INT. J. COMPUTER INTEGRATED MANUFACTURING, 1991, VOL. 4, NO. 6, 351–363

INformation Systems for Integrated Manufacturing (INSIM): a design methodology

G. HARHALAKIS, C. P. LIN, L. MARK and P. R. MURO-MEDRANO

Abstract. Full control and management of information flow in manufacturing has not yet been achieved, mainly because of the data inconsistencies and lack of established functional relationships among different manufacturing application systems. Research toward CIM has been concentrating on the computerization of individual functions, such as computer-aided design and shop floor control, and the integration of data relations, such as global database frameworks and distributed database management systems. 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. This design methodology features an enhanced graphic modelling tool—Updated Petri Nets (UPN)—which is capable of modelling database updates and retrievals, under specific constraints and conditions, and uses a hierarchical modelling approach. Finally, a prototype rule-based system, using the INSIM methodology, is being implemented. It assimilates the functionality and ascertains the control of information flow between computer aided design, process planning, manufacturing resource planning, and shop floor control.

1. Introduction

1.1. *Background*

Due to the substantial improvements in computer technologies and increasing competition in manufacturing industry, more computerized manufacturing applications have been developed to automate various manufacturing functions in many existing factories, and to design new factories. Current research and computer software developed in the area of manufacturing automation has been quite intensive in dealing with product and process design, production planning, and job execution. However, the design of such systems was 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, has led to 'islands of automation' (Fig. 1) which suffer from data inconsistencies and lack of control of functional interactions between manufacturing application systems. Current and future trends for the use of computers in manufacturing include the control and the integration of information flow of a production operation into a computer-controlled factory management system. Various research projects in the area of computer-integrated manufacturing (CIM) have been conducted by NIST (Jones and McLean 1986, Davis and Jones 1988), ESPRIT (Bonnevie and Krzesinski 1987, Meyer 1987), CAM-i (Chryssolouris 1987), and AT&T (Franks *et al.* 1987). Many research projects emphasize individual aspects of CIM, such as RPI (Hsu *et al.* 1987) on developing a global database framework, TRW (Sepehri 1987) on synchronizing the interface between application systems and distributed databases, and the University of Illinois (Lu 1986) on developing a framework to perform common manufacturing tasks such as monitoring,

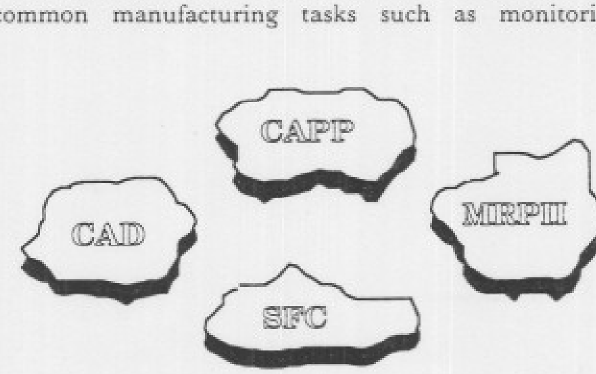

Figure 1. Islands of automation.

Authors: G. Harhalakis, C. P. Lin, and L. Mark, Systems Research Center, University of Maryland, College Park, MD 20742, USA; and P. R. Muro-Medrano, Department of Electrical Engineering and Computer Science, University of Zaragoza, C/María de Luna, 3, 50015 Zaragoza, Spain.

diagnostics, control, simulation, and scheduling. Their approach is to develop a generic CIM architecture, to create a global database framework, or to interface shop floor activities. However, our approach is to develop a control mechanism for managing the information flow among all the manufacturing application systems, and to fill the gap between the high level production management and the low level factory automation.

The main characteristics of the manufacturing environment dealt with here are described below:

Islands of automation: Different computer-based manufacturing application systems have been developed and used independently, without communicating with each other. The information flow between them is carried out through unreliable paper work, and it is controlled by various company departments without a unified set of procedural rules (Anderson *et al.* 1984). Thus, the need for the development of a company-wide policy for the management and control of automated information flow among existing application systems.

Emergence of new applications: New computer based manufacturing application systems are being introduced and these will also need to be integrated into the existing framework (Appleton 1984). Thus, the need for modularity.

Distributed systems: Most factories are already using distributed computer systems, with multiple databases, that serve specific applications in the factory. This has become the current trend in computer technology (Jablonski *et al.* 1988). Thus, the need for communications.

Integrity and security: Data integrity and security in individual databases have become major issues of concern in information integration (Date 1986). Thus, the need for controlled asscessibility.

1.2. *Objectives*

The primary objective of our research is to develop a methodology of acquiring domain knowledge (company policy), to verify and implement it as a knowledge-based system, in order to automate and control the information flow among all those computer-based manufacturing application systems. Our research 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 communication protocol for data transformation. Our control mechanism over existing distributed database management systems can achieve a fully-integrated manufacturing information system. Our view of the control in an information system includes:

Data ownership: Each application system has authority to create, update, and delete specific data entities, in reflection of specific company policies and procedures.

Precedence constraints: The pre-conditions for and sequences among activities within all the application systems involved are also defined according to specific company policies and procedures.

Our research, aiming at linking product and process design, manufacturing operations, and production management, focuses on the control of information flow between each of the key manufacturing applications 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 between manufacturing application systems is based on data commonalities and the dynamic control of functional relationships between them (see Fig. 2). 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 and processes, while the latter deal with the functioning of the system as it operates to satisfy the market demand. More specifically:

1. Static data:
   - product data: part master data, part revision records, and bills of material;
   - resource data: work centres and labour resources;
   - process data: routeings or process plans.
2. Dynamic data:
   - planning data: purchase orders and manufacturing orders.

