A supervisory control expert system can coordinate failure diagnosis, assessment of process behavior, automated controller re-tuning or reconfiguration and consequently provides intelligent process control. A supervisory control expert system with object based knowledge representation has been developed such that the general knowledge confined to classes and subclasses forms a core knowledge base which can be used as primitives for rapid prototyping of intelligent control systems for various chemical processes. The expert system is embedded in an application program and communicates with the control system structured in the form of control blocks. The expert system prototype is developed for retuning model-based controllers to improve the controlled behavior of a packed-bed chemical reactor system.

I. INTRODUCTION

Use of expert systems in chemical process industries has increased significantly in recent years. Most applications are monitoring or diagnosis activities which act as advisory systems to plant operators. Training of expert system technology for process control is an attractive idea but since automatic control eliminates the plant operator from the control loop there is considerable concern in implementing expert control systems in plant environment. The consensus among most plant managers is not to close the industrial control loops with expert systems embedded in the controller, until the advantages, reliability and safety of real-time on-line expert controllers have been proven. Two factors that will contribute to the acceptance of such expert control systems are the availability of appropriate development tools and the development of successful demonstrations. For simple tasks such as set-point optimization, a rule-based expert system shell may provide a sufficient development environment. For more complex tasks such as intelligent (fault-tolerant) process control, more sophisticated development tools that are capable of capturing efficiently the process and control system characteristics and their interactions are necessary. Efficient knowledge representation is crucial for rapid processing of information and inferencing, and recent reviews on real-time expert systems in process industries (McGraw, 1989; Rowan, 1989) confirm that for large, interacting processes an object-oriented knowledge representation is the best approach.

Expert systems have been applied at the supervisory level to tune controllers, to perform process and control system fault diagnosis, or to restructure the control system. Combination of these activities would yield an intelligent control system which permits optimized process operation with tighter control, faster and more accurate fault diagnosis and reduced process down-time. An intelligent control system which coordinates assessment of process behavior, fault diagnosis, controller tuning and restructurig has been proposed by Basila et al. (1988). This Model-Object Based Expert Control System (MOBES) is developed using the Nexpert Object shell by Neuron Data Corporation. It is embedded in an application program written in C and runs on a PC/386. This paper focuses on the structure of the MOBES prototype and on the implementation details. The MOBES prototype concentrates on a subset of tasks that will be accomplished by the final version of MOBES. The prototype tailors the process control algorithms for improving the behavior of a tubular packed-bed CO oxidation reactor. This prototype permits the evaluation of all key components of a real-time expert control system such as data acquisition, information dissemination to control blocks and object slots, inferencing, calls to external routines and control command implementation. An experimental reactor system is available at IIT for validation of the control strategies and tools developed.

The expert system is developed such that the knowledge coded in the object based representation is process independent at class and subclass levels. Process specific information is represented at the object level. The general characteristics of each equipment is described at class level with further refinement of properties specified at subclass level. These classes and subclasses are instantiated to specific equipment in the process at object level, their properties being inherited from higher levels in the hierarchy. This configuration permits rapid prototyping of intelligent control systems for other chemical processes. The development of the MOBES prototype took about 3 person-years. With the in-house experience gained during
prototype development, it is estimated that building an intelligent expert controller for other chemical processes of similar complexity would take about six months.

II. OBJECTIVES OF THE PROCESS CONTROL EXPERT SYSTEM

MOBECs has been developed for a chemical reactor system in order to provide robust, intelligent control over the entire reactor operating range. The expert system must be able to diagnose equipment failure and to recognize deteriorating control performance or drift toward instability and provide corrective action. Depending on the root causes of unacceptable process behavior, the remedies may include controller retuning, selection of a new sensor (new location assignment or measurement of another variable), activation of state estimation, selection of a new manipulated variable and/or control system restructuring. The discussion here is limited to the performance of the MOBECs prototype in detecting instability in an Internal Model Controller (IMC) and adjusting the filter time constant to stabilize the system.

In an earlier study (Chylla and Cinar, 1989), the performance of proportional-integral (PI) controllers and IMCs were investigated to control the CO oxidation reactor. Two regions of reactor operation were considered. In the smooth operating regions, a PI controller provided adequate control. At the sensitive operating points reactor performance under PI control was unsatisfactory but a third order IMC provided adequate control. Since the controller design was based upon a linearized, reduced order model of the process, the controller performance deteriorated as the operating point moved away from the steady state value used for linearization. Adjustment of the IMC filter time constant usually yields a satisfactory controller modification for small variations in the operating point. For larger drifts in operating conditions, it is necessary to have a method which slows down performance deterioration and consequently generates extra time for development of more effective cures such as control system restructuring.

