Artificial Intelligence in the Factory of the Future

Mark S. Fox

Intelligent Systems Laboratory
The Robotics Institute
Carnegie-Mellon University
Pittsburgh, Pennsylvania 15213

Abstract This paper describes the Intelligent Management System (IMS) project, which is part of the Factory of the Future project in the Robotics Institute of Carnegie-Mellon University. IMS is a long term project concerned with applying artificial intelligence techniques in aiding professionals and managers in their day to day tasks. This report discusses both the long term goals of IMS, and current research.

1 Introduction

Classical research in the area of factory automation has been concerned more with production processes than with management. Yet, it has been observed that in many small batch-size factories, white collar labor accounts for a large fraction of total labor cost, and in some cases exceeds 50%. And that small batch-size production accounts for 50%-75% of the dollar value of durable goods produced in the United States. In metal-cutting, job-shop production environments, it has been found that only 20% of the time an order is in a factory, is it actually mounted on a machine. And during only 5%-10% of its time on the machine, are value-adding operations being performed (Am. Mach., 1980). It appears that decisions made manually in the shop today are less than satisfactory as testified to by high in process time of orders, low machine utilization, and high overheads. The introduction of robotic and other flexible technologies into manufacturing both increases the number of ways a product may be produced, and decreases production times. Increasing the number of alternatives increases the complexity of deciding what to do next, while shorter set up times reduce the time in which a decision is to be made. Hence, persisting in the use of such manual planning and control methods limits our ability to utilize the flexibility afforded by robotic technology.

It is obvious that our current manual and automated techniques are not sufficient. New theories and systems are required whose functionality will aid managerial, professional and production personnel in their day to day decision making. These systems must be more effective in the tasks they perform. They must integrate and communicate the knowledge and skill of the whole organization, making them available for management decisions. More importantly, they must aid in the performance of tasks. These systems must become more *intelligent*. In the summer of 1980 we began the design and construction of what we call an *Intelligent Management System* (IMS) (Fox, 1981). Research in IMS is concerned with the extension and application of artificial intelligence (AI) techniques to decision making at levels within a manufacturing organization.

The methodological focus of IMS is not the creation of an information system to support decision making as typically found in decision support research (Alter, 1979), but to explore more ill-structured problems (Simon, 1960) by means of heuristic problem-solving techniques. Ten years ago the promise of artificial intelligence techniques was unclear, but advances in the last decade have resulted in systems that display surprising capabilities and in some cases perform better than experts in the field. Examples can be found in Computer Configuration: R1 (McDermott, 1980); Machine Diagnosis: PDS (Fox et al., 1983); Medical Diagnosis: Mycin (Shortliffe, 1976), Internist (Pople, 1977); Geological Analysis: Prospector (Duda et al., 1978); Speech Understanding: Hearsay-II (Erman et al., 1980), Harpy (Lowerre, 1977). It is our goal to explore the application of artificial intelligence to managerial and professional problems.

This paper describes both the goals of the IMS research, and the research performed to date in organization modeling, operations control, management, and analysis. The next two sections of the paper describe both the functional and architectural goals of IMS. This is followed by an overview of research being performed in the Intelligent Systems Laboratory towards an IMS.

2 Functional Goals

The broad functional goals of the Intelligent Managment System Project include:

1. Providing expert assistance in the accomplishment of professional, managerial, and production tasks, and

2. Integrating and coordinating the management of the organization.

These goals were refined by analysing professional and managerial needs in a variety of industries including: power generation, defense, and computers. The following is a sampling of questions we found to be asked but often went unanswered. Some of the questions were concerned with the status of activities. But the majority were in the area of "decision support", where the decisions spanned the areas of operations control to advance planning. The following is just a few of their replies:

- What is the effect of changes in engineering specifications: designs, materials, process specifications, etc?

- What is the proper inventory level at various levels, i.e., raw, work in progress, finished parts, etc?

- What if the jig grinder goes down with maintenance problems?

- Based on downtime, cost of maintenance and cost of replacement parts, how do I determine when to buy a new piece of production equipment?

- Charlotte moves one (1) BB72 rotor two (2) months ahead of schedule and Lester moves two (2) BB22 rotors back in schedule.

  1. How are all promised dates for next six months affected?

  2. What manpower changes are needed by cost center to accomplish changes?

  3. What material delivery changes need to be made?

- How do I select the process to be used? What if the shop changes the plan and needs to perform a particular job on a different machine from the standard?

- Not enough information concerning the current state of production on the floor, orders in process, problems etc. is communicated quickly and succinctly to interested parties within the plant.

- Changes to products or tools by the engineering division are not adequately communicated and coordinated with changes taking place in the factory.

- If I have a database containing lamp prices, sales volume, ics (internal product cost) transportation cost, overhead allottment, .. for all lamp types. What will the profit be if I change transportaiton, or change the source of manufacturing.

- Predict machine failures and prescribe preventative maintenance by sound viabration, machine timing, shrikiage trends, and end of normal part life.

- Determine correlations between gas, fill, temperature, stack up, and shrinkage.

In order to solve these problems, the Intelligent Management System must:

- *Sense*: Automatically acquire state data. Sense the location of objects, state of machines and status of activities both on the plant floor and in supervisory departments.