The functional relationships that control the sequences

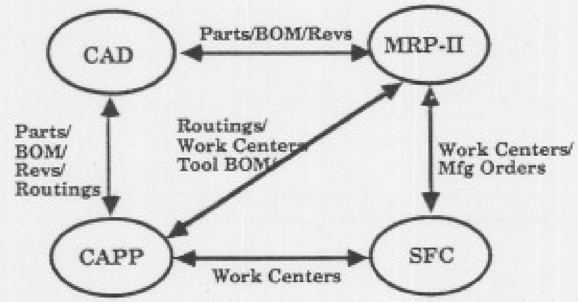

Figure 2. Data commonalities between CAD, CAPP, MRP II, and SFC.

of functions within and among those manufacturing application systems, and the database updates and retrievals, are purely domain-dependent, and can be very different from one company to another, depending on specific company policies. The methodology, however, for knowledge acquisition, modelling, validation, and implementation is totally generic.

The three major research issues to be addressed are:

Design of the CIM architecture: The CIM architecture is concerned with defining the functionalities of each manufacturing application system and the relationships between them.

Development of a methodology for the design of the KBS: The design and maintenance of a knowledge-based system (KBS) to control the functional relationships and information flow within the elements of the integrated system is the major task of this research. Based on a graphical modelling tool, Petri Nets, a hierarchical graphic representation of the KBS verifies its properties, and validates the company policy prior to the implementation phase.

Implementation: A procedure to automatically translate the KBS representation by Petri Net models into a rule specification language has been developed, and a prototype test system is in place.

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

2. Rule-based specifications of INSIM

2.1. *Overview*

The integrated system is intended for a discrete-parts, make-to-stock environment, where CAD, CAPP, MRP II, and SFC systems are best utilized. As described previously, the design of the model is based on the information flow established between the four functional areas. The common entities involved in this integrated model are part master data (including part revisions), bills of materials, work centres, process plans, and manufacturing orders, as shown in Fig. 2.

The model does not attempt to provide a 'bridge box', allowing users to hook up any existing commercial CAD, CAPP, and MRP II, and SFC system. The goal is to demonstrate the viability of achieving the integration and 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 package. It relies only on the basic functions available to most such commercial systems. In an actual implementation, the system would act as a controller between existing CAD, CAPP, MRP II, and SFC packages, while utilizing their respective capabilities. It must be understood that an appropriate model has to be developed, when attempting to control and integrate specific CAD, CAPP, MRP II, and SFC software, depending upon their specific features and characteristics, under a specific company policy. Therefore, the value of our work lies mostly in developing a methodology for designing a CIM information system, rather than the system itself.

2.2. *Role of each application system*

To maintain the generality of the model, only the most basic data carried by CAD, CAPP, MRP II, and SFC have been incorporated. If necessary, the model can be easily extended to reflect additional data and functions specific to particular commercial packages.

Specifying precisely the roles of the respective areas, i.e., CAD, CAPP, MRP II, and SFC, is of utmost importance when designing the model. CAD, being the centre of design activity, is the primary controller of product design information. The evaluation of design alternatives, the creation of new product parts, and the modification of existing parts is performed within CAD, often using inputs from other departments. Marketing and manufacturing are two major contributors to information regarding product designs. In addition, CAD initiates the bills of material for all product assemblies. An important problem commonly encountered is that, as a function, manufacturing succeeds design. Therefore any manufacturing problems occurring due to part design specification, are relayed to CAD only after designs have been finalized. It is therefore necessary for CAPP, the originator of process plans in the system, to work in concert with CAD, as the design of a part is ongoing. This approach, known as concurrent engineering, reduces the product development cycle and enhances competitiveness.

CAPP is solely responsible for developing manufacturing process plans. It organizes the manufacturing activities to be performed on a part into specific operations, each being assigned to a particular work centre, and each requiring tools, jigs, fixtures, and set-up and run times. CAPP, on the other hand, can initiate its own parts and bills of materials as they relate to necessary tools, jigs, and fixtures for production purposes. In addition, most CAPP systems maintain detailed work centre files, the information being used while preparing process plans.

MRP II plays a co-ordinating and monitoring role. It plans for and monitors the actual procurement of raw materials and manufacture of parts, respectively. It can also initiate non-product parts, such as tools and supply items, in the system. In addition, it records process plans as generated by CAPP, and also product structures of assembly parts, to provide them later to the SFC module. Work centre data are maintained here, with MRP II having sole discretion as to their initiation, maintenance, and deletion in the system.

While the definitions and functionalities of CAD, CAPP, MRP II systems are quite clear and widely accepted, the functions and inputs/outputs of a shop floor control system in a CIM environment are not yet well defined. Shop floor control is basically a system which directly controls the transformation of planned manufacturing orders into a set of jobs, for the transformation of raw materials into products. The basic activities of a shop floor control module can be summarized as follows:

- capacity planning and resource allocation based on inputs from MRP II;
- short-term capacity adjusting by using alternative routeings, planning overtime, and altering priorities;
- feedback for reporting machine performance and status, job completion stage, and actual labour and material usage.

Our definition of a generic shop floor control system and identification of the input/output requirements, have been influenced by a study of shop floor control systems from three major research projects: NIST (Jones and McLean 1986), ESPRIT (Bonnevie and Krzesinski 1987), MADEMA (Chryssolouris 1987). The Productivity Improvement Systems for Manufacturing (PRISM) (Franks *et al.* 1987) developed in AT&T Bell Laboratories has been our primary guideline.

SFC is designed to communicate with an MRP II system and to perform job scheduling and monitoring using detail routeing information from CAPP. Once a market demand arrives, MRP II will generate planned purchasing and manufacturing orders. In doing this, lead times from part master records and existing inventories are taken into account. SFC will then schedule the manufacturing jobs, based on the current load and the detail routeing information from CAPP. It will then dispatch these job orders down to the shop floor. During production, SFC will constantly monitor the job status, work centre status, actual material, and labour consumption, and report them back to the MRP II for costing and updating purposes. Finally SFC is supposed to react to and provide real time solutions in the event of disturbances, such as machine breakdowns, critical labour absenteeism, and material shortages. Issues that cannot be resolved at the SFC level must be communicated to MRP II for further action (Nagi 1988).

