1.1 Expert Systems

An expert system is an intelligent program that uses knowledge and inference procedures to solve problems that are difficult enough to require significant human expertise for their solution. The knowledge necessary to perform at such a level, plus the inference procedures used, can be thought of as a model of the expertise of the best practitioners of the field [20].

The knowledge of an expert system consists of facts and heuristics. The facts are a segment of information, that is agreed upon by experts in a domain and is publicly available. The heuristics are empirical pieces of knowledge about good judgement in the domain. Heuristics characterise the expert level decision making process in the field and are mostly private. The performance level of an expert system is primarily a function of the quality and size of the knowledge it possesses.

An expert system consists of

- a knowledge base composed of the domain facts and problem solving heuristics.
- an inference procedure for solving the problem utilising the knowledge base
- a working memory (global database) which acts as a repository for all data including the inputs for the current problem, the present problem state and any other relevant data.

Expert systems differ from conventional software in many important aspects. In expert systems, there is a clear demarcation between the knowledge about the
problem domain and the knowledge for applying the same in solving a problem [18]. The former is encoded in the knowledge base, while the later is implemented as an inference procedure of the expert system. Ideally, it should be possible to change the system, by simple additions and deletions to the knowledge base.

The potential uses of expert systems are many. They can be used to

- design
- monitor
- interpret
- analyze
- diagnose
- explain and
- consult.

Consequently, there are many application domains of expert systems. Some of the domains are

- mission planning
- signal analysis
- command and control
- manufacturing
- image analysis
- software engineering
- electronic warfare and adaptive control
- logistics
- real time monitoring and process control.
1.1.1 Rule Based Systems

The most popular way of representing domain knowledge in an expert system is by using production rules. An expert system using production rules to encode domain knowledge is termed a rule based system.

The cognitive strategies of most human experts in complex domains are based on the mental storage and use of a large collection of pattern based rules [50]. Rules are a powerful paradigm for acquiring, organising and utilising expertise. A well chosen rule base may maintain control over otherwise intractable explosion of combinatorial complexity [24].

The problem solving paradigm used in rule based systems is rule chaining. If problem solving is initiated with a set of conditions and moves towards a conclusion, the method is called forward chaining. Production systems are a well known type of rule based systems in which the control structure can be mapped into the forward chaining paradigm. The production system interpreter embodies a mechanism that implements forward reasoning through repeated execution of Match-Select-Act cycles [53, 92]. If the conclusion is known, but the path to the conclusion is not known, then it is necessary to work backwards. This method is termed backward chaining. Forward chaining (data driven reasoning) is used for functions like monitoring, interpretation and analysis. Backward chaining (event driven reasoning) is mostly used for diagnosis and to a limited extent for functions like design and planning.

Rule based systems are used successfully in many domains. To deploy the rule based systems in domains requiring real time performance several methods have been suggested. They are

- Use of faster technology: As fast hardware becomes available, it can be used to speed up production system execution.

- Use of better algorithms: According to Forgy [22], in forward chaining systems the match phase contributes 90% of the run time and is a bottleneck that needs to be improved. TREAT [51] and LEAPS [54] are examples of fast match algorithms.
• Use of better architectures: Architectures like asynchronous production systems [79] are employed.

• Use of innovative inference procedures: The process of firing a single rule in a cycle leads to large number of inference cycles. Multiple rule firings within a single inference cycle can be used to speed up rule based system execution [31, 33, 40, 81].

• Parallelism: Parallelism inherent in a rule based system like match parallelism, act parallelism [27, 83] and also application parallelism [29] are exploited.

1.1.2 Blackboard Systems

The blackboard model is a scheme proposed for organising reasoning steps and domain knowledge to construct a solution for such problems. A blackboard model consists of [59]

• Knowledge sources: Knowledge needed to solve the problem is partitioned into knowledge sources which are separate and independent.

• Blackboard: The problem solving state is kept in a global data store called the blackboard. Knowledge sources produce changes in the blackboard. Communication and interaction among knowledge sources is through the blackboard

• Control: The control component decides which knowledge source can be applied when and to what part of the blackboard.