- *Model*: Model the organization at many levels of abstraction. For example, machines, people, materials, orders, departments, need to be modelled in detail from both an attribute and a process view, including their interactions, authority, and communication.

- *Operate*: Provide expert assistance in the accomplishment of complex professional tasks within the organization.

- *Manage*:

  - Analyse and manipulate the model to schedule production and resource utilization, and answer short and long term state and planning questions. The system in this role is *passive* in that it responds to user initiated queries.

  - *Actively* monitor the organization and inform responsible personnel when important events occur. For example, when a machine break down occurs, not only is the foreman informed, but also maintenance, and the salesman who must inform the client that the order will be delayed.

- *Analyse-Optimize*: Analyse how the structure and the processing of the organization should be changed to further optimize some criteria such as cost, throughput, and quality.

The above goals are concerned more with "what" functinality is provided than "how" it is provided. Included in the functional goals of IMS is the need for better user interfaces. The following scenerios depict how the user interface should be integrated with the rest of the system.

"Tell me when ...": The marketing manager is under heavy pressure to get a rather large order out of the factory. He wants to be informed the minute it is shipped. He turns to his terminal and types the message: "Inform me when order X is shipped." His User Interface Process (UIP) translates the request into a rule "IF order X is Shipped THEN send message to manager Y", and sends it to the shipping Task Management Process (TMP). The TMP monitors the system to determine when the rule condition occurs. When the order is shipped, the TMP interprets the rule, resulting in the shipping message being sent.

"You've got problems ...": The milling machine breaks a tool while cutting a high priority order. The machine and order are damaged. The machine's sensors transmit the information to its Machine Supervisory Process (MSP). The MSP analyses the problem, and shuts down the machine. It then informs the floor supervisor and the scheduler TMP of the breakdown; the scheduling TMP re-routes orders. A message is also sent to the maintenance TMP which allocates a maintanence person to fix the machine. Lastly, the MSP checks the importance of the order, and informs marketing and other personnel of the problem, if it affects their tasks.

"What if ...": The manager in charge of production is considering the problem of a continually large back-log of orders. Should another machine be bought, or should the orders be subcontracted? He/she turns to his/her terminal and types in: "What are the effects on orders over the next six months if we buy another machine X?" The system then enters into a dialogue with the manager determining other information required to analyse the question. The UIP then scans the system wide functions that may help answer the problem. It finds a simulation module that can analyse structural changes in an organization. It gathers the initialization data and alters the factory model. It then runs the simulation, analyses the output and provides the manager with the answer, and further explanations.

Obviously, creating a system that satisfies these goals will require many years of research and development. Never the less, it is important to understand the long-term goals so that short-term research is properly directed.

3 Architectural Goals

A necessary characteristic of any management system is that it provide the requisite functionality. But in most cases this is not sufficient. Unless the organization and its environment is static, system functions will quickly become inappropriate and fall into disuse. When the software maintenance is quoted to be 90% of software costs, this is due more to software adaptation than to software errors. We believe it is just as, if not more important, to attend to architectural goals as to functional goals. Then if the provided functionality is not applicable, the architecture will reduce the complexity of alteration.

In IMS, three levels of architecture are distinguished:

Hardware: The hardware level encompasses both the number and types of processors, and their networking.

Software: The software level covers the process architecture, process description languages, protocols for communication, module functionality, and databases.

User Interface: The user interface level covers the means by which the user can communicate with the system functions.

As an organizations changes, the hardware must also change, reliably providing services at the point of need. While timesharing systems are *accessible*, i.e., can provide services at the point of need, their *reliability* and *adaptability* are limited. (Some) Distributed systems connected by general communication networks can expand on demand, provide service at the point of need, and also shift processing loads when nodes fail.

Software must be *general* in the sense that modules, i.e, programs, can be used in more than one situation or application. For example, a factory modelling system can support more than one application such as simulation and scheduling. If the software does not provide the functionality it should not be difficult to *extend*. It should be extendable in the sense of program functionality, and in the sense that new modules and processes can be added to the system without requiring alterations to existing software. Currently, software systems are tightly coupled, requiring extensive re-programming whenever changes are required.

Lastly, the user interface must have the following characteristics:

Accessibility: Interfaces to computer systems are usually idiosyncratic and difficult to learn and use. Also, systems that change require that their users be continually re-educated. Our goal for IMS is to enable all personnel to meaningfully communicate with it. The interface will gracefully interact with the user and provide guidance and help in deciding what the user needs.

Accountability: A major obstacle to computer acceptance is that users are unable to question how and why output was generated. Our goal is to construct an explanation system which will allow IMS to explain its actions at various levels of detail.

Adaptability: As the user's needs change, the interface must be able to alter its processing and responses to fit the changes.

The results of our research have been targeted to run in a distributed, multi-processor and process environment (figure 1). Employee's will have a User Interface Process (UIP) that will act as an intelligent "aide". A UIP is composed of a personal computer, graphics display, keyboard, microphone, and network interface (e.g., a SPICE machine (CMU-CSD, 1979)). The UIP will have either voice or typed natural language input. It will act as an "aide" in the sense that it will interpret and implement user requests and queries. All UIPs will be inter-connected via a communication network allowing them to cooperatively interact to solve problems and communicate information. The UIP will also carry out many of the employees well-structured tasks automatically. Each machine will have a Machine Supervisory Process (MSP) which monitors and controls it. It is also connected to the network, and can reply to queries and commands initiated by other MSPs or UIPs on the network.