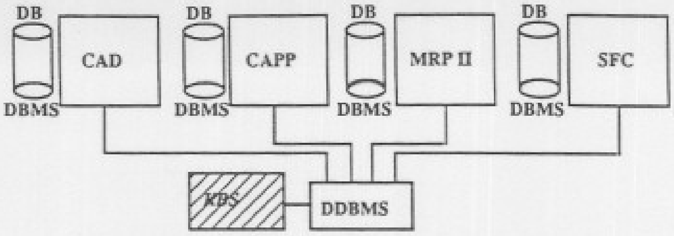

Figure 3. Overall CIM information flow architecture.

2.3. *Overall CIM information flow architecture*

Our CIM architecture concentrates on the integration of manufacturing applications at the factory level, referring to the facility level of the NIST hierarchical architecture, as depicted in Fig. 3. CAD, CAPP, MRP II, and SFC can be integrated together through a general distributed database management system (DDBMS). The KBS drives the DDBMS to control the information flow, following procedural rules constraints and other procedures 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.

3. Knowledge base design methodology

As mentioned above, the design and maintenance of a KBS to control the functional relationships and information flow within the integrated system is a major task of this research. Our design methodology for it is depicted in Fig. 4. It starts from user-defined rule specifications, reflecting a specific company policy, which is then modelled using a special set of Coloured Petri Nets—UPN (Updated Petri Nets) and a hierarchical modelling methodology, discussed respectively in sections 3.2.1 and 3.2.2. The next step is to convert UPN model into General Petri Nets (GPN) for validation purposes, and feed the results back to the user to resolve (i)

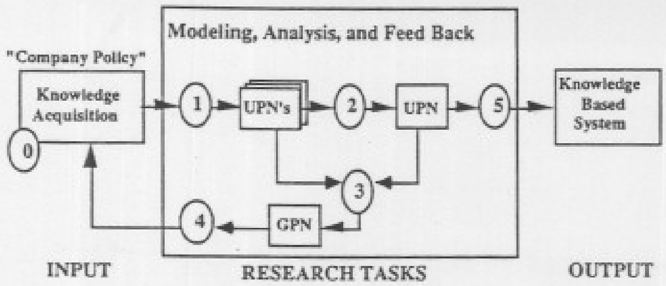

Figure 4. Knowledge base design methodology.

conflicting company rules; (ii) errors introduced during the modelling phase. After the model has been validated, a parser translates the UPN model into a rule specification language. The end result is a software package that controls the data flow and accessibility between several data bases. In short, the input is a set of company rules and the output is an AI production system for controlling operations, accessibility and updates of data within the manufacturing applications involved. A review of the major design phases of the system is presented below.

3.1. *Knowledge acquisition (company policy)*

The design of the model is based on the information flow established between all the manufacturing applications, namely CAD/CAPP/MRP II/SFC. The expert rules embedded in the knowledge base are 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 our 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.

3.1.1. *Scenarios of the proposed CIM system*: The development of a KBS usually starts from designing a set of abstract rules for all specific entities within the system. A set of scenarios under each entity which represent these abstract rules is listed below.

Part data:

- adding new product parts in CAD;
- adding new product part revisions in CAD;
- adding new non-product parts in CAPP;
- adding new non-product part revisions in CAPP;
- adding new non-product parts in MRP II;
- adding new non-product revisions in MRP II;
- making parts obsolete;
- deleting parts.

Product structures (bills of material):

- adding component relationships in CAD (for products);
- adding component relationships in CAPP (for tools, jigs, fixtures, etc.);
- deleting component relationships;
- substituting components in relationships;
- changing the required quantity of a component;
- copying relationships from one assembly to another.

Work centres:

- establishing new work centres in MRP II;
- modifying work centre in MRP II, CAPP, and SFC;
- deleting work centres in MRP II.

Process plans:

- establishing new process plans in CAPP;
- modifying process plans in CAPP;
- deleting process plans in CAPP.

Manufacturing orders:

- adding orders in MRP II;
- modifying orders in MRP II;
- modifying order BOM/routeing in SFC;
- updating job status in SFC (including actual material issued and time taken);
- deleting orders from MRP II.

3.1.2. *Status codes*: The flow of information within the system is controlled using a set of status codes assigned to each set of entity data within each functional area. The status codes are designed to provide for triggering the right action, while controlling the sequence of various part and process design and manufacturing-related activities. The following are the status codes used in the system.

Working: The 'working' status is given to CAD part data related to designs that have not yet been finalized. In a similar fashion it is given to process plans in CAPP, work centres in CAPP, and shop orders in SFC. In the case of work centres it is intended to signify that some work centre data is till missing in CAPP.

Released: The 'released' status is indicative of an entity becoming active in the system. It is applied to CAD and MRP II part revision data; CAPP, SFC, and MRP II work centre data; CAPP and MRP II process plan data; SFC and MRP II order data.

Hold: The 'hold' status is given normally to an entity, when it is being reviewed for possible revision or replacement, and the entity should not be used while on hold. For example, it can be given to a work centre in the case of an extended breakdown. It is used for CAD and MRP II part revision data; CAPP, SFC, and MRP II work centre data; CAPP and MRP II process plan data; SFC and MRP II order data.

Obsolete: Data related to entities that are no longer considered active, are given the 'obsolete' status. This code is used by CAD part revisions and CAPP routeings. However, MRP II and SFC handle obsolescence with the use of effectivity start and effectivity end dates. Therefore they do not require this status code.

3.2. *Modelling of the knowledge base*

Although General Petri Nets initially adopted in this research can in principle handle the modelling of the knowledge base, it has become necessary to define more complex semantics in order to handle the increasing complexity of the knowledge base, due to the involvement of more applications and their entities. Hence we have started developing Updated Petri Nets (UPN), and a hierarchical modelling methodology with a systematic approach for the synthesis of separate nets with common places. Some of this work was inspired by Jeng and DiCesare (1990).

