Design of a Supervision System for Integrated Software Architectures in CIM
= A Framework for Concurrent Engineering =

Luis Antonio Freixo Guedes Osorio
Luis Camarinha-Matos
Grupo de Robótica Inteligente
UNINOVA - Universidade Nova de Lisboa
Quinta da Torre - 2825
Monte Caparica
PORTUGAL

Paulo Eigi Miyagi
Jose Reinaldo da Silva
Lab. de Automação e Sistemas (Mecatrônica)
Universidade de São Paulo
BRAZIL

Abstract: This paper describes a topic of an on-going bilateral research cooperation between UNINOVA and USP in the context of the ECLA FlexSys project and presents a plan of future actions. The topic is a supervision system design for an integrated federation of heterogeneous and distributed CIM software modules towards a platform for Concurrent Engineering. The work plan makes for combining the UNINOVA/UNL research in Systems Integration and the extended Petri nets PFS/MFG dynamic modeling approach of USP. The Editor / Simulator of MFG graphs developed by UNINOVA, used in the supervision of an integrated software architecture for CIM is being extended to fit PFS/MFG specifications and be connected to an analytical module. The analytical module under development at USP is intended to upgrade the supervision system with structural and behavioral analysis functionalities. The setting up of a concurrent platform needs some support to coordinate the activity sequence.

1. Introduction

Any effort made to develop a true Computer Integrated Manufacturing (CIM) system, has to highlight such aspects as Information Modelling and Information Integration and Exchange among heterogeneous and distributed computational agents. These two aspects get additional connotations like the definition of common modeling methodologies, modeling languages, information repositories (persistency), information exchange protocols, and others. There is, therefore, a need for a concertation of different cultures which nowadays provide concurrent and non-compatible solutions to solving partial problems of an enterprise.

A CIM system implies the integration of a large number of heterogeneous software modules, running on a distributed and heterogeneous computational infrastructure [3][2]. From an Information System perspective, the integration of a set of heterogeneous tools/subsystems requires various problem-solving modalities, including: i) definition of common models for shared concepts, in order to support an effective exchange of information; ii)development /acquisition of an Information Management System (IMS); and iii) in conjunction with the IMS, setting up an integrating infrastructure that offers a functional support to integration, i.e. a sort of "software bus" offering high level interprocess communication services to the connected tools.

From a functional perspective, the analysis of a CIM system could make each application tool be modelled as an activity box whose inputs and outputs represent tools' access to the Information System (IS) (and Integrating Infrastructure). In other words, in heterogeneous multi-module systems each component has large autonomy and only "contact points" - i.e. data Inputs and Outputs - are important for a global co-ordination. Therefore, a first attempt at controlling the system could rely on these I/O's, i.e. monitor tools' behavior by the information they handle.

To build an integrated system is not merely guaranteeing that its component modules are able to communicate and share information. In other words, it will not do to provide a "software bus" and standardize information models to get a consistent system out of a set of components. Even though such aspects are vital, it is also necessary to establish a **supervision or control architecture** that checks on how and when tools get access to the infrastructure and modify shared information.

In our opinion, the definition of such supervision architecture is a basic requirement to support

Concurrent Engineering (CE). The control over information exchanges between application tools (software modules implementing a set of enterprise activities) and the CIM Information Management System (IMS) is one possible of way of managing multiple activities running concurrently in the distributed computational environment of a manufacturing enterprise.

A general aspect covered by Concurrent Engineering and widely dedicated research work is the need for providing a well-established co-operation framework for the engineering activities involving interdisciplinary contributions. An engineering team is a group of experts in different areas doing their work with a set of specialized software modules and exchanging, during developing phase, partial results among them.

In order to implement a control strategy for such a set of software modules, the dynamic behavior of the system being controlled, should be modelled. In the case of a CIM software architecture, as it is assumed to be a multi-purpose system, capable of carrying out different tasks, it is necessary that the running processes and their execution stage be represented.