Lastly, there are Task Management Processes (TMP). A TMP provides the focus for task management. It does more of the mundane task monitoring and control, freeing managers to do the more complex decision making tasks.

4 IMS Applications

The construction of an intelligent management system is a long term project. Nevertheless, progress is being made by looking at problems at various levels within an organization. In the folowing, four levels are recognized:

1. Management level activities such as planning,
2. Professional level activities including engineering,
3. Production level activities including scheduling and inventory control, and
4. Processs level activities including machine control and diagnosis.

In each level, one or more projects under investigation in the Intelligent Systems Laboratory are described briefly.

4.1 Management Activities

4.1.1 Callisto: Project Management

[1]Innovation is playing an increasing role in the continued vitality of industry. New products and innovations in existing products are occurring at an increasing rate. As a result, product lives are ever decreasing. In order to maintain market share, companies are being forced to reduce product development time in order to enter the market. By entering the market as early as possible, the ever decreasing product life will be extended.

A major portion of the development cycle is consumed in the performance and management of activities. For example, in high technology industries such as the computer industry, thousands of activities are required to be performed in the design and prototype build of a new product. Poor performance or poor management of an activity can result in critical delays. If product development time is to be reduced, then better management and technical support should be provided to each of the activities.

The Callisto project examines the extension and application of artificial intelligence techniques to the domain of large project management. Managing large projects entails many tasks, including:

Plan Generation and Scheduling: selecting activities and assigning resources to accomplish some task.

Monitoring and Control: monitoring the status of parallel activities in order to ascertain both plan and schedule changes required to meet project goals.

Product Management: maintaining a current description of the product (which is usually the outcome of a project), and determining the effects of changes to its definition (e.g., engineering change orders).

Resource Management: acquisition, storage, and assignment of the many resources required to support a project.

A close observation of project tasks shows that errors and inefficiencies increase as the size of the project grows. The successful performance of project tasks are hindered by:

Complexity: due to the number and degree of interactions among activities (e.g., resources, decisions, etc.).

Uncertainty: of direction due to the unknown state of other activities and the environment.

Change: in activities to be performed and products to be produced, requiring project flexibility and adaptability.

While CPM and PERT techniques provide critical path and scheduling capabilities, the bulk of the tasks are performed manually.

Callisto provides decision support and decision making facilities in each of the above tasks. The ability to extend the capabilities found in classical approaches is due to Callisto's project model. Starting with the the SRL knowledge representation language, a set of conceptual primitives including time, causality, object descriptions, and possession are used to define the concept of activities and product. The language is further extended by the inclusion of a constraint language, representing the constraints amongst activities. The modeling language provides Callisto with the ability to model both products and activities in enough detail that inferential processing may be performed.

Callisto's decision support and decision making capabilities include:

- interactive change order management for products,
- multi-level scheduling of activities,
- rule-based analysis and maintenance of activities, and
- automatic generation of graphic displays of project models.

These functions are constructed from a combination of three problem-solving architectures:

Event/Agenda Based: Callisto can interpret a user's process, represented as a network of activities and states, by setting up and maintaining an agenda of goals and monitored events. This processing facility is used by the scheduling algorithms.

Rule-Based Programming: PSRL, a production system language built on top of SRL, is used to implement managerial heuristics for project management. PSRL can monitor and act on arbitrary SRL conditions.

Logic Programming: HSRL, a Horne clause theorem prover, is used as a question answering mechanism. HSRL represents assertions and theorems within SRL.

Current research focuses on a distributed Callisto system. Each member of a project has a "mini-callisto" to aid in managing their task. Each mini-callisto is able to communicate with other mini-callisto s to collectively manage the entire project.

---

[1]Callisto research is supported by Digital Equipment Corporation. Research is performed by Mark Fox, Michael Greenberg, Arvind Sathi, Mike Rychener, Joe Mattis, and Drew Mendler.

A version of Callisto is currently being tested at Digital Equipment Corporation.

4.1.2 Rome: Long Range Planning

[2] Many business decisions are based on information produced by computerized financial planning models. While the models themselves may be quite sophisticated, their computer implementations generally do little more than calculate and display the results. Not much attention is given to screening the input data for anomalies, verifying that the data satisfy the assumptions of the model, or checking to make sure that the outputs seem reasonable for the situation at hand. Nor are there facilities for explaining what the outputs represent, showing their derivation, or justifying the results to users who are not familiar with what a particular program does. Traditionally, these tasks have been left to human analysts who could intelligently apply a programmed model to answer managerial questions.

