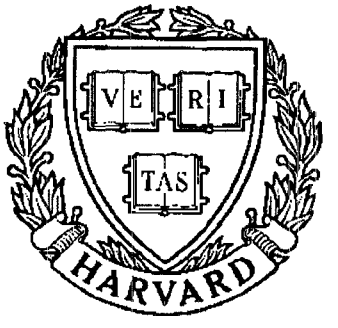
TECHNICAL RESEARCH REPORT

SYSTEMS RESEARCH CENTER



Supported by the National Science Foundation Engineering Research Center Program (NSFD CD 8803012), Industry and the University

INformation Systems for Integrated Manufacturing (INSIM)

by G. Harhalakis, C.P. Lin, L. Mark, and P.R. Muro-Medrano

TR 92-15

INformation Systems for Integrated Manufacturing(INSIM)

G. Harhalakis, C.P. Lin, L. Mark

Systems Research Center
University of Maryland
College Park, Maryland 20742
USA

P.R. Muro-Medrano
Department of Electrical Engineering and Computer Science
University of Zaragoza C/María de Luna, 3, 50015 Zaragoza
SPAIN

Abstract – A mechanism with the potential to control the information flow among all of the manufacturing application systems, in order to streamline factory activities, based on company-specific and company-wide policies and procedures is proposed here. The goal is to achieve a fully integrated manufacturing management system. The INformation Systems for Integrated Manufacturing (INSIM) reflects a design methodology to build a knowledge base to serve as the control mechanism.

1 INTRODUCTION

Current research in the area of manufacturing systems software is quite intensive in dealing with product and process design, production planning, and job execution. However, the design of such systems has been traditionally made in a functional fashion that emphasized "local" solutions, using closed and self-contained architectures. This, together with the use of heterogeneous databases and incompatible computer operating systems, have led to "islands of automation" of various engineering application systems. Naturally, these systems suffer from data inconsistencies and lack of control of functional interactions between them.

Current and future trends for the use of computers in manufacturing include the control and the integration of information flow for production operations into a computer-controlled factory management system. Various research projects in the area of Computer Integrated Manufacturing (CIM) have been conducted by NIST [1] [2], ESPIT [3] [4], CAM-i [5], and AT&T [6]. Most of research projects emphasize on individual aspects of CIM, such as RPI [7] on developing a global database framework, TRW [8] on synchronizing the interface between application systems and distributed databases, and U. of Illinois [9] on developing a framework to perform common manufacturing tasks such as monitoring, diagnostics, control, simulation, and scheduling. These approaches are developing a generic CIM architecture, by creating a global database framework, or by interfacing shop floor activities. However, our approach is to develop a control mechanism for managing and controlling the information flow among all the manufacturing application systems, and to fill the gap between high level production management and low level factory automation [10]. Our work is unique in that it addresses the control and the management aspect of information flow, while other research projects aim at developing a consistent database framework or a standard communications protocol for data transformation. Our control mechanism over existing distributed database management systems is capable of achieving a fully integrated manufacturing information system.

The second section presents our INformation System for Integrated Manufacturing (INSIM), its architecture and specifications. The third section details the design methodology and the implementation strategy of our knowledge base, and provides examples of the interfaces built between CAD/CAPP/MRP II/SFC. The last section presents our conclusions with recommendations for future work.

2 FUNCTIONAL SPECIFICATION OF INSIM

In this paper, we present a mechanism for the control

of information flow between each of the key manufacturing applications software at the factory level, including Computer Aided Design (CAD), Computer Aided Process Planning (CAPP), Manufacturing Resource Planning (MRP II), and Shop Floor Control (SFC) systems. This linkage is based on data commonalities and the dynamic control of functional relationships between these application systems. The common data entities, which form the basis of the integrated system, can be classified in two categories: Static and Dynamic. The former define the various entities of the distributed system such as parts, products, equipment and processes, while the latter deal with the functioning of the system as it operates to satisfy the market demand.

Our goal is to demonstrate the viability of achieving the integration and the control of information flow, using generic operations on generic entities. In order to remain as general as possible, this model does not emulate any particular CAD, CAPP, MRP II and SFC packages.