3.2.1. *Evolution of Updated Petri Nets*: Petri Nets were originally developed by Carl Adam Petri in his doctoral thesis, 1962, at the University of Darmstadt, (then) West Germany. There have been many reports and papers published on Petri Nets with a wide variety of applications due to their modelling power. Petri Nets can be applied to most systems in representing graphically not only sequential but also concurrent activities. Because of their mathematical representation, they can be formulated into state equations, algebraic equations, and other mathematical models. Therefore, Petri Nets can be analysed mathematically for the verification of system models and are ideal for modelling dynamically and formally analysing complex dynamic relationships of interacting systems. Readers may refer to Peterson (1981) and Reisig (1985) for the fundamentals of Petri Nets theory. A survey of literature where various types of Petri Nets are used in modelling various systems in general and manufacturing systems in particular, has been conducted. Worth mentioning are artificial intelligence in network systems (Courvoisier *et al.* 1983), flexible manufacturing systems (Crockett and Desrochers 1986), scheduling and sequencing (Ravichandran and Chakravarty 1987, Merabet 1987, Dridi *et al.* 1985), and information integration for manufacturing applications (Harhalakis *et al.* 1988, 1990, 1991).

To facilitate the modelling of complex systems, a special type of the high level Petri Nets called Coloured Petri Nets (CPN) have evolved. A Coloured Petri Net (Jensen 1981) is a generalization of a Petri Net in which information is aggregated in tokens, places, and arcs. Tokens are distinguishable and are assigned different 'colours', and arcs are labelled with associated functions. In addition, the use of CPN 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 CPN in modelling not only the rule base, but also the database changes which ensure consistency in representing the database status in the CIM system. An enhanced version of CPN, named Updated Petri Nets (UPN), has been developed and used in modelling the CAD/CAPP/MRP II/SFC integrated system (Harhalakis *et al.* 1991).

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:

$$\langle P, T, MT, C, Ic^-, Ic^+, In^-, In^+, M \rangle$$

where

- $P = \{p_1, \ldots, p_n\}$ denotes the set of places (represented graphically as circles);
- $T = \{t_1, ., t_m\}$ denotes the set of primitive transitions (represented graphically as black bars);
- $MT = \{mt_1, \ldots, mt_l\}$ denotes the set of compound transitions (represented graphically as blank bars);
- $C = \{C(p_1), \ldots, C(p_n)\}$ denotes the colouring, i.e., 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 jth component m_j denotes the colour 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*, and *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 operation) and will be detailed below. Sample scenarios modelled by using UPN will be demonstrated in section 3.3.

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. Horizontal relations are established by means of what we call 'hmrules'. A hmrule, hm_a, specifies a relation in a set of transitions $hm_a^t = \{t_1, t_2, \ldots, t_m\}$, where $m \geqslant 1$ and t is defined at the level of abstraction a, $\forall t \in hm_a^t$. The sub-net, defined by the metarule hm_a is composed by the set of transitions hm_a^t and the places that are connected to the surrounding arcs of transitions in hm_a^t. Hmrules are generally used to describe scenarios or sub-nets at a given level of abstraction.
- **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). The vertical relationship is based on the identification of the input and output transitions which are mainly the links between the sub-net with the input and output place sets of its parent transition of the super-net. Transitions, representing sub-nets, are named as *compound* transitions to be distinguished from the *primitive* transitions representing single operation. A vmrule, vm_a, specifies relations between the input/output place sets $(I^-(t)/I^+(t))$ of the compound transition t' (at level $a-1$) and its lower level (level a) transitions $\{t_1, t_2, \ldots, t_m\}$, where $m \geqslant 1$ and t_i is defined at the level of abstraction a. Vertical decomposition in UPN is performed in such a way that the behaviour at the higher level of abstraction is also preserved when working at lower levels of abstraction.

3.2.2. *Modelling 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. Following a similar concept, a hierarchical modelling method using UPN has been developed 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.

Some work in hierarchical representation using Petri Nets has been done for various applications (Valette 1979, Suzuki and Murata 1983, Narahari and Viswanadham 1985, Jeng and DiCesare 1990). The hierarchical modelling methodology facilitates the modelling task. It incorporates:

Top-down stepwise refinement technique for the modelling of each scenario from an abstract and aggregate level to a detailed level. This approach necessitates the development of new Petri Net modelling entities which include two types of transitions;

one to represent primitive operations, and the other to represent compound operations which can be further exploded into sub-nets as mentioned above. 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 until no compound transition exists.

Synthesis technique for synthesizing separate nets, which represent different scenarios of the system, to form a coherent net. A breakthrough in our modelling approach is to incorporate the modelling of the databases of the manufacturing application systems involved, using UPN, by defining the database states as global variables, and synthesizing nets through them systematically. This enables the structuring of Petri Nets in a progressive manner, and facilitates the transformation of the 'company policy' of Fig. 4 to a formal model.

3.3. *Sample scenarios from the CAD/CAPP/MRP II/SFC integrated system*

3.3.1. *System specification of a sample scenario*: A set of scenarios of the CAD/CAPP/MRP II/SFC integrated system, which form the knowledge base to manage and control information flow among manufacturing applications, has been developed as described before. In this section, two scenarios, 'Establishing and deleting new work centres' in MRP II, are detailed initially in plain English like most company policies. Both the sequence of activities within the various manufacturing applications and the precedence constraints of operations prescribed by the company policy are detailed below.

I. Sample scenario A: Establishing new work centres in the system via MRP II:

To establish a new work centre in the system, MRP II users must provide the following basic information, in addition to assigning its unique identification number:

- ID number;
- description;
- department.

Subject to the condition that this ID number does not already exist in the system, this work centre record is established in MRP II with its status set to hold. MRP II users then finalize all the other work centre details needed in MRP II module as follows:

- capacity;
- resource code;
- rate code;
- dispatch horizon;
- effectivity start date.

Then the MRP II user releases the work centre.