The ROME project (Kosy & Dhar, 1983) is an effort to develop a knowledge-based system which could itself perform many of the above tasks and hence more effectively support decision-making in the area of long-range financial planning. Our approach is based on the idea that current programs are limited by a lack of knowledge, i.e., they simply don't know what the variables in the models they manipulate mean. For example, they don't have knowledge of how the variables are defined in terms of real-world entities and so they can't explain what the variables stand for. They don't themselves keep track of the relationships used to derive the variables and so they can't explain how they got their values. They have no knowledge of "normal" versus "abnormal" circumstances and so cannot detect peculiar values, whether they be for input, intermediate, or output variables. Finally, they have no sense of the consequences implied by the variables and hence cannot tell "good" values from "bad" ones with respect to the goals of the organization.

In contrast, our overall goal for ROME is to make the meaning of the variables available to and usable by the system itself. Therefore, we have developed an expressive representation for financial models using the SRL1.5 knowledge representation language. This representation allows ROME to keep track of the logical support for model variables, such as their external source, method of calculation, and assumptions that must hold for the variable's values to be valid. Tracing back through the dependencies associated with a variable's computation can be used to explain why a value should be believed. Similarly, ROME can challenge the values of a particular variable by comparing them against relevant expectations, organizational goals, and independently derived values. Prototype implementation of two ROME subsystems, called ROMULUS and REMUS, has recently been completed.

ROMULUS: Reason-Oriented Modeling Using a Language Understanding System. ROMULUS is the user interface for the ROME system. Instead of the rigid and stylized input language used with most computerized support systems, ROMULUS has been designed to accept natural language queries about the model expressed in English sentence form. The query types currently understood are those which relate to definitions and calculations such as "What is the definition of production spending?" and "How was line 46 calculated?" ROMULUS also supports the interactive construction and editing of financial models in natural language, by allowing the addition of new variables, formulas and constraints on variables. Examples of acceptable user assertions are "Define year end people to be direct labor + indirect labor", and "Expect direct labor to go up." A major goal for ROMULUS has been to make the system as cooperative as possible, by including ways to recover from user mistakes (e.g. by spelling correction) and to tolerate user ignorance (e.g. by accepting synonyms and variations in syntactic form).

REMUS: Reviewing and Analyzing a Model's Underlying Structure. REMUS is the financial model reviewing expert for the ROME system. Given a financial model and a set of constraints entered by users which represent plan reviewer expectations and corporate goals, REMUS scans the model to detect constraint violations, which are then reported to the user. When a constraint violation is detected, REMUS attempts to determine the underlying circumstances that account for it by examining the formulas, input and intermediate variables that involved in it's computation. By this process, REMUS can localize the source of a constraint violation to the input variable(s) which seem to be responsible.

An integrated version of the ROME system, called ROME1.0, was delivered to Digital in October of 1983. It is currently undergoing testing and development. We are presently involved in extending the capabilities of the ROME system in three areas: the causal diagnosis of constraint violations, the dependency-based revision of financial models in the face of inconsistency or change, and the support of user exploration of hypothetical plans. This work will involve major additions to not only the REMUS and ROMULUS subsystems, but to the SRL1.5 knowledge representation language itself.

[2]This research is supported by Digital Equipment Corporation, and is performed by Brad Allen, Don Kosy, Ben Wise, and Dave Adam.

4.2 Professional Activities

4.2.1 KBS: Knowledge-Based Simulation

[3] KBS (Reddy & Fox, 1982)is a tool for interactively constructing models and simulating them, based on Artificial Intelligence (AI) techniques. It has been under active research and development for over two years, and has been successfully applied to the modeling and simulation of factory organizations. Applications of KBS to date, included:

- Printed circuit board factory
- Alternate assembly plans in a factory aided by robots
- Computer terminal assembly and test
- Computer manufacturing
- Corporate distribution system

The current version of KBS provides the following features:

- creating a system modeling language that can simultaneously support multiple applications in addition to simulation. Thus eliminating the need and cost of maintaining multiple models.

[3]This research is supported by Digital Equipment Corporation, and is performed by Ramana Reddy, Mark Fox, and Malcolm McRoberts.

- representing the behavior of system entities directly in the model.

- allowing the system to be selectively instrumented. This restricts data analysis to areas of interest, and provides support of graphics displays.

- representing the system at multiple levels of abstraction. This allows the user to specify the level of simulation and the detail-level of results.

- consistency and completeness checking. Much time is spent verifying that models are consistent and complete. We have developed a checker which detects model incompleteness and inconsistencies.

- providing interactive access to the model building and simulation system. This appears to reduce model building and debugging time because it provides facilities for stepping through the model and thus provide an intimate understanding of the simulation.

- graphic display of model dynamics. This appears to be of great use in making the model builder understand the internal workings of the model and thus bolster confidence in the results produced.

- model management. This provides facilities for creation and management of scenarios thus providing interaction with a model in its various forms on a continuing basis.

The current focus of research in KBS is directed towards development of the following:

- Natural Language Interface

- Planning/Problem-Solving to analyze utterances and plan model alteration.

- Goal Directed Instrumentation to collect relevant data to draw inferences.