Our CIM architecture concentrates on the integration of manufacturing applications at the Factory level (similar to the Facility level of the NIST hierarchy) as depicted in figure 1. CAD, CAPP, MRP II, and SFC can be integrated together through a general Distributed Database Management System (DDBMS). The Knowledge Based System, which is the subject of our research, drives the DDBMS to control the information flow, following procedural rules, constraints and other guidelines derived from the company policy. In order to build a prototype of the CAD/CAPP/MRP II/SFC integrated system, we have defined data structures of the common data entities involved in the various manufacturing applications of our integrated system and their relations, which are stored in the DDBMS. Therefore, it can be said that the management and control of information flow is performed by the KBS, while the integration aspect is addressed by the DDBMS.

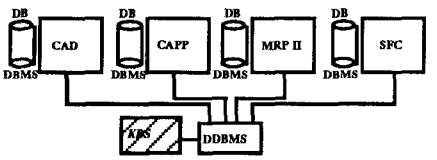


Figure 1: Overall CIM Information Flow Architecture at the Factory Level

3 KNOWLEDGE BASE DESIGN METHODOLOGY

The proposed methodology for the design and maintenance of a Knowledge Based System(KBS) to control the functional relationships and information flow within the integrated system is a major task of this research and is depicted in figure 2.

It starts from user-defined rule specifications , reflecting a specific company policy, which is then modeled using a special set of Colored Petri Nets - UPN(Updated Petri Nets) and a hierarchical modeling methodology, discussed in section 3.2. The next step is to convert UPN models into General Petri Nets (GPN) for validation purposes, and feed the results back to the user to resolve (i) conflicting company rules and (ii) errors introduced during the modeling phase. This process is described in section 3.3. After the model has been validated, a parser translates the UPN model into a rule specification language, briefly described in section 3.4. The end result is a software package that controls the data-flow and accessibility between several databases.

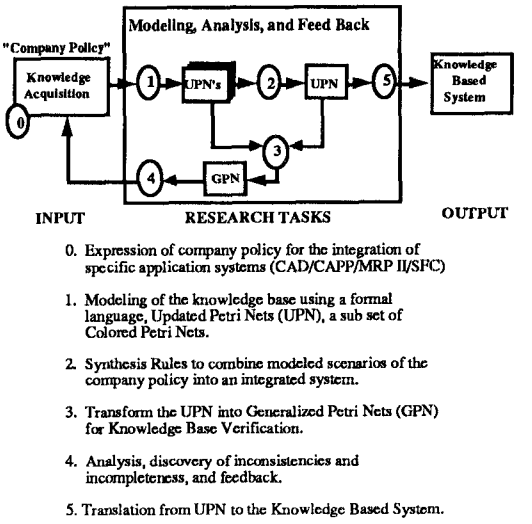


Figure 2: Knowledge Base Design Methodology

3.1 Knowledge Acquisition (Company Policy)

The design of the model is based on the desired information flow between all the manufacturing applications, namely CAD/CAPP/MRP II and SFC. As mentioned above, this information flow is driven by the expert rules embedded in the knowledge base, extracted from company expertise, policies and procedures, which can be obtained through a number of individual interviews and group meetings with experts from all manufacturing application systems to be integrated, and managers responsible for making company policy. Therefore, substantial effort may be required for gathering all expert rules to form the knowledge base. However, since we are here to develop and demonstrate a generic design methodology, our prototype will only include limited rules extracted

from our own industrial experience and other industries involved with this and other projects in the CIM Laboratory.

The development of the knowledge based system usually starts from designing a set of abstract rules which are then refined to more detailed rules for this initiation and maintenance of specific entities within the system. A set of scenarios under each entity which represent these abstract rules, include creation, update and deletion of parts, bills of materials, work centers, routings and work orders.

3.2 Modeling of the Knowledge Base

Although General Petri Nets [11] initially adopted in this research can in principle handle the modeling of the knowledge base, it has become necessary to define a large set of of semantics in order to handle the increasing complexity of the system, due to the involvement of more applications and their entities. Hence we have started developing a new generation of Colored Petri Nets, called Updated Petri Nets (UPN), together with a hierarchical modeling methodology and a systematic approach for the synthesis of separate nets with common places. Some of this work was inspired by [12].