The problem solving behaviour of a system is determined by the knowledge application strategy encoded in the control module. The choice of the appropriate knowledge application strategy is dependent on the characteristics of the application.

1.1.3 Model Based Systems

Model based systems are thought to be suitable for diagnosis and design applications. Knowledge for diagnosis and design applications consists of descriptions of hardware
and software components and their functions, causal descriptions of how they achieve their functions, malfunction modes, design plans and links to underlying scientific knowledge [48]. A coherent description of all the above is a model. Model based systems are based on reasoning from first principles.

### 1.2 Requirements of Real Time Expert Systems

The above three architectures are widely used in implementing knowledge based expert systems. In traditional application domains, the performance of the expert system is measured in subjective terms like correctness of solution and its quality. Response time is secondary and no specific requirement exists. However in real time applications the timing behaviour of the system is a significant parameter. Expert systems that support reasoning with real time data and its temporal properties are called real time expert systems. In real time expert systems, the time at which the solution is produced plays a major role in determining the correctness. Real time expert systems can broadly be classified into monitoring and control systems. A real time monitoring expert system puts to use the incoming data and generates advice at a rate greater than or equal to the rate at which data arrives into the system. A real time control expert system always produces a response at or before the time at which the response is required [42].

Typical real time domains like military, industry, space and medicine are pregnant with a variety of possible applications. These real time application domains possess peculiar characteristics that differentiate them from other expert system application domains. Jakob et.al classified these characteristics along three dimensions - time, complexity and uncertainty. Another dimension that has been added is fault behaviour. The characteristics of the application domains along these dimensions are [34]

- **Time**
  1. Continuous operations: Plants operate continuously, making vigilance and maintenance operations difficult.
2. Propagation delays: Since the plants are composed of physical systems, delays are inherent to the system. Changes in the system do not occur instantaneously but, do so, over a period of time. This is true for both normal and abnormal states.

3. Evolution: The system is continuously changing and therefore, there is no single discrete steady state but only a continuum of steady states.

• Complexity

1. Structural: Many diverse subsystems are brought together and structural diversity is inherent to the process system. Monitoring as many as 2000 to 3000 signals is not uncommon in such system.

2. Behavioural: The structural complexity imparts a certain amount of behavioural complexity too. Even though steady state behaviour of the system can be predicted, it becomes impossible to predict all possible failure modes and their combinations. Failures can combine in unexpected ways and produce disastrous consequences like Three Mile Island case.

3. Operational modes: In different operational modes, like start-up, steady state, shut down and failure modes, the system behaves differently.

4. Multiple models: Different models of the system become relevant in different operational modes. Mathematical models are apt for steady state operations, but could fail to explain system behaviour in start-up, shut down or failure modes. Even in steady state, different aspects of the system may require different models. For example in a nuclear power plant, steady state operations could need at least two models one for the nuclear fission and, another for the generator mechanisms used.

• Uncertainty

1. Data: Data are obtained from sensors. Sensors themselves being physical systems prone to errors and degradations, events like erroneous data or data washouts should always be considered. Further scarcity of sensors due to design flaws, cost constraints and other limitations could also cause non availability of data.
2. Model: The second most important source of uncertainty is the model of the system. A model, determines the behaviour of the given system in a specific bandwidth of operation. It is possible that the present operational state of the system could fall beyond this bandwidth resulting in model failure. Further, the approximations used in the model could also result in uncertainty in the expected system behaviour.

- Fault behaviour

1. Multiple faults: When faults occur in process systems, they normally do not occur in isolation but occur in large numbers. For example, it is reported that in the Three Mile Island accident some 300 lights indicating alarms went on and off in the first few seconds. This clearly taxes the operator’s cognitive capability.

2. Propagated or chained faults: Due to tight coupling of various subsystems, faults propagate from one subsystem to another. The common cause for these apparently different faults must be identified and acted upon. Piece meal action independently for each fault can result in more disasters.