III. EXPERIMENTAL SYSTEM

The experimental reactor system (Figure 1) is a packed-bed tubular reactor operated autothermally for the catalytic oxidation of carbon monoxide to carbon dioxide (Chylla and Cinar, 1989). The reactor behavior exhibits severe multiplicities and high parametric sensitivity in some ranges of operating conditions (Adomaitis and Cinar, 1988). Thermocouples measure the feed, coolant, reactor bed and off gas temperatures. The off gas CO content is monitored with an infrared analyzer to allow direct calculation of the extent of reaction. Mass flow sensors/control valves regulate the flow of feed gases and of the cold bypass. All of the transmitters, control valves and gas flow controllers are connected to a data acquisition subsystem (Daytronic 10K1), which is interfaced to a process control computer (PC/286). The process control computer performs all of the data transfer, control, and operator interface functions. A second computer (PC/386) contains the expert system and communicates with the process control computer over a serial interface. The expert system computer can access a Vaxstation 3100 workstation via Ethernet to execute computation intensive modeling and controller design programs written in Fortran. They are spawned by procedures and rule action clauses as child processes. This arrangement purposely reflects the multiple loop controller/process control computer structure that is found in modern distributed control systems.

IV. STRUCTURE OF THE EXPERT SYSTEM

Continuous processes are characterized by the high degree of interaction between plant equipment and process control entities due to the flow of material and energy within the process. The effects of process and control system faults propagate through the process with the flow of mass and energy. A single fault can manifest itself as multiple symptoms throughout the process, complicating the tasks of diagnosis and troubleshooting. Interaction can also cause corrective actions to affect other areas of the process. Because of the dynamic, highly interactive nature of the process, rule-based representations of the process would be quite inefficient and would result in large, poorly structured knowledge bases. Such an expert system would be difficult to manage and brittle, and would have excessive inference cycle times. For complex processes and intelligent process control expert systems, objects are the best knowledge representation paradigm. Objects use a slot-filler type of data structure and inheritance to represent knowledge within the expert system. Methods (procedures) can be attached to the object slots to obtain a needed value or perform an action if the value in the slot changes. Thus, an object based knowledge representation paradigm is capable of representing the dynamic relationships of a process control system. Figure 2 shows a portion of a mixing-tee object, including the IF-NEEDED methods used to obtain the values for the stream flow and composition slots (DOVs routines). The mixing-tee objects have similar methods that solve the material balances to calculate the outlet stream flow and composition. Furthermore, pointers may be included in the object slots. Pointers are used in MOBECs to describe the
process topology. For example, in Figure 2, the slot "INLET #1 UPSTREAM OBJECT" points to PIPE11 object.

The class-object structure of MOBECS was developed using an entity-relationship (E-R) knowledge engineering model (Date, 1986; Chen, 1976). Entities, represented as blocks in Figure 3, are directly mapped into objects in the knowledge base. Entity attributes, shown as ovals, are directly mapped into object properties (slots). The relationships between entities are represented as diamonds. Classification (ISA, CONNECTED-TO) relationships form the class-object structure of the knowledge base. The remaining relationships such as the input relationships in Figure 3, define the functions and interactions of the entities. These relationships form the basis for rules and message passing in the expert system. Mapping these relationships is more complex and dependent in part upon the rule format of the expert system shell.

The types of knowledge that must be represented in the expert system include causal relationships, performance criteria and monitoring methods, the process and control system structure and topology. The expert system also requires meta-knowledge to control the inferencing process, such as procedures for focusing on specific areas of the problem domain and the methods for conflict resolution. Finally, deep knowledge that describes the behavior of the process and process control engineering must be incorporated into the knowledge base. The data types in the knowledge base consist of temporal (real time) data, static parameter values such as alarm limits, logical (Boolean) values, and string descriptors. Knowledge can be synthesized into core knowledge bases and provided as primitives for the rapid prototyping of new expert systems (Glasgow and Graham, 1988). The knowledge base in MOBECS is structured following this principle. Knowledge is divided into general knowledge applicable to any control system and knowledge specific to the CO oxidation process. As shown later, general knowledge is confined to classes and subclasses in the knowledge base and process specific knowledge is contained in the objects. As a result, the class structure forms a knowledge base kernel that can be used in developing other supervisory control expert systems.