3.2.1 Evolution of Updated Petri Nets

Special features of Updated Petri Nets (UPN),which reflect a special type of the Colored Petri Nets (CPN) [13], and a hierarchical modeling methodology with a systematic approach for the synthesis of separate nets with common places. The use of UPN allows the model designer to work at different levels of abstraction. Once we have this net we can selectively focus the analysis effort on a particular level within the hierarchy of a large model. We use UPN in modeling not only the rule base, but also the database changes which ensure the consistency in representing the database status.

We have extended the primitives of the classical CPN descriptions in order to reflect more closely the terminology and semantics involved in our application domain. These primitives do not contribute with new concepts to Petri Net theory (in the sense that they do not increase its analytical capabilities) but allow for a procedure to automate and formalize the interpretation process of the model to a rule based system.

An UPN is a directed graph with three types of nodes: places which represent facts or predicates, primitive transitions which represent rules or implications, compound transitions which represent meta-rules (sub-nets). Enabling and causal conditions and information flow specifications are represented by arcs connecting places and transitions.

An UPN is represented as follows:

$$UPN =< P, T, MT, C, Ic^-, Ic^+, In^-, In^+, M >$$

where

- $P = \{p_1, ..., p_n\}$ denotes the set of places (represented graphically by circles)
- $T = \{t_1, ..., t_m\}$ denotes the set of primitive transitions (represented graphically by black bars)
- $MT = \{mt_1, ..., mt_l\}$ denotes the set of compound transitions (represented graphically by blank bars)
- $C = \{C(p_1), ..., C(p_n)\}$ denotes the coloring; ie., each $C(p_i)$ denotes the set of data associated to place p_i.
- $Ic^-(p,t)$ and $Ic^+(p,t)$ denote the set of conditional arc functions, and $In^-(p,t)$ and $In^+(p,t)$ denote the set of non- conditional arc functions.
- $M = [m_j]_{n \times 1}$ denotes a marking, where the j-th component m_j denotes the color and number of tokens on place p_j. M_o denotes the initial marking.

We have divided the representation of UPN components in the following four groups: *Data, Facts, Rules, Metarules. Data* and relations between different data are used in relational database management systems. *Facts* are designed to declare a piece of information about some data, or data relations in the system. The control of information flow is achieved by *Rules.* Here, we are considering domains where the user specifies information control policies using "if then" rules. Rules are expressed in UPN by means of transitions. Any transition t has a data set, $C(t)$, associated with it. Metaknowledge and hierarchical net descriptions are represented by *Metarules* (compound operations) and will be detailed below.

Metarules are mainly used in UPN as a mechanism to define sub-nets. They are used in two different directions to allow a structural and hierarchical composition of the domain knowledge:

- **Horizontal**
  Rules at the same level of abstraction can be related to form sub-nets. This horizontal composition allows the aggregation of rules under specific criteria.
- **Vertical**
  The vertical top-down decomposition of rules is used to establish relations between one rule and other rules which define knowledge at a lower level of abstraction. Therefore, vertical decomposition in UPN allows a structure of rules that form an abstraction hierarchy. This abstraction method makes

the design and verification process easier by allowing the designer to follow a stepwise refinement process working at different levels of detail (see section 3.2.2).

3.2.2 Modeling Methodology

Generally speaking, any "company policy" starts from the specification of general global rules which describe aggregate operations for a given entity within the system. These rules are then further refined into more detailed specifications on a step by step basis, until no aggregate operations are left [12] [14]. Following a similar concept, a hierarchical modeling method has been developed in UPN which allows the system designer to start from abstract global nets and continue with successive refinements until the desired degree of detail has been reached. In addition to the refinement of rules within each scenario, a technique is needed to synthesize all the scenarios to form a coherent net representing the company-wide policy for all entities in the system.