The above features of real time application domains require real time expert systems to possess some specific properties. Time is the most important entity, and the way it is handled characterises the system’s real time behaviour. Various issues like time representation, representation of temporal data, encoding of temporal knowledge and management of temporal reasoning are to be addressed, by the system architecture. The other essential characteristics are [36]

1. Continuous reasoning: Real time systems are characterised by continuous streams of input data. This implies continuous reasoning. If all data and conclusions are stored, very soon the demands on the storage system will out strip supply. Even if a large memory system is available the response time becomes too large to be acceptable. However the old data and conclusions will become invalid as new data come and their deletion will not hamper system
performance. So, it should be decided when and how data can be discarded to facilitate continuous reasoning.

2. Reactive response behaviour: The process system is dynamic and the expert system should react to the changes in the process system i.e. initiate reasoning immediately upon sensing change.

3. Temporal representation of data: Data in process systems is dynamic and varies with time. Propagation delay and slow state change in process systems necessitate the expert system to study trends, make forecasts and generate early warnings. Therefore a time based representation and storage of data must be available for the expert system to study historical data.

4. Reason about temporal relationships of events: Temporal relationships form an important input for fault diagnosis of process systems.

5. Focus of attention: The expert system should focus on important events in the system rather than divert attention to trivial and frivolous events.

6. Asynchronous arrival of events: Monitoring system in real time domains look for fixed threshold alarms. When process variables exceed these threshold values, alarms are generated. The expert system should accept these asynchronous alarms to take necessary action.

7. Model delayed feedbacks: The expert system should consider occurrence of delayed feedbacks in its model of the process system.

8. Handling of hardware and software interrupts: Since the expert system is a part of the total process control computing system, it is expected to handle software and hardware interrupts that can occur during the course of execution.

9. Embedded operation: Early expert systems are non communicative, and are available on a stand alone computer. This is not acceptable in real time systems. Ideally, they should be embedded in the target system itself (e.g. on board computers in space vehicle) or should accept data on line through a network from the process control computers.
10. Simultaneous Handling of Multiple Events: The expert system should be able to handle simultaneously occurring multiple events. These events can be either independent events or connected events in a chain.

11. Guaranteeing response in real time: This pertains to ensuring that either certain deadlines are met by the expert system or quick reaction is initiated in response to inputs. This is an area where least attention is focused in AI research. It is a difficult goal to achieve, due to the non-polynomial algorithms employed in AI systems.

1.3 Goals of Thesis

Though many expert systems are reported for real time applications, most of them, do not meet all the requirements enunciated in Section 1.2. The aim of the thesis is the design and development of an object-oriented real-time asynchronous production system for dealing with continuous streams of input data, meeting the following requirements.

- Temporal Representation of Data
- Representing and Reasoning about Temporal Relationships
- Modelling Delayed Feedbacks
- Continuous Reasoning
- Simultaneous Handling of Multiple Events
- Interruptible Reasoning
- Embedded Operation
- Reactive Response
- Focusing Attention
- Handling Asynchronous Events
1.4 Thesis Outline

Chapter 2 presents a survey of current techniques and tools in real time expert systems. A review of some real time expert systems is also presented.

Chapter 3 presents an Extended Petri Net formalism to model the knowledge representation, reasoning and knowledge verification aspects in real time expert systems. The architecture of REX based on the Extended Petri Net model is explained with a block schematic diagram.

Chapter 4 presents an object oriented scheme for representing data and knowledge of the real time domain. This representation is particularly tuned for temporal data and knowledge.

Chapter 5 discusses the match algorithm used in the inference engine. The techniques used in the algorithm for meeting specific real time knowledge representation requirements are also discussed. The inference engine uses a multiple rule firing strategy. This requires synchronising interfering rules. Chapter 6 presents a new interference analysis algorithm and the asynchronous rule firing strategy.

The concepts developed in these chapters are used in the implementation of REX. Chapter 7 details the implementation aspects of REX.

Finally in Chapter 8, the conclusions from this work are presented. Directions that can be pursued for enhancement the work are also discussed.