The class-object structure of MOBECS is organized into two orthogonal trees representing the physical process hierarchy and the control system hierarchy, and one network (graph) which captures the topology of the process and control system. The process control tree represents an ISA classification of control system entities (Figure 4). The physical process tree represents a similar classification of the process entities (reactor, jacket, pipe, etc.). The nodes of these two trees are classes that become more specific as the trees are traversed from the roots to the leaves. The leaves of the trees are objects that represent actual process and control system entities. The properties (slots) of the classes and objects are data structures of the form class.property and object.property, respectively.
The process and control system topology network represents the interconnections between the process and control system entities and a portion of this network is shown in Figure 5. The defining relationships in the network are IS_CONNECTED_TO and ISA_PART_OF. Objects representing pipes, mixing tees, and branch tees are used to form these connections. Process pipe conveys the mass and energy flows between process entities. Mixing tees combine several input process streams into a single output stream. Branch tees split a single stream into multiple output streams. Pipes and branch tees pass flow and composition data between upstream and downstream objects. The mixing tees also perform gross material or energy balance calculations for use in fault diagnosis and troubleshooting. The purpose of the topology network is to reduce the search space of rules when a fault is detected. Since the symptoms of a fault are propagated by material and energy flows, the topology tree serves to precompute the problem search space for a directed depth first search. When a fault such as an abnormal reactor temperature is detected, the expert system first examines the entity associated with the fault, which in this case is a particular thermocouple. If this entity is functioning correctly the search is expanded to all other entities connected to the first one. For the example case, if the thermocouples are functioning correctly, then the rules that examine the performance of the reactor are placed in the inferencing agenda. This process continues through the upstream entities until the root cause is isolated. The network structure is stored in the knowledge base as pointers. The upstream and downstream object properties contain the corresponding object names that act as pointers in the inferencing process. As shown in Figure 2, the mixing-tee object TEE5 has upstream objects PIPE11 and PIPE13 and the downstream object is PIPE14.

Causal relationships, performance monitoring functions and the meta-knowledge to control the inferencing process are represented as rules in the knowledge base. Rules for fault detection and performance monitoring are applied at the class level for all standard process control functions such as absolute, deviation, and rate of change alarms for measurements and controller outputs. Threshold alarms are used for performance indices such as integrated squared error (ISE) and integrated absolute error (IAE). Overrange alarms are used on transmitter objects to detect transmitter failure. The advantages of applying these rules at the class level are twofold. General process control functions are limited to the class portion of the knowledge base in keeping with the principle of a knowledge base kernel. This also reduces the problem of configuration management since objects can be added or deleted without the need to define new rules or delete unused references.

A simplified flow chart of the application program is shown in Figure 6. The application program initializes NEXPERT, loads the MOBECS knowledge base, maps the object property atom ID's into an array of pointers, and then instantiates the static parameters in the knowledge base with the values from the process control system configuration files. The function EXSYS COM tests the communication link to the process control computer and then receives the real time data transmission. Real time values are transferred to the knowledge base by the function DATA-MAP using the array of atom ID pointers. The function NXP CNTRL suggests (places on the inferencing agenda) a core set of 50 rules for fault detection and performance monitoring and then transfers control to the inference engine. A single object is used inside the knowledge base to control the inferencing and communication functions. Some slots of this object act as Boolean flags that are used in these rules as other slots receive integer return codes from external functions that are interpreted by rules associated with the function calls.
Still other slots contain integer values that specify actions to be performed by external functions such as controller tuning methods. A set of rules in MOBECS periodically checks the communication port for a data transfer semaphore. When the semaphore is received, the inferencing process is halted and control is passed back to the application program.

The core rule set is applied at the class level to detect faults in the process or control system. Based upon the results of the inferencing process, additional data such as static block parameters can be requested from the control system. MOBECS can also change individual block parameters such as controller gains or time constants. Finally, the expert system can completely restructure the control system by changing the control block and linkage array values. When modifications are made to a controller block, the control system forces the block to initialize to effect a bumpless transfer.

V. MOBECS PROTOTYPE TEST CASE

The initial application of the MOBECS prototype to the autothermal reactor system concerns the ability to detect instability in the IMC composition controller and detune the controller until stable operation is restored. The simplicity of this application is necessary since we are also testing the process control software, and the data acquisition and inferencing control functions of the expert system. The tests are conducted at both the robust and sensitive operating points. Each test consists of bringing the reactor to steady state at the desired operating conditions, perturbing the feed gas temperature, and then allowing the expert system to restore stable operation by adjusting the filter time constant of the IMC controller.