These data may be entered separately into the system as each data item becomes known. Otherwise, the system will request for them during releasing of the work centre in MRP II as described below. In addition to the data fields mentioned, effectivity end date, status code, work centre state, and work centre load profile are also part of the work centre record in MRP II. The status code is not a user input, but is updated automatically from hold to released by the release transactions on the work centre. The work centre state is maintained by SFC users and is not provided at this stage. The work centre load profile will be entered and maintained by SFC and updated automatically in MRP II after the work centre is allocated for job scheduling.

The release of a work centre in MRP II triggers a set of consistency checks, which are as follows: the ID number provided must exist in MRP II with hold status; all the required data fields should have been filled, and any data fields left out by users are requested at this stage. If all these checks are satisfied, the system changes the work centre status code from 'hold' to 'released', and a skeletal work centre record is created automatically in the work centre file in CAPP, with its status set to 'working' as well as a work centre record in the work centre file in SFC with its status set to 'hold'. These work centre records contain all the common information between MRP II and CAPP, and between MRP II and SFC.

CAPP users then input the detailed technical information regarding that work centre. The following fields are required to be completed in CAPP, before the work centre can be given a released status, and made effective and ready to be used in process plans. If a particular data field does not apply to the specific work centre, then 'inapplicable' will be entered automatically. However no field can be left blank.

- horse power;
- speed range;
- feed range;
- work envelope;
- accuracy;
- tool change time;
- feed change time;
- speed change time;
- table rotation time;
- tool adjusting time;
- rapid traverse time.

Similar to the MRP II, invoking the work centre release transaction in CAPP triggers a set of consistency checks, which are as follows: the ID number is checked to ensure that a work centre file with status set to 'working' exists in CAPP; all the required data fields should have been filled, and any data fields left out by users are requested at this stage with the help of system generated prompts. Upon the satisfaction of these checks, the work centre obtains a released status in CAPP and the common data fields between CAPP and SFC are copied automatically from CAPP to SFC.

This scenario ends by releasing the work centre in the SFC module. Invoking the work centre release transaction in SFC triggers a consistency check: the ID number is checked to ensure that a work centre file with status set to 'hold' exists in the SFC; the WC status in both MRP II and CAPP are set to 'r' to ensure all the necessary information has been provided; the current data is past the effectivity start date. The work centre state is then automatically set to 'av' in SFC and MRP

II for being available. Upon the satisfaction of these checks, the work centre obtains a released status in SFC and work centre state is updated in MRP II.

II. Sample scenario B: Deleting work centres from the system via MRP II:

Work centres can only be deleted via MRP II. As explained earlier, MRP II is the execution function in most companies, being in charge of maintaining static data regarding parts and work centres in the system. It is the sole centre for purchasing of resources, and in turn, is the function through which equipment is phased out, or deleted from the system.

When the delete operation is invoked in MRP II, the following system checks must be initiated. A check must be made to see that the work centre being deleted exists in MRP II. The status of the work centre is not relevant to the operation. In addition all the routeings maintained by the MRP II routeings module are checked. If any routeings utilizing this work centre exist, and are in the 'hold' or 'released' status in CAPP, the operation fails, and a message to this effect is displayed. This is because work centres which are utilized by active routeings, cannot be deleted. In addition, if any manufacturing order in MRP II and SFC utilizing this work centre exists, the operation fails, and a message to this effect is again displayed. This is because work centres which are utilized by active orders cannot be deleted. If the above checks are satisfied, the work centre can be deleted from the routeings module of MRP II, CAPP, and SFC.

3.3.2. *UPN model of sample scenarios*: Once the company policy is clearly expressed, a model representing it can then be developed to simulate its logic as described in this section, and furthermore to detect any inconsistencies and incompleteness in it which will be described in section 3.4. The UPN model is presented in two forms, graphically (UPN graph) and mathematically (UPN incidence matrices).

I. **Top down refinement**—*establishing new work centres in the system via MRP II*

This scenario is one metarule itself, which in turn involves four metarules (as defined in section 3.2.1) 'insert a work centre in MRP II', 'release a work centre in MRP II', 'release a work centre in CAPP', 'release a work centre in SFC'—connected to each other based on the horizontal composition formalism. Each of these metarules is then refined to one sub-net and connected to the higher level net, based on the vertical composition formalism.

The above example modelled by a UPN graph is shown in Fig. 5. An abstract company policy, which represents the scenario 'create a work centre via MRP II', is modelled with compound transitions (blank bars), representing (t1) 'insert a work centre in MRP II', (t2) 'release a work centre in MRP II, (t3) 'release a work centre in CAPP', (t4) 'release a work centre in SFC', and the sequence and constraints among these compound transitions.

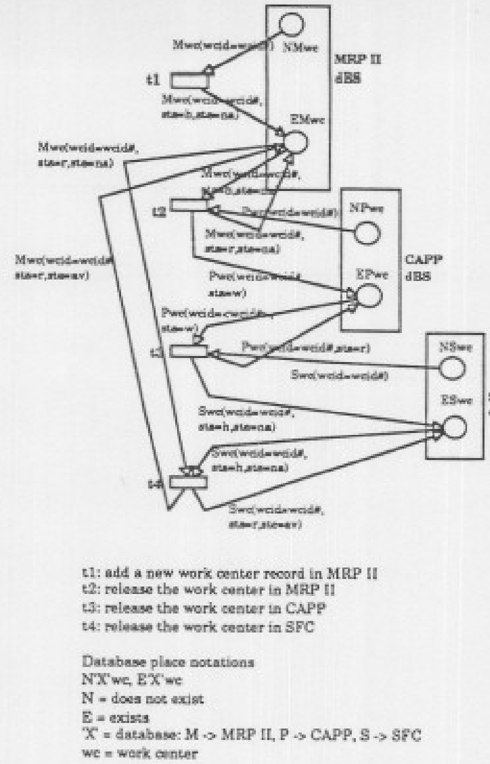

Figure 5. UPN graph of the scenario: 'Create a work centre via MRP II' at an abstract level.