This hierarchical modeling methodology facilitates the modeling task. It incorporates:

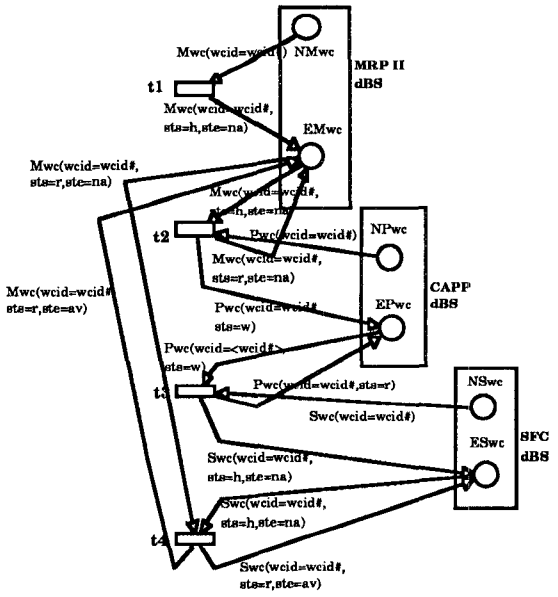
A top-down stepwise refinement technique for the modeling of each scenario from an abstract and aggregate level to a detailed level. The design process for each scenario (each set of functional relations) starts from an abstract net with both primitive and compound transitions, and continues by exploding each compound transition to corresponding sub-net until no compound transition exists.

A synthesis technique for synthesizing separate nets, which represent different scenarios of the system, to form a coherent net. A contribution of our research in modeling is to incorporate the modeling of the databases of the manufacturing application systems involved, using UPN, by defining the database states as global variables (places), and by synthesizing nets through these places systematically . This enables the structuring of Petri Nets in a progressive manner, and facilitates the transformation of the "company policy" of figure 2 to a formal model.

3.2.3 Examples

Top Down Refinement - Establishing new work centers in the system via MRP II.

This scenario composes one metarule by itself, which in turn involves four metarules: "insert a work center in MRP II", "release a work center in MRP II", "release a work center in CAPP", "release a work center in SFC" - connected to each other using the horizontal composition technique. Each of these metarules is then refined to one



t1: add a new work center record in MRP II
t2: release the work center in MRP II
t3: release the work center in CAPP
t4: release the work center in SFC

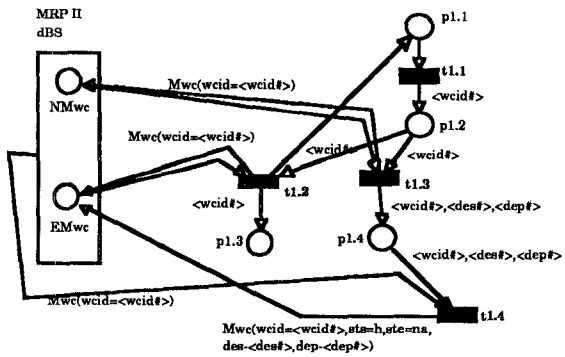
Database place notations
N'X'wc, E'X'wc
N = does not exist
E = exists
'X' = database: M -> MRP II, P -> CAPP, S -> SFC
wc = work center

Figure 3: UPN graph of the Scenario: "Create a work center via MRP II" at an abstract level.

sub-net and connected to the higher level net, based on the vertical composition formalism.

The above example modeled by a UPN graph is shown in figure 3. An abstract company policy, which represents the scenario "create a work center via MRP II", is modeled with compound transitions (blank bars) (t1) "insert a work center in MRP II", (t2) "release a work center in MRP II", (t3) "release a work center in CAPP", (t4) "release a work center in SFC", and includes the sequence and constraints among these compound transitions.

One of these compound transitions (metarules), "insert a work center in MRP II" shown in figure 4 is further refined into a lower level sub-net, through the vertical decomposition technique. This sub-net contains four primitive transitions (t1.1) "request Wcid", (t1.2) "output error message", (t1.3) "request Des and Dep", (t1.4) "add work center record into the MRP II DB", which are connected using the horizontal composition technique and can not be further refined. The partially refined net



t1.1: user enter the work center number
t1.2: work center number check fails and repeat request
t1.3: work center number check succeeds and
request more information
t1.4: add work center record in MRP II

NMwc: work center does not exist in the database
EMwc: work center exists in the database
p1.1: MRP II user starts the insert wc transaction
p1.2: work center id number is provided
p1.3: all the necessary dat are provided
p1.4: message print out to ask another wc id

Figure 4: Sub-net of the scenario "Insert a work center in MRP II".

is shown in figure 5.