- Auto-analysis of model dynamics and validation

- Auto-selection of Abstraction Level

- Expectation-Based Analysis of Simulation Results

- Natural Language Consistency and Completeness

4.2.2 INET: Distribution Analysis

[4]The I-NET project (Reddy et al., 1983) is an application of Knowledge Based Simulation (KBS) techniques to the domain of corporate distribution management. Corporate distribution management provides a rich environment for studying new techniques developed in KBS. Consider a typical manufacturing organization which manufactures a number of products and whose components are manufactured in a number of widely separated locations. These components are warehoused and merged at different locations and distributed to reseller locations. In such a system there are numerous decisions that have to be made about the transportation, warehousing, manufacturing and order administration policies. The purpose of I-NET is to provide a simulation model which can be understood, modified and used by managers directly without the assistance of a programmer. These facilities should provide the manager with an indepth understanding of the distribution network and aid in decision making.

Currently the I-NET system consists of:

- A knowledge based network editor to create the corporate distribution network

- A demand editor to model demand distribution

- A map perusal facility to graphically display the corporate distribution network

- A report generator to facilitate comparison of the performance of various scenarios

Two newly created facilities of KBS: model management and graphic display of model dynamics are being applied to the I-NET project in order to study their utility and determine future directions for research.

4.3 Production Activities

4.3.1 ISIS: Job-Shop Scheduling

[5] The ISIS system (Fox, 1983; Fox et al., 1983) is an artificial intelligence constraint-directed reasoning system which addresses the problem of how to construct accurate, timely, realizable schedules, and manage their use in job shop environments. The contribution ISIS makes to the job shop planning and scheduling problem is its focus on the ***representation***, ***utilization***, and ***relaxation*** of constraints in the scheduling process. ISIS's knowledge representation language, SRL (Wright & Fox, 1983), can represent an extensive set of constraints and their relaxations. Categories of constraints which ISIS covers include organization goals (e.g., due dates, cost, quality), preferences (e.g., for machines), enabling states (e.g., resources, previous operations), physical characteristics (e.g., accuracy, size), and availability (e.g., existing reservations for tools). ISIS uses a constraint-directed search paradigm to solve the scheduling problem. ISIS provides:

- A knowledge representation language (SRL) for modeling organizations and their constraints.
- Hierarchical, constraint-directed scheduling of orders, which includes:
    - constraint-directed bounding of the solution space,

[4]This research is supported by Digital Equipment Corporation, and is performed by Ramana Reddy, Mark Fox, Malcolm McRoberts, Kevin Doyle, John Arnold, and Phil McBride.

[5]This research is supported by Air Force Office of Scientific Research and Westinghouse Electric Corporation, and is performed by Mark Fox, Brad Allen, Ranjan Chak, Stephen Smith, and Gary Strohm.

- context-sensitive selection of constraints, and
- weighted interpretation of constraints.

- Analytic and generative constraint relaxation.
- Techniques for the diagnosis of poor schedules.

Due to the conflicting nature of certain constraints (e.g., cost vs quality) there may not be a schedule which satisfies all of them. Hence ISIS considers relaxations of such constraints when generating and selecting schedules.

ISIS performs a limited amount of process planning. The constraint set utilized in ISIS includes constraints on the physical features of a product (e.g., size and form), and the ability of machines to produce them. During the constraint-directed search, infeasible process routings are removed by these constraints.

As schedules are implemented on the shop floor, unpredicted events are detected by comparing actual job and resource status to predicted. When a deviation is detected, a complete rescheduling is not performed. Instead, only the affected jobs are rescheduled, with an attempt to minimize their variation from their earlier schedule.

ISIS is the first system to consider the myriad of constraints found in job shops and generate schedules which meet them or their relaxations in polynomial time.

4.4 Process Control

4.4.1 PDS: Process Diagnosis System

[6] Research in the field of AI diagnosis systems has been evolving rapidly since the first event-based (Nelson, 1982) or surface (Hart, 1982) reasoning systems (Shortliffe, 1976; Pople, 1977; Fox & Mostow, 1977; Duda et al., 1978), to systems that have functional or deep knowledge of their domain (Davis et al., 1982; Genesereth, 1982; Underwood, 1982; McDermott & Brooks, 1982). Whatever the style of diagnosis, these systems assume that information is provided manually through a question asking/answering dialogue, or automatically by means of sensors, or other devices. In both cases, the information is handled in the same manner, which, we have found, should not always be the case. In applications where the sources of information may be errorful (e.g., sensors), we found that it is just as important for a diagnostic system to reason *about* the sources of its information and their veracity, as it is to perform diagnosis based on the information.

During the summer of 1981, we began the design and construction of a rule-based architecture, called PDS (Fox Lowenfeld & Kleinosky, 1983), for the on-line, realtime diagnosis of malfunctions in machine processes. Diagnoses would be based on information acquired from tens to hundreds of sensors attached to a process. During the application of PDS, a number of sensor related problems arose. First, the process sensors in our applications degrade over time, reducing their diagnostic value. Second, a properly operating sensor may provide spurious readings periodically due to factors exogenous to the process. Though the frequency of such malfunctions are small, their detection may result in substantial savings. For example, in the electrical power utilities, replacement cos's of electricity lost due to sensor malfunction averages $500,000/year/plant (Meijer et al., 1981). Any diagnosis system which is to receive its information directly from devices which possess these characteristics must be able to handle the information without providing incorrect diagnoses, or at least have its diagnosis degrade gracefully with the sensors. As a result, PDS was extended to deal gracefully with these problems. PDS implements techniques called retrospective analysis and meta-diagnosis as solutions to these problems. Retrospective analysis enables PDS to perform a variety of time analyses on sensor data. Meta-Diagnosis enables PDS to alter its rule-base automatically inorder to decrease the effects of degrading sensors.