One of these compound transitions (metarules), 'insert a work centre in MRP II' of Fig. 6 is refined into a lower level sub-net, through the vertical decomposition. This subnet contains four primitive transitions representing (t1.1) 'request Wcid', (t1.2) 'output error message', (t1.3) 'request Des and Dep', (t1.4) 'add work centre record into the MRP II DB', which cannot be further refined. It also defines the horizontal composition among them and the interaction with the MRP II database.

An incidence matrix ($C = C^+ - C^-$) is an algebraic representation of one Petri Net with columns representing transitions, rows representing places, and the matrix elements representing the weights of arcs from transitions to places (C^+) and from places to transitions (C^-). It enables us to model and analyse the dynamic behaviour of Petri Nets mathematically. The matrix elements in GPN are integer values representing simple connection between places and transitions. However, in UPN the elements in the incidence matrix become functions.

A state equation to transform one marking of Petri Nets into the next one is shown below.

$$M_t = M_{t-1} + Cx$$

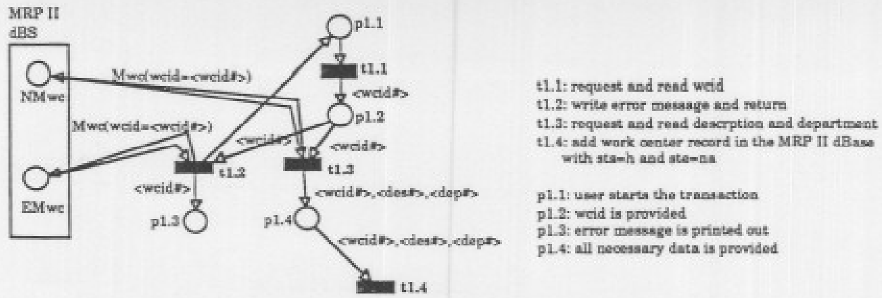

Figure 6. Sub net of the work centre creation scenario: 'Insert a work centre via MRP II'.

where M_t and M_{t-1} are the successive markings t and $t-1$, and x is a vector containing the fired transitions.

The following matrices represent the UPN model in Fig. 5.

C^- =

	t_1	t_2	t_3	t_4
NMwc	⟨*wcid* #⟩	0	0	0
EMwc	0	(⟨*wcid* #⟩, *h*, *na*)	0	(⟨*wcid* #⟩, *r*, *na*)
NPwc	0	⟨*wcid* #⟩	0	0
EPwc	0		(⟨*wcid* #⟩, *w*)	0
NSwc	0	0	⟨*wcid* #⟩	0
ESwc	0	0		(⟨*wcid* #⟩, *h*, *na*)

C^+ =

	t_1	t_2	t_3	t_4
NMwc	0	0	0	0
EMwc	(⟨*wcid* #⟩, *h*, *na*)	(⟨*wcid* #⟩, *r*, *na*)	0	(⟨*wcid* #⟩, *r*, *na*)
NPwc	0	0	0	0
EPwc	0	(⟨*wcid* #⟩, *w*)	(⟨*wcid* #⟩, *r*)	0
NSwc	0	0	0	0
ESwc	0	0	(⟨*wcid* #⟩, *h*, *na*)	(⟨*wcid* #⟩, *r*, *av*)

The following matrics describe the UPN model in Fig. 6.

C^- =

	$t_{1.1}$	$t_{1.2}$	$t_{1.3}$	$t_{1.4}$
NMwc	0	0	⟨*wcid* #⟩	0
EMwc	0	(⟨*wcid* #⟩)	0	0
$p_{1.1}$	1	0	0	0
$p_{1.2}$	0	⟨*wcid* #⟩	⟨*wcid* #⟩	0
$p_{1.3}$	0	0	0	0
$p_{1.4}$	0	0	0	⟨*wcid* #⟩, ⟨*des* #⟩, ⟨*dep* #⟩

C^- =

	$t_{1.1}$	$t_{1.2}$	$t_{1.3}$	$t_{1.4}$
NMwc	0	0	⟨*wcid* #⟩	0
EMwc	0	(⟨*wcid* #⟩)	0	0
$p_{1.1}$	0	1	0	0
$p_{1.2}$	⟨*wcid* #⟩	0	0	0
$p_{1.3}$	0	⟨*wcid* #⟩	0	0
$p_{1.4}$	0	0	⟨*wcid* #⟩, ⟨*des* #⟩, ⟨*dep* #⟩	0

The connections between higher and lower levels are made through the identified **input** and **output** transitions (Harhalakis *et al.* 1991). 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 (Fig. 6) of Fig. 5 is shown in Fig. 7.

II. **Net synthesis**—*establishing and deleting work centres from the system via MRP II*

The two scenarios, 'Creation and deletion of work centres via MRP II modelled in UPN', are shown in Figs 5 and 8 respectively. Each of the figures has its incidence matrices ($C = C^+ - C^-$) associated with it.