The connections between higher and lower levels are made through the identified **Input** and **Output** transitions [15]. In this example the **Input** transition is $t_{1.4}$ and the **Output** transition is also $t_{1.4}$. Thus, all the input places of transition t_1 are connected to $t_{1.4}$ and all the output places of transition t_1 are connected from $t_{1.4}$. The result of refining t_1 (figure 4) of figure 3 is shown in Figure 5.

Net Synthesis - Establishing and Deleting Work Centers in the system via MRP II. (refer to step 2 in figure 2)

The two scenarios, creation and deletion of work centers via MRP II modeled in UPN are shown in figures 3 and 6 respectively. Synthesis is made by merging one place into another place if the interpretations of the two are identical, or if the interpretation of one place is included into the interpretation of the other place. In our case, it is the database places which form the basis of common places. The resulted UPN of synthesizing the nets of figure 3 and figure 6 is shown in figure 7.

In the above example, the company policy, which involves the scenario "create a work center via MRP II" and "delete a work center from MRP II", is now represented by a single net. This net retains the dynamic behavior of both scenarios and reflects the relationships between them. Following the same synthesis technique one by one for all scenarios , the result is a unified net representing the formal model of the company-wide policy.

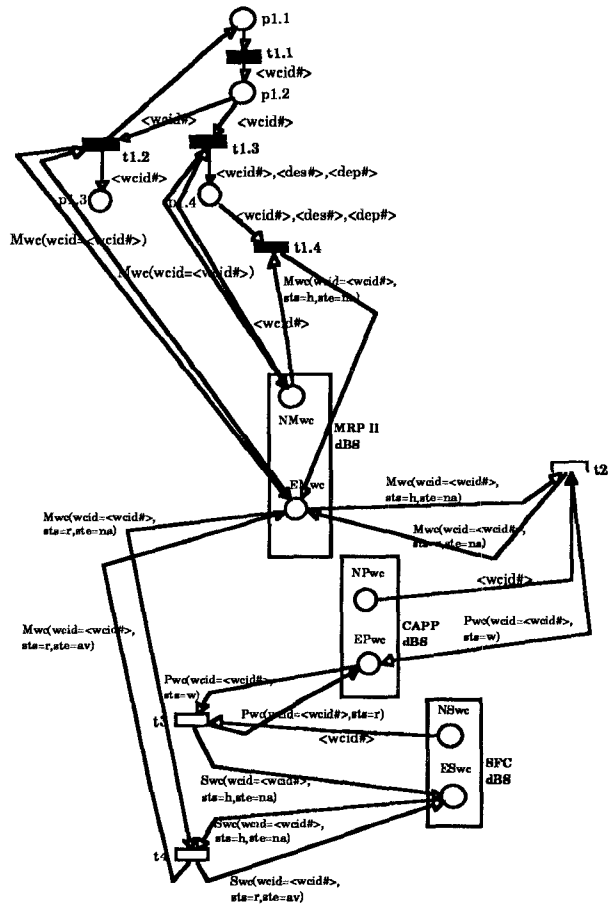


Figure 5: Partially refined UPN of the scenario "Create a work center via MRP II".

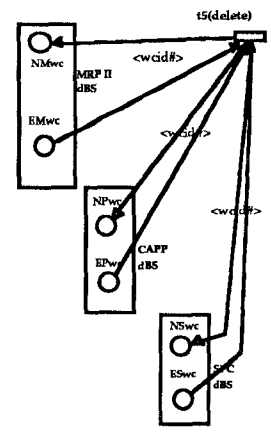


Figure 6: UPN graph of the scenario "Delete a work center via MRP II" at an abstract level