For this purpose, in our current experiment, the CIM-OSA concepts of Business Process and Enterprise Activity [7] are being used. The dynamic behavior is thus modeled by hierarchies of Business Processes (BP) and Enterprise Activities (EA). At each level of the hierarchy, a Procedural Rule Set (PRS) defines precedence rules between BP or EA at that level as well as their starting (firing) conditions.

Once defined a "global plan" with the correct sequencing of activities and the executor (Application/Tool Activity) assigned to each activity , it is necessary that a co-ordinator synchronizes tasks among them. Such control strategy will impose functional constraints on each engineering or other enterprise activity by enabling or disabling access to the IS, depending on their current execution status.

Another important formalism to model a system dynamic behavior, which manufacturing systems make extensive use of, is Petri nets (in various derivations) [13]. In order to take advantage of tools and methods available for Petri nets, namely in qualitative and quantitative analyses, the attempts at deriving PN models from BP/EA/PRS are motivated. In our approach we are using a derivation of the basic PN model called Mark Flow Graph (MFG) [5] [6]. To put it differently, we start from a hierarchy of BP/EA, as a "global plan" whose execution has to be controlled, and automatically derive a MFG representation from it.

Petri nets, as well as its derivation, MFG, takes an important role in analysis and modeling. It can be applied in association with behavioural analysis (simulation) and/or with structural analysis, where the most important objective aimed at is to find out the properties which are essential to a class of systems. Most frequently, behavioural analysis deals with the construction and observation of all states which can be attained from a specific initial one, what yields explosive algorithms.

On the other hand, if an application area is well-known (or even if it is concerned with standardization and normalization of activities in some area), it is very important to translate such standards into structural properties associated with a MFG or Petri Net and Computer Aided System. In this work we propose to insert a structural analysis based on an extended MFG in an IMS (Information Management System) created by UNINOVA, which now uses a behavioral standard MFG simulator. A workplan of our co-operation activity is presented in Section 4. Section 2 describes the UNINOVA/UNL approach and Section 3 focuses on the PFS/MFG methodology developed by USP.

2. UNINOVA/UNL Approach

2.1 Integrating Infrastructure

The Engineering Information Management System (EIMS) developed in the context of the ESPRIT Programme CIM-PLATO and JNICT SARPIC/CIM-CASE projects, provides an integration infrastructure and an object oriented information management system to support common concepts shared among a set of heterogeneous and distributed software modules (Tools) of which specific activities contribute to the enterprise functional needs [1][2].

EIMS offers a list of information management services that are remotely available to client software modules via a library of Information Access Methods (IAM). An overview of the main blocks of EIMS is given in Figure 1. This IAM library hides or encapsulates aspects of remote calls, through the network, to the EIMS services and is available for computational platforms which have Remote Procedure Calls (RPC) services. Complementarily, a loose connection mode via STEP neutral files is also available. These capabilities are nevertheless insufficient to obtain a truly functional coherent system. The set of

activities performed by each contributing tool should be controlled by some co-ordination mechanism. Thus, UNINOVA/UNL group has been working on improvements to such integrating platform. They have developed an extended EIMS with supervision and control mechanisms to guide the integration and inter-operation of such a set of enterprise software modules. One important aspect is the clarification of relationships among enterprise needs or goals and the activities that support/implement them. Another aspect is the way such activities are related to real executing agents, like software modules able to perform the required functional needs.

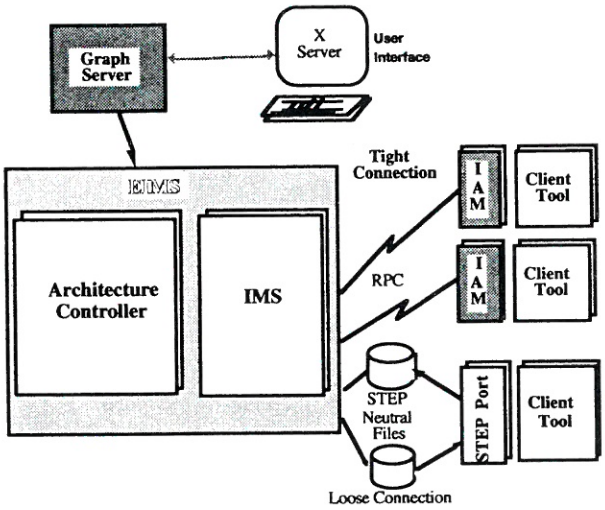

Figure 1 EIMS Implementation Architecture

2.2 Enterprise Functional Model

CIM-OSA is one of the proposals for a global framework to model a complete CIM system, starting from the business requirements and ending at shopfloor realization and control level [7].