PDS is about to enter production use in the monitoring steam turbines.

[6] This research is supported by Westinghouse Electric Corporation, and is performed by Mark Fox, Simon Lowenfeld, and Pamela Kleinosky.

4.4.2 Waste: Process Diagnosis and Simulation

[7] The WASTE project is an investigation into the use of causal models for the purpose of process diagnosis and design. The domain is the treatment of chemical wastewater. During operation of the treatment system pipes and valves clog, pumps break and sensors fail. While shallow fault models of the system might tie symptoms of these failures to the fault, they cannot operate robustly. When the problems are unusual or complex the shallow models are of little use.

We are also interested in sharing models between multiple applications. Models are difficult and expensive to both build and maintain. If a single model can be constructed which can be used for multiple functions, efficiencies have been achieved. Thus, a simulation model constructed to aid in design of a process which can be used later to also diagnosis process faults is of great value. A single model can live on with the system.

Hence, complex, quantitative simulation models of the kind used for process design and analysis are taken as a starting point for the diagnosis system. These are represented as SRL models and augmented by representations of the physical structure of the process' environment. Diagnosis is then performed based on these deep models. The complexity of these models, however, requires that a variety of tools be brought into play to make diagnosis possible. First, the simulation models typically consist of real valued functions. There are literally an infinite number of different configurations of these model's parameters that could be explored to find one which explains the process' faults. To abstract away some of the information, a qualitative interpretation of the simulation models is taken. The actual real-valued functions are transformed into equations which reflect only how the direction of change of one parameter effects the direction of change of other parameters.

Unfortunately, this still leaves a great number of diagnosis possibilities. Hence, further abstraction of the model is performed. The system is represented at multiple levels of abstraction to allow diagnoses to take place first on the simplier

[7] This research is supported by the Robotics Institute, and is performed by Mark Wright.

abstract models. As diagnoses are hypothesised, they are further tested at more detailed abstraction levels.

As a final tool, belief knowledge is used to help focus the search for correct diagnoses. Components that have a history of failure are noted as such and faults involving these components are be explored more quickly than those which are usually functioning. As the more common faults are exhausted, less and less likely faults are explored.

4.5 Organization Modeling

[8] The management and analysis of an organization requires a richness and variety of information not commonly found in the databases of management information systems. For example, a simulation system requires knowledge of existing processes including process times, resource requirements, and its structural (routing) relation to other processes. It must also know when routings for products are static, or are determined by a decision process such as a scheduler. In the latter case, it must know when and where to integrate the scheduler into the simulation. If the IMS is to generate the sequence of events to produce a new product, it must have knowledge of processes (e.g., machines) which includes the type of processing it can do, its operating constraints, the resources it consumes, and its operating tolerances. If data is to be changed in an interactive, possibly natural language mode, the IMS must have knowledge of generic processes such as machines, tasks, and departments if it is to understand the interaction. It must also know what information is important and how it relates to other information in order to detect missing information and inconsistencies. Hence, the organizational model must be able to represent object and process descriptions (structural and behavioral), and functional, communication and authority interactions and dependencies. It must represent individual machines, tools, materials, and people, and also more abstract concepts of departments, tasks, and goals.

[8] This research is supported, in part, by Air Force Office of Scientific Research, Digital Equipment Corporation, and Westinghouse Electric Corporation, and is performed by Mark Fox and Stephen Smith.

Current organizational models are found typically in databases fragmented across one or more computer systems. How information in the database is interpreted is defined by the program and not by agreed upon conventions of field and relation names (though work in relation schemata is proceeding). By taking an AI knowledge representation approach to organization modelling, the variety of information described above can be represented. The model is accessible by all subsystems while the semantics of the model is jointly understood. Secondly, an AI approach to organization modelling provides the information required by all the management and analysis functions.

While many of the information enumerated above can be represented using current AI knowledge representation techniques, there is still much that requires craftsmanship and is poorly understood. More work is required to standardize the representation of causal relations, data changes over time, and idiosyncratic inheritance relations.

To date, our research has focused on the use of the SRL knowledge representation system as the basis for organization modeling. SRL has been extended to include conceptual primitives such as:

- actions, states, and objects,
- constraints and their relaxations,
- time,
- causality and dependencies, and
- belief.

With these primitives, detailed models of over 5 plants have been constructed.

5 Conclusion

Most of the costs of producing products in complex organizations are attributed to overhead, much of which is comprised of managerial, professional, and salaried personnel. If significant productivity gains are to be made in this decade, more attention must be paid to aiding both the professional and the manager. The Intelligent Management System is a step towards this goal. By combining artificial intelligence, computer science, and management science techniques, more intelligent aids and solutions for the operation and management of complex organizations can be found.

This paper provides an overview to the Intelligent Management System. It provides a glimpse into a goals and the systems we are creating and have created to date. Each of these working applications represent a piece of an evolving Intelligent Management System.