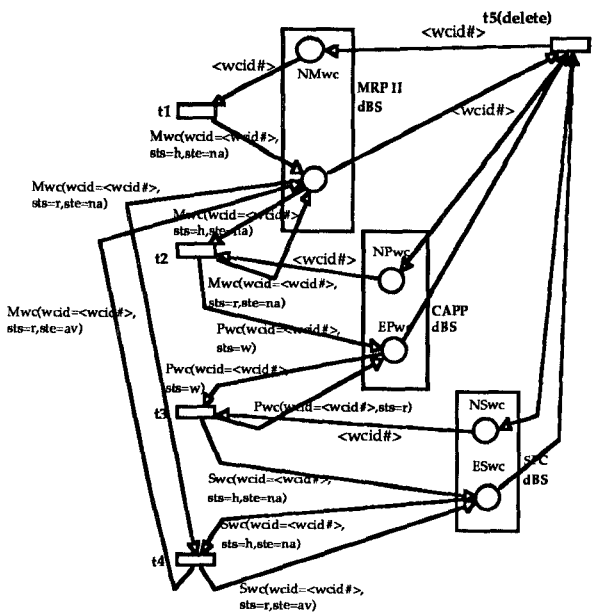


Figure 7: UPN graph of the synthesized Scenario "Create and Delete a work center via MRP II" at an abstract level.

3.3 Knowledge Base Verification

The major objective of creating a knowledge based system using Petri Nets is the ability of validating the KBS systematically. The *completeness* (dead-end rules, unfirable rules), *consistency* (redundant rules, subsumed rules, under-constrained rules), and *conflicts*, are the major issues in knowledge base validation [16], [17]. The incidence matrices of Petri Nets representing the rule base can be used to perform some of these validation checks and verify them with the aid of specific domain knowledge. Several other analysis techniques for Petri Nets, including, reachability trees, behavioral nets, and net invariants, are also used [11] [18]. The net invariants, which represent mutually exclusive conditions within the "company policy", can reveal logical conflicts in the specification of the original rules and possibly errors introduced during the modeling process. The reachability tree can be used to detect any deadlocks or inconsistencies in the model. The behavioral net can be used to detect redundancies in the net and is a useful tool for reducing the size of the model. The programs for computerizing these analysis methods have been developed and applied extensively. Some reduction rules [19] have also been investigated to reduce the size of nets prior to the analysis phase [15].

However, these analysis techniques were initially developed for Generalized Petri Nets (GPN), and do not apply to Colored Petri Nets (CPN) which are characterized by a great diversity of linear functions that are associated to their arcs. Therefore, unlike analysis algorithms for GPN that use integer matrices, analysis algorithms for CPN need to manipulate matrices composed by linear functions. This fact introduces difficulties in the development and execution of these algorithms. Due to this fact, a necessary step must be made to convert the high level net of the UPN model into a low level net expressed in terms of Generalized Petri Nets (GPN). We call this conversion process: "*abstraction* and *unfolding*". As the first step of the conversion process, we make abstractions of the UPN to reduce the complexity of the net due to so-called non-fixed data sets, such as work center identification numbers. More specifically, we eliminate the colors of different work centers under the same operation and only consider one "generic" work center for the analysis of the knowledge based system. The unfolding consists in taking the information that is aggregated in data structures, data sets and arc functions, and create a new net where this information is split in more places and transitions, as many as the number of colors. The arcs of this net have associated integer numbers (but no linear functions) and there exist just tokens without any colors. The result of this conversion process will be analyzable GPN [15] (Refer to step 3 in figure 2).

3.4 Implementation

The implementation phase is being realized through the development of the Update Dependencies language in the Department of Computer Science, at the University of Maryland [20]. (Refer to step 5 in figure 2) Database interoperability can be described as the concatenation of the schemata of each of the databases of the application systems, along with a rule set constructed for each separate database, called update and retrieval dependencies. These update and retrieval dependencies control inter-database consistency through inter-database operation calls. We are using Update Dependencies (UD) as a special rule specification language for the implementation of our Knowledge Based System. The algorithm for the automatic translation between the UPN and the UD language has been developed [15], which reduces substantially the implementation effort.

4 CONCLUSION

The design and maintenance methodology - INformation Systems for Integrated Manufacturing (INSIM) - is fairly generic in that it can be applied to generate knowledge based systems for various applications. Depending on the rule specification language used, this design methodology can be applied in the same way except using a different translator to translate the UPN model to that language. Another feature of INSIM is that it can be used dynamically to introduce updates and/or additions to the company policies and procedures as they evolve. Future

work includes the extension of the synthesis techniques for modeling more complicated scenarios of the company policy, and the development of new techniques for validating rule bases with more complex structures.