The CIM-OSA- proposed functional decomposition, is based on a hierarchy of business processes (BP), modeling different levels of enterprise objectives. Each detailed BP includes a set of enterprise activities (EA) which implements its objectives. Precedence among BP/EA is expressed in terms of procedural rules sets (PRS). In our approach each EA is decomposed into a set of sub-activities, each one being implemented by some Tool Activity (TA). A TA is a functionality implemented by a software package and contributing to meet global/partial EA needs. Figure 2 shows a partial example of such decomposition. The business process BP1.6, representing the enterprise goal "produce P1", is partially implemented by two enterprise activities: EA1.3.8 as a design resource, and EA1.3.9 as a planning one. The design activity takes a two-step CAD, CADx provides a first rough design for market evaluation and once approved, the final product is designed in CADy.

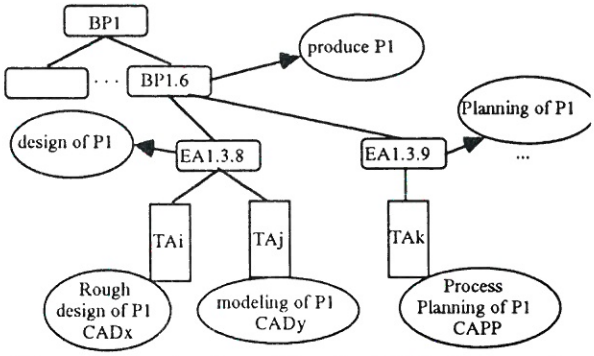

Figure 2 Production of Product P1 in Terms of BP/EA

The reality of a production system is, of course, much more complex than the example in Figure 2. However the need to integrate all contributions in order to establish the final product shape, is related to the reduction of design/engineering/production cycle times and to the improvement of the product quality and to easy to manufacture facility. Concurrent Engineering team members should have flexible mechanisms to cooperate when they are doing their work with the help of software tools. In this direction, the presented model is intended to obtain a flexible unification of enterprise activities, offered as capabilities to implement the business processes, and provide a correct execution of planned functionalities.

Software modules or tools are progressively becoming multi-functional packages able to perform/support multiple activities. This tendency justifies the representation of a tool as a set of activities or Tool Activities (TA), which provides a correspondence between enterprise needs and implemented functionalities of selected software packages.

So, from the EIMS point of view, there is a set of heterogeneous and distributed Tool Activities that can do a planned work in a concurrent way, i.e. activities are fired by users, on unpredictable times, during enterprise life cycle. It is the responsibility of EIMS to control the coordination of all activities and to limit access if conflicting interdependencies exist.

2.3 Control of Enterprise Activities

The "global plan", expressed in terms of BP/EA/TA and the PRS, defines the interdependencies among activities. During system execution (Business Process Interpretation), tools implementing such activities, access IS to retrieve Inputs or to store