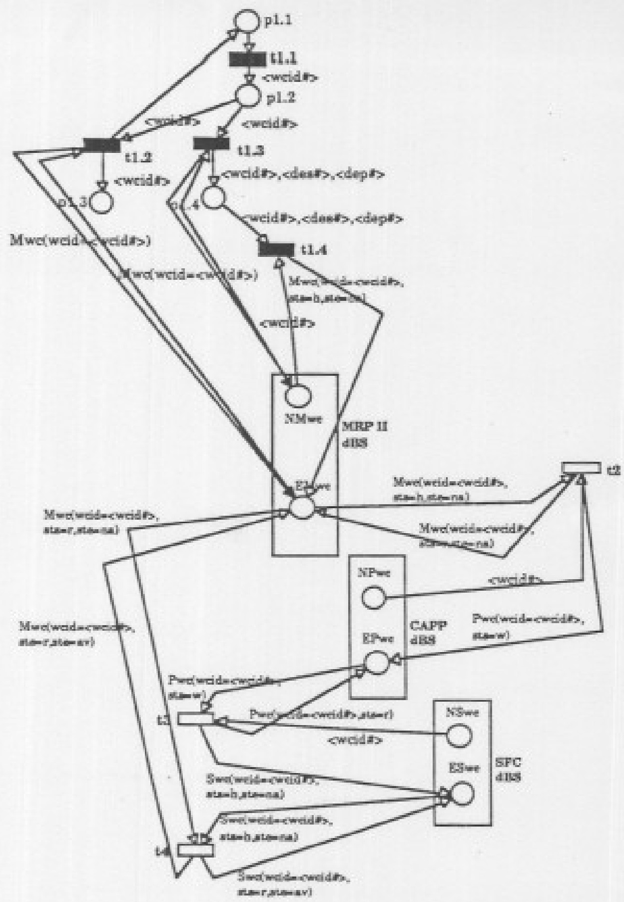

Figure 7. Partially-refined UPN of the scenario: 'Create a work centre via MRP II'.

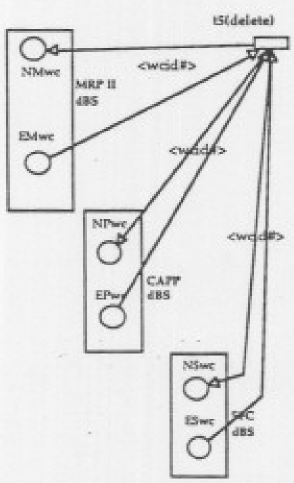

Figure 8. UPN graph of the scenario: 'Delete a work centre via MRP II' at an abstract level.

The following incidence matrices describes the UPN model of Fig. 8.

$$
C^- = \begin{array}{lc} & t_1 \\ NMwc & 0 \\ EMwc & \langle wcid\#\rangle \\ NPwc & 0 \\ EPwc & \langle wcid\#\rangle \\ NSwc & 0 \\ ESwc & \langle wcid\#\rangle \end{array}
\qquad
C^+ = \begin{array}{lc} & t_1 \\ NWwc & \langle wcid\#\rangle \\ EMwc & 0 \\ NPwc & 0 \\ EPwc & \langle wcid\#\rangle \\ NSwc & \langle wcid\#\rangle \\ ESwc & 0 \end{array}
$$

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 Figs 5 and 8 is shown in Fig. 9.

In the above example, the company policy, which involves the scenario 'Create a work centre via MRP II' and 'Delete a work centre from MRP II', is now represented by a single net. This net retains the dynamic behaviour 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 model of the entire system.

3.4. *Knowledge base verification*

The major objective of creating a KBS using Petri Nets is the ability of validating the KBS mathematically and systematically. The completeness (dead-end rules, unfirable rules), consistency (redundant rules, subsumed rules, under-constrained rules), and conflicts, are the

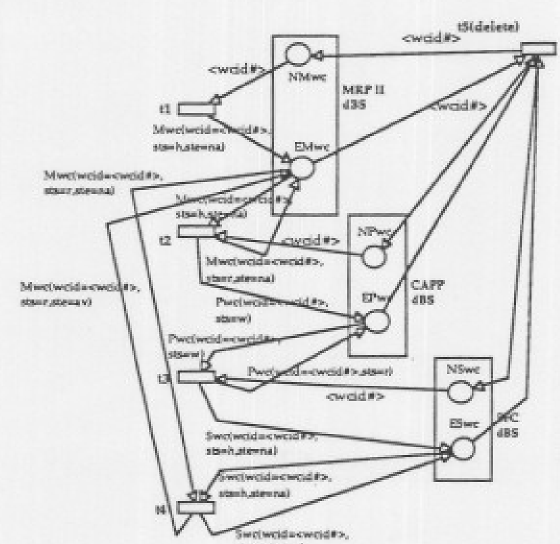

Figure 9. UPN graph of the synthesized scenario 'Creation and deletion of a work centre via MRP II' at abstract level.

major issues in knowledge-based validation (Nguyen *et al.* 1987, Lopez *et al.* 1990). 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, behavioural nets, and net invariants, are also used (Peterson 1981, Jensen 1986, Martinez and Silva 1982). 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 modelling process. The reachability tree can be used to detect any deadlocks or inconsistencies in the model. The behavioural net can be used to detect redundancies in the net and is a useful tool for reducing the complexity of the model. The programs for computerizing these analysis methods have been developed and applied extensively. Some reduction rules (Lee and Favrel 1985) have also been investigated for reducing the complexity of nets prior to the analysis phase (Harhalakis *et al.* 1991).

However, these analysis techniques were initially developed for Generalized Petri Nets (GPN), and do not apply to Coloured 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 a high complexity in the development and execution of these algorithms. An alternative approach will be to *unfold* UPN into GPN before they can be analysed (Harhalakis *et al.* 1991).

3.5. *Implementation*

The implementation phase is being realized through the development of the Update Dependencies Language (UDL) in the Department of Computer Science, at the University of Maryland (Mark and Roussopoulos 1987). 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 as a special rule specification language for the implementation of our KBS. The algorithm for automatic translation between the UPN and the UDL language has been developed (Harhalakis *et al.* 1991), which reduces substantially the implementation effort.

4. Conclusions

The INformation Systems for Integrated Manufacturing (INSIM) methodology has been developed and implemented for generating a knowledge-based system to manage and control the information flow among CAD/CAPP/MRP II/SFC application systems effectively. This design methodology is fairly generic in that it can be applied to generate knowledge-based systems for other applications. Depending on the rule specification language used, this design methodology can be applied in the same way with a modified Petri Nets translator. 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 synthesis techniques for modelling more complicated scenarios of the company policy, and the development of new techniques for validating rule bases with more complex structures.

Acknowledgement

The authors acknowledge the Engineering Research Centre program NSFD CPR88003012 of the University of Maryland for funding this research.

References

ANDERSON, D. C., SOLBERG, J. J., and PAUL, R. P., 1984, Factories of the future: how will automation research be integrated? *Computers in Mechanical Engineering*, 2, 31–36.