REFERENCE

References

[1] A.T. Jones and C. Mclean, "A Proposed Hierarchical Control Model for Aotomated Manufacturing Systems," *Journal of Manufacturing Systems, vol. 5, no. 1*, 15-25(1986).

[2] W.J. Davis and A.T. Jones, "A Real-time Production Scheduler for a Stochastic Manufacturing Environment," *Int. J. Computer Integrated Manufacturing, vol. 1, no. 2*, 101-112(1988).

[3] A. Bonnevie and P. Krzesinski, "A Double Approach for Analysis and Design of Production Systems," *ESPRIT'86: Results and Achievements*, 469-478(1987).

[4] W. Meyer, "Knowledge-Based Realtime Supervision in CIM - The Workcell Controller," *ESPRIT'86: Results and Achievements*, 33-52(1987).

[5] G. Chryssolouris, "MADEMA: An Approach to Intelligent Manufacturing Systems," *CIM Review*, 11-17(Spriing 1987).

[6] R.L. Franks, J.P. Holtman, L.C. Hsu, L.G. Raymer and B.E. Snyder, "Productivity Improvement Systems for Manufacturing," *AT&T Technical Report, vol 66, issue 5*, 1987.

[7] C. Hsu, C. Angulo, A. Perry and L. Rattner, "A Design Method for Manufacturing Information Management," *Proceedings of Conference on Data and Knowledge Systems for Manufacturing and Engineering, Hartford, Connecticut*, 93-102(1987).

[8] M., Sepehri, "Integrated Data Base for Computer Integrated Manufacturing," *IEEE Circuits and Devices Magazine*, 48-54(March 1987).

[9] S.C.Y. Lu, "Knowledge-Based Expert System: A New Horizon of Manufacturing Automation," *Proceedings of Knowledge-Based Expert Systems for Manufacturing in the Winter Annual Meeting of ASME, Anaheim, California*, 11-23(1986).

[10] G. Harhalakis, C. Lin, H.Hillion and K. Moy, "Development of a Factory Level CIM Model," *Journal of Manufacturing Systems, vol. 9, no. 2*, 116-128(1990).

[11] J.L. Peterson, "Petri Net Theory and the Modeling of Systems," Prentice Hall, Englewood Cliffs, New Jersey, 1981.

[12] M. D. Jeng and F. DiCesare, "A Review of Synthesis Techniques for Petri Nets," *Proceedings of IEEE Computer Integrated Manufacturing Systems Conference, RPI*, May 1990.

[13] K. Jensen, "Colored Petri nets and the invariant method," *Theoretical Computer Science, vol. 14*, 317-336(1981).

[14] I. Suzuki and T. Murata, "A method for stepwise refinement and abstraction of Petri nets," *Journal of Computer System Science, vol. 27*, 51-76(1983).

[15] G. Harhalakis, C.P. Lin, L. Mark and P. Muro, "Formal Representation, Verification and Implementation of Rule Based INformation Systems for Integrated Manufacturing (INSIM)" *Technical Report TR 91-19, Systems Research Center, University of Maryland, College Park*, 1991.

[16] T.A. Nguyen, W.A. Perkins, T.J. affey and D. Pecora, "Knowledge Base Validation," *AI Magazine, summer*, 67-75(1987).

[17] B. Lopez, P. Meseguer and E. Plaza "Knowledge Based Systems Validation: A State of the Art," *AI Communications, Vol.3, No. 2*, 58-72(June 1990).

[18] K. Jensen, "Computer Tools for Construction, Modification and Analysis of Petri Nets," *Advances in Petri Nets, Part II*, 4-19(1986).

[19] K.H. Lee and J. Favrel, "Hierarchical Reduction Method for Analysis and Decomposition of Petri Nets," *IEEE Transactions on Systems, Man, and Cybernetics, vol. SMC-15, no. 2*, 1985.

[20] L. Mark and N. Roussopoulos, "Operational Specification of Update Dependencies," *Systems Research Center Technical Reprot No. SRC TR-87-37, University of Maryland*, 1987.