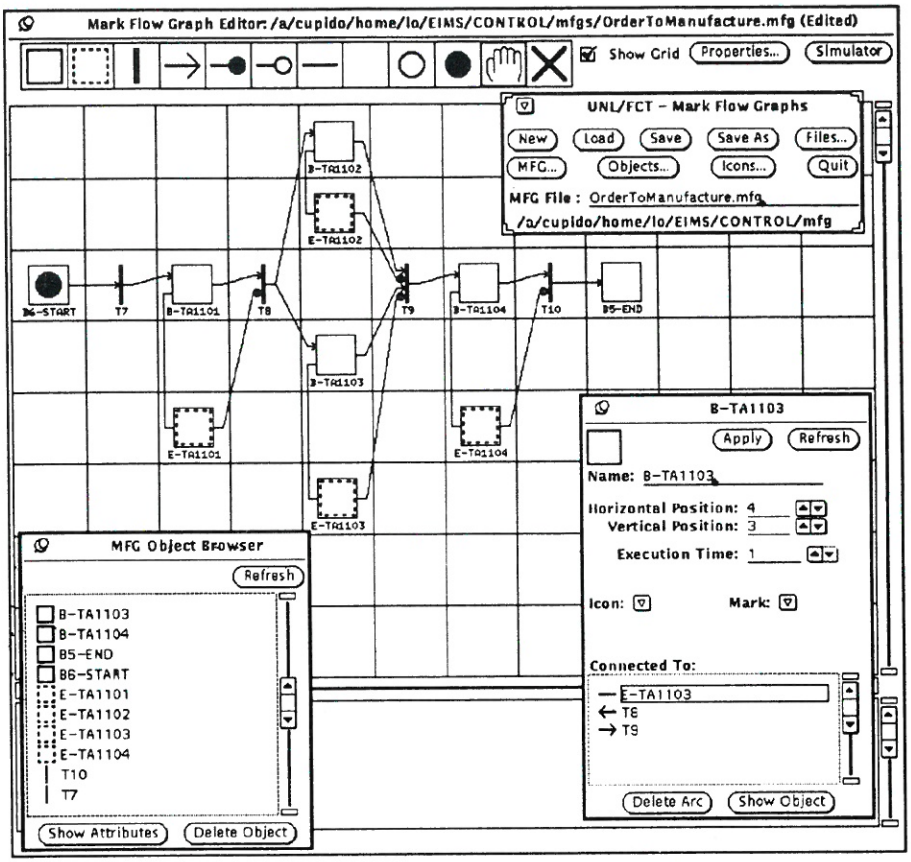

Figure 4 . Editor and Executor of MFG

Outputs. For instance, when a tool's user finishes some activity and fires the transaction with EIMS the outputs can be sent to IS. The procedural rules defined during functional modelling are a good support in supervising and controlling the execution of the planned activities. Nevertheless, it is not possible to learn more about the dynamics of the system, i.e. to know if some activity cannot definitely access EIMS while another one is waiting for its results. To provide analytical capabilities to deadlock situations like the mentioned one, an event/condition extended Petri net graph, Mark Flow Graph (MFG), was chosen. The purpose of such a modelling language is to systematically follow the EIMS state evolution and to structurally analyse deadlocks prevention.

Each BP, EA and TA in the "global plan" is represented by an internal box. A mark in a box means that the corresponding activity can be executed, i.e. all previous activities are finished with the appropriate ending contributions. When a mark arrives to a box, a signal is sent to an external box (machine) representing its supporting tool. When a mark appears in the external box it means that the event "activity concluded" happened and that the next activities could start.

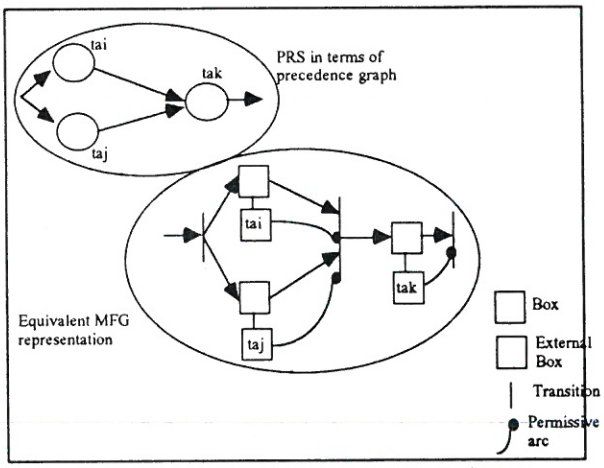

Figure 3 . Mapping of PRS to MFG Graphs

The decision of putting a mark in the external box is a hard task because there is no "sensorial" information in a control, as it is in a robotic cell. EIMS control should decide on the conclusion of some activity, through a deductive process based on the analysis of Inputs/Outputs and on specific knowledge as to each particular activity.