6 Acknowlegements

IMS is the result of the efforts of many people in the Intelligent Systems Laboratory. They include David Adam, Brad Allen, Ranjan Chak, Don Kosy, Joe Mattis, Malcolm McRoberts, Drew Mendler, Tom Morton, Ramana Reddy, Arvind Sathi, Steve Smith, Gary Strohm, Ari Vepsalainen, and Mark Wright. Also, thanks to Lydia Leonberg for her assistance in preparing this document.

This research was supported, in part, by Digital Equipment Corporation, Air Force Office of Scientific Research under contract F49620-82-K0017, and Westinghouse Electric Corporation.

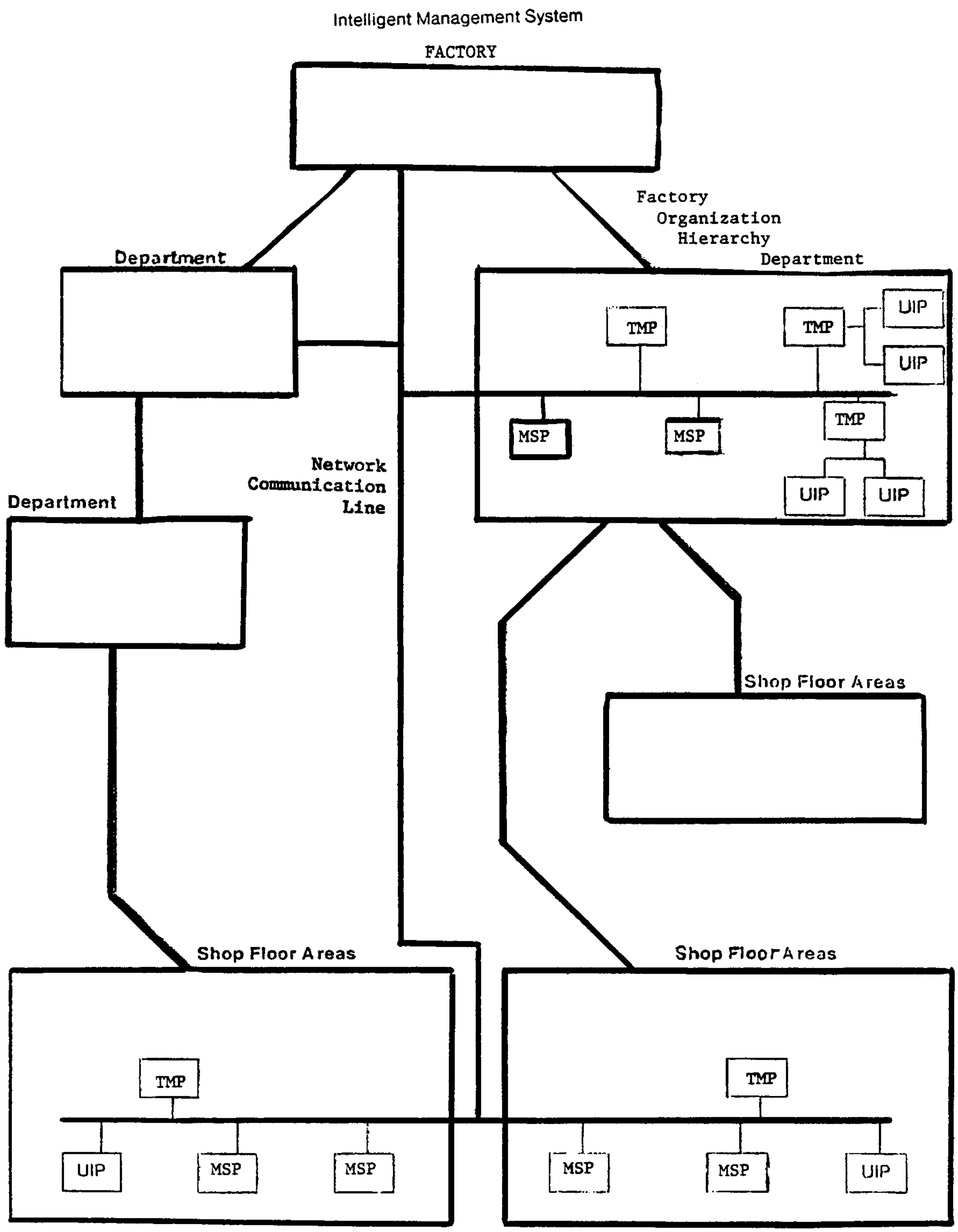

Figure 1: IMS System Architecture

7 References

Alter S.L., (1980), *Decision Support Systems: Current Practice and Continuing Challenges*, Reading, MA: Addison-Wesley Pub. Co.

American Machinist, (1980), "Machine Tool Technology", *American Machinist*, Vol. 124, No. 10, Oct. 1980, pp. 105-128.

CMU-CSD, (1979), "Proposal for a Joint Effort in Personal Scientific Computing", Computer Science Department, Carnegie-Mellon University, Pittsburgh PA.

Davis R., H. Shrobe, W. Hanscher, K. Wieckert, M. Shirley, and S. Polit, (1982), "Diagnosis Based on Description of Structure and Function", *Proceedings of the American Association for Artificial Intelligence*, Aug. 1982, pp. 137-142.