APPLETON, D. S., 1984, The state of CIM. *Datamation*, 30, 66–72.

BONNEVIE, A., and KRZESINSKI, P., 1987, A double approach for analysis and design of production systems. *ESPRIT'86: Results and Achievements*, 469–478.

CHRYSSOLOURIS, G., 1987, MADEMA: an approach to intelligent manufacturing systems. *CIM Review*, Spring, 11–17.

COURVOISIER, M., *et al.*, 1983, A programmable logic controller based on a high level specification protocol. *Proceedings of IECON Conference on Industrial Electronics*, 174–179.

CROCKETT, D. H., and DESROCHERS, A. A., 1986, Manufacturing workstation control using petri-nets. Technical Report 83, Robotics and Automation Laboratory, Department of Electrical, Computer, and System Engineering, Rensselaer Polytechnic Institute.

DATE, C. J., 1986, *An Introduction to Database System* (Addison-Wesley, Reading, MA).

DAVIS, W. J., and JONES, A. T., 1988, A real-time production scheduler for a stochastic manufacturing environment. *International Journal of Computer Integrated Manufacturing*, 1, 101–112.

DRIDI, N., LEOPOULOS, V. I., and PROTH, J. M., 1985, Properties of FMS regarding optimal control. *Advances in Production Management Systems 85*, 325–327.

FRANKS, R. L., HOLTMAN, J. P., HSU, L. C., RAYMER, L. G., and SNYDER, B. E., 1987, Productivity improvement systems for manufacturing. *AT&T Technical Report*, **66**, 61–76.

HARHALAKIS, G., MARK, L., and LIN, C. P., 1989, A knowledge-based prototype of a factory-level CIM system. *Journal of Computer Integrated Manufacturing Systems*, **2**, 11–20.

HARHALAKIS, G., LIN, C. P., HILLION, H., and MOY, K., 1990, Development of a factory level CIM model. *Journal of Manufacturing Systems*, **9**, 116–128.

HARHALAKIS, G., LIN, C. P., MARK, L., and MURO, P., 1991, 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.

HSU, C., ANGULO, C., PERRY, A., and RATTNER, L., 1987, A design method for manufacturing information management. *Proceedings of Conference on Data and Knowledge Systems for Manufacturing and Engineering*, Hartford, CT, 93–102.

JABLONSKI, S., RUF, T., and WEDEKIND, H., 1988, Implementation of a distributed data management system for manufacturing applications. *Proceedings of International Conference on Computer Integrated Manufacturing*, Troy, 19–25.

JENG, M. D., and DICESARE, F., 1990, A review of synthesis techniques for Petri Nets. *Proceedings of IEEE Computer Integrated Manufacturing Systems Conference*, RPI.

JENSEN, K., 1981, Coloured Petri Nets and the invariant method. *Theoretical Computer Science*, **14**, 317–336.

JENSEN, K., 1986, Computer tools for construction, modification, and analysis of Petri Nets. *Advances in Petri Nets, Part II*, 4–19.

JONES, A. T., and MCLEAN, C., 1986, A proposed hierarchical control model for automated manufacturing systems. *Journal of Manufacturing Systems*, **5**, 15–25.

LEE, K. H., and FAVREL, J., 1985, Hierarchical reduction method for analysis and decomposition of Petri Nets. *IEEE Transactions on Systems, Man, and Cybernetics*, **SMC-15**, 272–280.

LOPEZ, B., MESEGUER, P., and PLAZA, E., 1990, Knowledge-based systems validation: a state of the art. *AI Communications*, **3**, 58–72.

LU, S. C. Y., 1986, Knowledge-based expert systems: a new horizon of manufacturing automation. *Proceedings of Knowledge-Based Expert Systems for Manufacturing in the Winter Annual Meeting of ASME*, Anaheim, CA, 11–23.

MARK, L., and ROUSSOPOULOS, N., 1987, Operational specification of update dependencies. Systems Research Center Technical Report No. SRC TR 87-37, University of Maryland.

MARTINEZ, J., and SILVA, M., 1982, A simple and fast algorithm to obtain all invariants of a generalized Petri Net. *Second European Workshop on Application and Theory of Petri Nets*, 301–310.

MERABET, A. A., 1987, Synchronization of operations in a flexible manufacturing cell: the Petri Net approach. *Journal of Manufacturing Systems*, **5**, 161–169.

MEYER, W., 1987, Knowledge-based realtime supervision in CIM: the work-cell controller. *ESPRIT'86: Results and Achievements*, 33–52.

NAGI, R., 1988, Selection and layout of facilities for cellular manufacturing. M.S. Thesis, University of Maryland.

NARAHARI, Y., and VISWANADHAM, N., 1985, A Petri Net approach to the modelling and analysis of flexible manufacturing systems. *Annals of Operations Research*, **3**, 381–391.

NGUYEN, T. A., PERKINS, W. A., LAFFEY, T. J., and PECORA, D., 1987, Knowledge base validation. *AI Magazine*, Summer, 67–75.

PETERSON, J. L., 1981, *Petri Net Theory and the Modeling of Systems* (Prentice Hall, Englewood Cliffs, NJ).

RAVICHANDRAN, R., and CHAKRAVARTY, A. K., 1987, Decision support in flexible manufacturing systems using timed Petri Nets. *Journal of Manufacturing Systems*, **5**, 89–100.

REISIG, W., 1985, *Petri Net* (Springer-Verlag, Berlin).

SEPEHRI, M., 1987, Integrated data base for computer-integrated manufacturing. *IEEE Circuits and Devices Magazine*, March, 48–54.

SUZUKI, I., and MURATA, T., 1983, A method for stepwise refinement and abstraction of Petri Nets. *Journal of Computer System Science*, **27**, 51–76.

VALETTE, R., 1979, Analysis of Petri Nets by stepwise refinements. *Journal of Computer and System Sciences*, **18**, 35–46.