UNLmfg was developed as a tool to model and simulate discrete and concurrent event systems. The tool is composed of two main parts: i) editor and ii) simulator.

The editor was developed in BIMProlog and takes advantage of XWindows graphical facilities. The user can interactively design an MFG network by putting new objects (box, external box, transition, direct arc, inhibitive and permissive arc and output arc), assigning icons to each box or external box, labels to box, external box and transitions and defining initial marking. At any moment the user can change the layout of objects by moving them along the drawing area.

By selecting a simulator, the user can realize, in either step- by- step or run mode, the flow of marks along the network. He can also insert breakpoints to inspect some abnormal situations. The events can be delayed by assigning times to activities (duration of conditions). When a box has a time greater than zero, as soon as a mark arrives (during a transition fire) that mark will only be available after the conclusion of the corresponding activity.

Including time in this implementation was possible by generating an intermediate marking state which corresponds to marks that arrive at boxes but are no more available for contributing to enabling the next transition.

This approach can lead to structural analysis problems. The intermediate state is not explicitly modeled but results from an assertion to boxes. In any case we can see this elapsed time not as an intermediate state but as a delay in the previous transition firing. A mark takes some time to achieve boxes with associated time greater than zero.

2.4 Improvements of Dynamic Model

Property analysis of MFG graphs, generated by precedence graphs of activities, was not implemented in the initial prototype. Nevertheless, it is important to include such analytical capabilities in order to obtain a deeper knowledge of inter-related activities and to avoid activity sequences causing to deadlock situations. In this direction and taking advantage of the co-operation strategy of FlexSys project, an agreement was reached between UNINOVA/UNL and the group of Mechatronics of University of São Paulo (USP) to study the enhancement of UNINOVA/UNL proposal with analytical facilities, in order to process, at a first stage MFG graphs, and at a second step the extended MFG/PFS (see point 3). A first stage of co-operation work includes the general problem analysis and the definition of common information models to represent MFG objects. During this stage, drawing up concepts should be unified, and the detailed implementation plan for a demonstration prototype should follow.

3. MFG/PFS Methodology

3.1 Concepts

Mark Flow Graph (MFG) is an extended net formalism derived from Petri Nets, which has become popular in several areas of modeling and analysis of discrete systems.

However, as any other net approach, it has the advantage of being visual and very suitable for engineers and researchers to use in order to formalize models and simulations, including themselves a proper interpretation of the elements. Although very convenient, such an approach raises the problem of expressiveness, that is, in order to represent large systems it is necessary to draw up and manage a very large net schema.

The expressiveness problem inspired researchers to look for extended nets such as MFG. This particular net has been successfully used in Japan to represent and design FAS . It has special elements which represent sub-nets allowing in their turn the representation of abstract static elements. It can be mapped into a programming C-like language which is very handful to simulate the models.

PFS (Production Flow Schema) is an extension that allows real abstract levels in MFG. The MFG/PFS methodology consists of a top down design method to the design of any discrete event system, in particular FMS and FAS.

A hierarchical approach was developed based on C/E (Condition/Event) nets which includes abstract elements called *activities* [4] standing for an entire sub-net.

