 (3 servers/machines and 1 client), so it´s necessary to spend $30700 to implement the communication system. For a better perception of this cost value, it is possible to make a comparation with the cost associated with the resources of the cell (two NC machines and one robot). The cost of these three machines is around $333300. So, the communication system cost value is approximately 9,2% of the costs associated with the resources of the cell.

With this scenario, it can be concluded that it´s very expensive to use a standardized communication protocol like MAP/MMS, basically because the price of the interface boards are very expensive. In alternative, other communications protocols could be implemented for the communication system: FieldBus and Serial link, for instance.

The FieldBus interface boards are cheaper, because they only implement three levels of OSI the stack. The FieldBus is a standardized protocol, like MAP/MMS, and uses the RS485 protocol to implement the physical layer. This protocol is used to interconnect the sensor and actuator devices. The Serial link is very cheap, but it requires the development of an API (Application Program Interface) for each industrial device, customized to the device funcionalities. For a higher number of devices, this task is complex due to the different funcionality of each device. Another disadvantage of the Serial link is the connection point to point between devices and the controller station, that is worst for the expansion of the system and also for the machines which are distant from the controller station. Solutions which only

transfer programs to the devices required from device operator, is a good answer for applications with low control degree or with manual production, which only requires the download of NC or RC programs.

Another disadvantage of this Cell Controller solution is the developing platform. The cost of the developing platform is $15340, due to the cost of the SUN SparcStation 10 and the developing software SUN SparcWorks. These costs can not be reached by SME (Small and Medium Enterprises), which also demands for new platforms like Windows NT. Actually, the companies need cheap control systems solutions and short developing time. The Windows NT platform allows the reduction of hardware costs and the use of more userfriendly programming tools, like Visual Basic or Visual C++.

An important problem with the Unix platforms is the monitoring or graphical interface that is very complex to implement. The use of Visual C++, for instance, allows the reduction of development costs and a more quick implementation of the solution for the desired application, using all the potentialities of these programming tools.

6. CONCLUSIONS

The cell controller architecture presented in this paper is now working well. The two major aspects of a Cell Controller specification were focused: the control structure and the communication system. In the control structure, the use of hierarchical architectures is growing up, mainly due to the increasing of computing system capacities and the reduction of the prices. The control structure presented in the Cell Controller solution is a blend of a centralized architeture and a hierarquical architecture, allowing the flexibility and modularity of the application. The communication system could be implemented with MAP/MMS, FieldBus, Serial link, between others. The protocol presented is the MAP/MMS due basically to it´s standardization and the simplicity of it´s implementation.

With the analysis of the MMS implementation, it was possible to conclude that the costs of the MMS interface boards are the limiting factor to the implementation of MMS protocol in the integrated applications. Nevertheless, the MMS protocol is a good solution for applications which require a total control of resources or when the integration involves a high number of resources with high degree of complexity. For cheaper solutions required by SME´s, it´s necessary to use other protocols, like serial link and field bus. These solutions are a valid alternative to implement the communication system of Cell Controller.

The Cell Controller architecture described in this paper, is developed in an Unix platform, and despite the good results of his operationality, it´s necessary, in the future, to build the Cell Controller on a cheaper development platform, like Windows NT, and also it is wise to use the Artificial Intelligence methods to perform the control and the faults recovery tasks.

7. REFERENCES

Diltis, D., Boyd, N., Whorms, H. (1991) *The evolution of control architectures for automated manufacturing systems*, in Journal of Manufacturing Systems, Vol 10 Nº1, pp 63-79

Lee, K., Sen, S. (1994) *ICOSS:A two-layer object-based intelligent cell control architecture*, in Computer Integrated Manufacturing Systems, Vol 7 Nº2, pp 100-112

Pimentel, J. (1990) *Communication Networks for Manufacturing*, Prentice Hall

Groover, M. P. (1987) *Automation, Production Systems and CIM*, Prentice-Hall
ISO/IEC 9506-1, (1992) *Industrial Automation Systems - Manufacturing Message Specification, Part 1 - Service Definition*
Rembold, U., B.O. Nnaji, B.O. (1993) *Computer Integrated Manufacturing and Engineering*, Addison-Wesley
CCE-CNMA 2, (1995) *MMS: A Communication Language for Manufacturing*, ESPRIT Project CCE-CNMA 7096, Volume 2, Springer