A low cost distributed expert system delivery environment for monitoring and control applications.

Tim Hendtlass
Swinburne University of Technology,
P.O.Box 218 Hawthorn 3122, Australia.
tim@brain.physics.swin.oz.au

Simon Goss
National Scientific Instrument Training Centre
P.O.Box 218 Hawthorn 3122, Australia.
goss@mulga.cs.mu.oz.au

Abstract.

A distributed data collection system developed for use in cost sensitive situations is described. The system design constraints included reliability, low cost, high flexibility, and real time performance. The resulting system which evolved is a distributed expert system. Each node of the distributed system supports its own inference engine, scheduler and rule base with context sensitive switching between partitions and fact bases. Each node can support an artificial neural network.

1. Introduction.

The concept demonstrator described is targeted as a support system for the elderly that is able to be retro-fitted to their own existing homes. The support of family or institutions is required to overcome simple problems arising as a result of poor health and forgetfulness as well as the fear of being helpless and alone. A non-invasive instrumented environment can perform many simple tasks releasing resources to other needs of the elderly person. This shifts the threshold of safety and allows the elderly the option of remaining in their own homes in circumstances which would otherwise require alternate arrangements such as moved to an institution. The system described has other potential uses in general distributed data acquisition, monitoring and control systems.

2. Design constants.

The system was developed for use in a cost sensitive situation. The system is designed for continual use and to be able to tolerate periodic communications failures. Each node in the system collects information and distributes it to all the other nodes with a need to know. Data is to be stored locally if internode communication is lost until communications are restored. For simplicity one node handles all communications with the outside world. A node has certain local responsibilities which it carries out as far as available data permits even if network wide access is lost. New nodes, or new functions for an existing node, may be added to update the system without compromising its normal activities. Updating might, for example, involve changing a transducer and its driver, or changing the rules to be evaluated at a node as the experience of the total network grows. Field modification requires compilation capability at the node to allow new program to be added. Compilation must be able to be executed concurrently with normal run time functions, which implies the use of multi-tasking. Forth was used in this project as it provides these facilities as well as producing very compact code.

3. The anatomy of a node.

A node is modular, consisting minimally of the software kernel, a communications module and a transducer interface. Nodes may also have an expert system shell and an artificial neural network (ANN).

In the target application of a domestic dwelling most nodes will be built into a power point with communication over the power lines and control over the appliances plugged into the point. Transducers collect data about the local environment which is shared with other nodes over the power lines. It is interpreted by the local expert system so local decisions may be taken. A node requests information that is not directly available to it from other nodes. A history of prior knowledge allows temporal reasoning by the expert system. An ANN can also learn from events over time and offer its evaluation of the current

0-8186-4260-2/93 $03.00 © 1993 IEEE

situations as a further input to the expert system. This helps adapt a set of general rules to the particular patterns of an individual occupant.

3.1 The expert system.

The expert system shell uses four state logic; true, false, unknown and unavailable. Data can involve recorded history as well as current values. A simple confidence value can be associated with the first two states. Rules are simple combinations of one or more conditional clauses and one or more consequent clauses. Conditional clauses consist of groups of IF statements that are connected by AND, NOT and OR. Consequent clauses can update any number of derived facts or run any number of routines. Initial evaluation is by forward chaining, all rules being evaluated until a complete pass is made without any new deductions. Evaluation then becomes backward, rules required but not to able be evaluated are inspected to see what information is preventing evaluation. The rule set is then inspected for a rule that would allow the sought after information to be deduced and the process recurses . When something is uncovered that cannot be established with the rules and data at hand, a request for the missing data is made to all the other nodes. On the receipt of this data, the forward then backward evaluation starts again. This process continues until no further information can be evaluated from both the local and shared data. The prior history of the node is updated and the process repeated at fixed time intervals. Any required action is performed by running the necessary software as necessary during the evaluation. A node always responds to requests for information it receives.

Rules may be grouped into a main set and a number of subsidiary rule sets. These subsidiary rule sets are called as needed from either the main rule set or from each other. They are normally called after establishing the need to know the information that the subsidiary rule set is designed to produced.

The rules for the expert systems at the nodes can be added to or pruned as needed. They are expected to be composed off line and down loaded to the nodes. Hence no rule construction and validation tools are included in the expert system shell designed for this project. The expert system shell is modular and allows unwanted features to be emitted from nodes that do not need them. These omitted features may be down loaded later while the system is running.

3.2 The artificial neural network.

Adaptive behaviour over time is provided by a back propagation ANN. This allows emergent relationships from the data to be used. The ability of an ANN to generalise and interpolate enhances the formal rule based system.

To maintain the relevance of the ANN, training must be a continuous on-going process. As an evaluation cycle starts the ANN is used to predict based on the current inputs, afterwards the ANN is trained based on the current situation and the previous inputs. Training using the last n {input at time t, situation at time $t+1$} pairs is preferable training the last data pair only. The learning rate must allow the underlying trends to be learned while still keeping the information current.

4. Discussion.

The requirements of the continuous real-time system described here include: data scanning over networks, temporal reasoning (collecting and analysing time varying data), calculation of derived quantities, logic solving in rule form and other pattern matching facilities. To fulfil these requirements the expressive power of an expert system was needed. This had to be distributed over a network of low cost hardware.

The system constructed meets these requirements. It also exhibits the complexities of real-time embeddable delivery systems: low level language capability is essential and debugging, validation and truth maintenance are complex issues.

The concept demonstrator monitored a region of a simple house. It was able to correctly identify simple conditions, such as the occupancy of a room, the status of entry and exit points (doors and windows open), power usage at a power point, ambient temperature and smoke detection. This information was used to successfully detect (simulated) emergency conditions. The concept demonstrator system described is a viable low cost solution to the particular application for which it was conceived. We believe that it has more general applications.

5. Acknowledgments.

The work described in this paper was supported by a grant from Swinburne University of Technology. The contribution of Mr J Billsborough in developing the communications software used in this project is gratefully acknowledged.