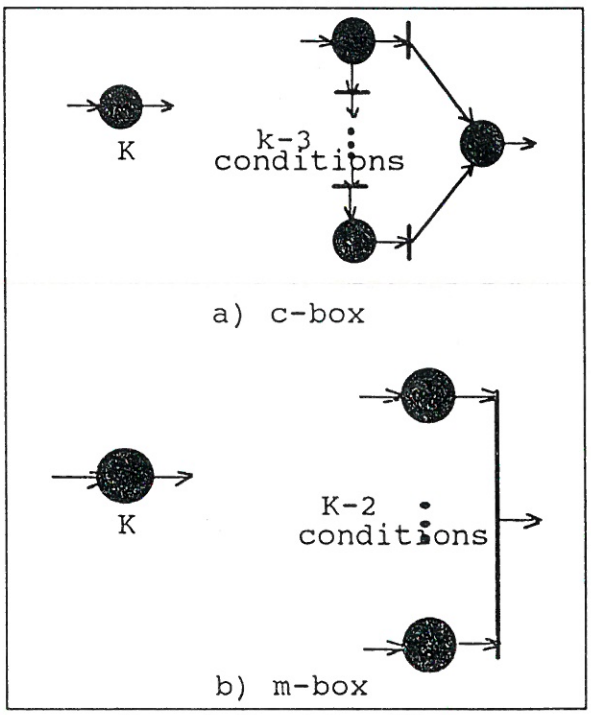

Figure 5 . Description of Boxes

The MFG/PFS [4] is an extended net composed of high level macro-elements such as *activities* and boxes. Each *activity* can be replaced by a sub-net starting and ending with an event, while a box can be replaced by a sub-net starting and ending with conditions (see Figure 5)[1]. Another special MFG element is the gate, which consists of a kind of flow or relation between conditions and events that comes from an "external condition". An "external condition" is a condition that is not part of the system, that is, the situation where it holds is not within the control of the system. Reset buttons, power switches, quality assurance stations, and other anthropomorphic control mechanisms are examples of "external conditions".

A review of the formalism of MFG/PFS was proposed by [11] where a new state equation, very close to the one for elementary nets, describes system behavior. The main idea is to treat the property analysis, either structurally or behaviourally. In [11] the concept of *activity* was extended to cover the idea of *event,* while *conditions* would circumscribe several kinds of boxes (capacity boxes, assembly boxes, distributors, etc.) and *gates* were included in the general notion of flow.

The state equation of the extended net is :

$$M_{i+1} = M_i + A^T \sigma_i - G^T \sigma_i \qquad (3.1)$$

where M is the state of marks, A is the incidence matrix of the net, σ_i is the vector of enabled transitions connected to the current state, and G is the matrix of gates. The extra term simply rejoins the persistent marks (corresponding to the state of pseudo-conditions associated with gates).

Equation 3.1 above shows that it is possible to simulate the dynamic behaviour of discrete systems by calculating the new states analytically. Also, the analysis of properties should proceed based on such a equation, and can be very useful in the process of analysis of an integrated software architecture.

Invariant analysis, for instance, is a very good example of how this formal methodology can be applied.

A set of marked conditions is called a *case*. Cases are representations of discrete states. Thus, it is possible to represent restrictions, that is, states that should be avoided, by a case which does not change its token count during events and activities firing. Such states would never be enabled, but if they were, a deadlock situation were present.

Therefore, the invariant analysis is also useful in deadlock prevention, which certainly is the most important goal in practical applications.

Synchronism is another important property. Synchronic distance can give the right metric scheduling to a process, and can be analysed during nets simulation [10].

Several other properties can be used, even if their interpretation, that is, the association of meaning in real problems, is not evident (coverability, B-fairness, etc.). For these and for the properties mentioned above, it is important to count on computerized systems which can manage part of the formalism, leaving the human designer free to fulfill the interpretations. In the sequel we will mention some applications developed by the USP/LAS in Brazil.

3.2 Available and Prospective Analysis Tools

The first modeling tool developed at LAS was the AIPUSP system which includes a Prolog analyzer of MFG/PFS in the original version [4].

An upgraded version is being developed to run on Sun workstations, allowing tagged elements, that is, including attributes associated with elements.

Another system was created to run on personal computers, that implements and analyses extended nets as defined in [11]. In such a system it is possible that non-sequential applications programming parallel processes should be analysed. It also includes an analyser of invariants and synchronic distance. A new version is being prepared to run on workstations, including other properties as well.

Two research lines at LAS are compromised with tools based on MFG/PFS: the first one is concerned with the application of nets to control algorithms envisaging Flexible Manufacturing Systems. The second line explores the capabilities of MFG/PFS as a framework for the design process applied to FAS. On this line the methodology is used as a support to intelligent CAD systems.