Duda R.O., P.E. Hart, P. Barrett, J.G. Gaschnig, K. Konolige, R. Reboh, and J. Slocum, (1978), "Development of the Prospector Consultation System for Mineral Exploration: Final Report", Tech. Rep., SRI International, Menlo Park CA, Oct. 1978.

Erman L.D., F. Hayes-Roth, V.R. Lesser, and D.R. Reddy, (1980), "The Hearsay-II Speech-Understanding System: Integrating Knowledge to Resolve Uncertainty", *Computing Surveys*, Vol. 12, No. 2, June 1980, pp. 213-253.

Fox M.S., (1981), "The Intelligent Management System: An Overview", Technical Report CMU-RI-TR-81-4, Robotics Institute, Carnegie-Mellon University, Pittsburgh, PA, July 1981. Also appeared in *Processes and Tools for Decision Support*, 1983, H.G. Sol (Ed.), North-Holland Pub. Co.

Fox M.S., (1983), "Constraint-Directed Search: A Case Study of Job-Shop Scheduling", (PhD Thesis), Technical Report, Robotics Institute, Carnegie-Mellon University, Pittsburgh PA.

Fox M.S., B. Allen, S. Smith, and G. Strohm, (1983), "ISIS: A Constraint-Directed Search Approach to Job-Shop Scheduling", *Proceedings of the IEEE Computer Society Trends and Applications*, Nation Bureau of Standards, Washington DC. An expanded version appeared as technical report CMU-RI-TR-83-3, Robotics Institute, Carnegie-Mellon University, Pittsburgh PA.

Fox M.S., S. Lowenfeld, and P. Kleinosky, (1983), "Techniques for Sensor-Based Diagnosis", *Proceedings of the International Joint Conference on Artificial Intelligence*, Karlsruhe, West Germany, August 1983.

Fox M.S. and D.J. Mostow, (1977), "Maximal Consistent Interpretations of Errorful Data In Hierarchically Modeled Domains", *Fifth International Joint Conference on Artificial Intelligence*, Cambridge MA, 1977.

Genesereth M.R., (1982), "Diagnosis Using Hierarchical Design Models", *Proceedings of the Second Conference of the American Association for Artificial Intelligence*, Aug. 1982, pp. 278-283.

Hart P., (1982), "Direction for AI in the Eighties", *SIGART Newsletter*, No. 79, Jan. 1982.

Kosy D.and V. S. Dhar, (1983), "Knowledge-Based Support System for Long Range Planning", Technical Report, Robotics Institute, Carnegie-Mellon University, Pittsburgh, Pennsylvania, December, 1983.

Lowerre B., (1976), The HARPY Speech Recognition System, (Ph.D. Thesis), Tech. Rep., Computer Science Dept., Carnegie-Mellon University, Pittsburgh PA.

McDermott D. and R. Brooks, (1982), "ARBY: Diagnosis with Shallow Causal Models", *Proceedings of the Second Conference of the American Association for Artificial Intelligence*, Pittsburgh PA.

McDermott J., (1980). "R1: an Expert in the Computer Systems Domain", *Proceedings of the First Annual National Conference on Artificial Intelligence*, Stanford University, Aug. 1980, pp. 269-271.

Meijer C.H., J.P. Pasquenza, J.C. Deckert, J.L. Fisher, D.B. Laning, and A. Ray, (1981), "Online Power Plant Signal Validation Technique Utilizing Parity-Space Representation and Analytic Redundancy", Electric Power Research Institute, Technical Report EPRI NP-2110, Nov. 1981.

Nelson W.R., (1982), "REACTOR: An Expert System for Diagnosis and Treatment of Nuclear Reactor Accidents", *Proceedings of the Second Conference of the American Association for Artificial Intelligence*, Aug. 1982, pp. 296-301.

Pople H., (1977), The Formation of Composite Hypotheses in Diagnostic Problem Solving: An Exercise in Synthetic Reasoning. *Proceedings of the Fifth International Joint Conference on Artificial Intelligence*, Cambridge, Aug. 1977.

Reddy Y.V., and M.S. Fox, (1982), "Knowledge-Based Simulation", Technical Report, The Robotics Institute, Carnegie-Mellon University, Pittsburgh PA.

Reddy Y.V., M.S. Fox, K. Doyle, and J. Arnold, (1983), "INET: A Knowledge-Based Simulation Approach to Distribution Analysis", *Proceedings of the IEEE Computer Society Trends and Applications*, National Bureau of Standards, Washington DC.

Shortliffe E.H., (1976), *Computer-Based Medical Consultations: MYCIN*, New York: American Elsevier.

Simon H.A., (1960), *The New Science of Management Decision*, New York: Harper and Row.

Underwood W.E., (1982), "A CSA Model-based Nuclear Power Plant Consultant", *Proceedings of the Second Conference of the American Association for Artificial Intelligence*, Aug. 1982, pp. 302-305.

Wright J.M., and Fox M.S., (1983), "SRL/1.5 User Manual", Robotics Institute, Carnegie-Mellon University, Pittsburgh PA.