Two heuristic methods are used to detect instability or deteriorating performance of the composition control loop: (1) Threshold limits on the short term averages of the integral of the absolute error (IAE) and the integral of the squared error (ISE) performance indices, and (2) Moving average filtering of the absolute value of the control errors (Liu and Gertler, 1987). The process control system performs the actual calculation of the stability monitoring parameters and passes the values to the expert system along with the real time data. Because of the relatively long open loop time constants of the reactor system (about 30 min.), the process control scan time is set to 10 sec. The performance indices are averaged over 60 minutes. The moving average filter is applied over the last five values of the control error. The value of the moving average (MA) is compared to two threshold values, W1 and W2, where W1 < W2. If MA < W1, the system is considered stable. If MA > W2, the system is unstable. If W1 < MA < W2, a trend analysis test is applied to determine stability. The trend analysis consists of fitting the last 12 values of the moving average filter to a straight line using a least squares regression. If the slope of this line is positive, the system is considered unstable. In the initial application of their stability monitor, Liu and Gertler (1987) set the stability threshold values W1 and W2 based on actual operating experience. MOBECS extends this method by incorporating statistical process control (SPC) techniques to automatically generate the stability thresholds based upon the actual process capability. A moving average Shew chart is constructed for the controlled variables to determine upper and lower control limits for a given set of operating conditions. The rules for detecting instability using the moving average filter are shown in Figure 7. These rules are part of the knowledge base kernel and are applied at the class level. Each IMC object has a stability flag referenced as object.STABILITY_ALARM FLAG, a controller tuning enable flag referenced as object.TUNING_ENABLE_FLAG, and a tuning failure flag referenced as object.TUNING_FAIL_FLAG. There is also a global IMC stability flag, IMC2, that is a rule hypothesis. At the start of every inferencing cycle of the expert system, all of these rules are placed on the inferencing agenda for evaluation. The first rule tests the value of the moving average filter for instability for every object belonging to the class of controllers. The second rule performs the trend analysis test. The third rule invokes the IMC controller tuning procedure if the tuning enable flag is set. For testing purposes, the detuning program iteratively increases or decreases the IMC filter time constant. After the tuning procedure returns a new filter time constant, the fourth and fifth rules apply a limit test to determine if the tuning procedure has failed (i.e., the IMC law is no longer valid for the current process operating conditions). If the filter time constant is acceptable, the function DATA_MGR is executed to implement new value in the process control database.

Figure 6. Flow chart of the application program with the embedded expert system.
VI. CONCLUSIONS

MOBECs is a supervisory control expert system designed to tailor the process control system for a tubular autothermal reactor in response to process or control system faults, and changes in the process behavior. Specific functions include process fault diagnosis, control sysstem performance monitoring and troubleshooting, controller tuning and control system restructuring. As this work progresses, automation of sensor placement, state estimation and process identification will also be addressed.

The knowledge base of MOBECs captures the complex structure and interdependencies of process control systems into a compact form that includes structure, declarative and heuristic relationships, and procedural and model based knowledge. In our discussion we have shown how process and control systems can be directly decomposed into a class-object structure. Since process specific knowledge is confined to objects, the class structure of MOBECs forms a knowledge base kernel that can be generalized to other process control systems. Rules are similarly divided into general relationships that are applied at the class level, and process specific rules. General procedures such as material balance relationships are attached to slots at the class level and inherited down to the objects.

Process specific procedures such as the process model are contained in separate modules linked to the MOBECs application program.

The knowledge base structure of MOBECs permits rapid prototyping of supervisory process control expert systems for other similar processes. A new expert system can be created by specifying the objects and entity connections directly from a process flow diagram. Objects are instantiated with process specific knowledge such as transmitter ranges and tag numbers. Process specific procedures (process model, etc.) are linked to the application code. The control system function blocks and the expert system objects have similar data structures. The control and expert systems use the same configuration data files during initialization. The similarity of the controller block structures and objects provides a redundancy which facilitates debugging and permits elementary diagnosis and control at the control computer level when the expert system or the expert system computer becomes inoperational. Embedding the expert system in an application program provided an efficient way of data transfer. The need for and the functionality of the application program would change as the callable interface of Nexpert Object is modified in future versions of the expert shell.

ACKNOWLEDGMENT: Partial financial support provided by IIT Education and Research Initiation Fund, Johnson’s Wax Fund, and IIT-IITRI Operation Synergy is acknowledged.

REFERENCES