4. Workplan

Co-operative activities were set out according to the efforts made by each group in modelling discrete event systems. The approach of concepts and

[1] We are actually admitting the existence of two kinds of sub-nets: those which resemble active elements - as milling machines or any kind of transformation of material - and passive sub-nets like assembly process or storing, where no transformation takes place.

techniques by each group seems to be a good way of improving research results of both sides. The workplan includes two main activities. The first one is related to the unification of concepts and techniques used in each group. To improve communication, working out a common glossary will be important. The second co-operation aspect is related to concrete co-operation work producing common reports and implementing prototypes to validate the obtained results.

This research cooperation work will be mainly supported by electronic exchange of ideas and results, common publications and exchange of people.

Given these general ideas, the following four points can make the co-operation work.

1. Unification of concepts between the two partners and definition of a conceptual mapping between the UNINOVA MFG editor/simulator and the extended MFG/PFS defined by USP.

2. Definition of common representation models of MFG/PFS concepts.

3. Construction of an elucidative example showing the application of the integrated systems, used in the supervision/co-ordination of a federation of heterogeneous software modules in a CIM environment.

4. Demonstrator planned to show the results of co-operative activities of the two research groups.

Acknowledgments

The work here described was developed in the context of the ECLA CIMIS.net and FlexSys (European Communities), SARPIC and CIM-CASE (JNICT) projects.

REFERENCES

1. OSÓRIO, A.L. and CAMARINHA-MATOS, L.M., **Information Based Control Architecture for CIM**, IFIP - Towards World Class Manufacturing'93, Phoenix, Arizona, USA, September 12-16, 1993.

2. CAMARINHA-MATOS, L.M. and OSÓRIO, A.L., **CIM Information Management System: An Express-based Integration Platform**, IFAC, Espoo, Finland, 23-25 November 1992.

3. CAMARINHA-MATOS, L.M. and SASTRON, F., **Information Integration for CIM Planning** Tools, CAPE'91 - 4th IFIP, Bordeaux, France, 10-12 September 1991.

4. MIYAGI, P.E., **Control System Design, Programming and Implementation Systems for Discrete Event Production Systems by Using Mark Flow Graph**, Ph.D Thesis, Tokyo, Japan, 1988.

5. MIYAGI, P.E., HASEGAWA, K.and TAKAHASHIT, K., **Mark Flow Graph (MFG) para Modelamento e Controle de Sistemas de Eventos Discretos**, Monografias em Automação e Inteligência Artificial, EPUSP, Vol.1, N°1,1989.

6. MIYAGI, P.E., **Modelagem e Controle de Sistemas Produtivos: Aplicação da Teoria de Redes de Petri**, Lectures, 1°Simpósio de Automação Integrada, CEFET - PR, Curitiba, Brazil, July 1990

7. AMICE, **Esprit Consortium AMICE, Open System Architecture for CIM**, SPRINGER-VERLAG, 1989.

8. MURATA, T., **Petri Nets: Properties, Analysis and Applications,** The IEEE Proceedings, Vol. 77, No.4, April 1989.

9. REISIG, W., **Petri Nets: An Introduction**, SPRINGER-VERLAG, Berlin, Heidelberg 1985.

10. GOLTZ, U. and CHONG-YI, Y., **Synchronic Structure,** Lecture Notes in Computer Science, Advances in Petri Nets, SPRINGER-VERLAG, 1985.

11. SILVA, J. R. and PESSOA, F.J.B., **Analise Semi-Automática de Mark Flow Graphs**, Ibero-American Workshop on Autonomous Systems, Robotics and CIM, Lisbon, 1992.

12. VALETTE, R. - **Les Réseaux de Petri** Lecture Notes, C.N.R.S./LAAS. Toulouse, France, May 1990.

13. ZHON, M. and DICESAR, F., **Petri Net Synthesis for Discrete Event Control of Manufacturing Systems**, KLUWER ACADEMIC PUBLISHERS, 1993.