AGENT-ORIENTED MODELLING OF REAL-TIME SYSTEMS FOR CRITICAL EVENTS

R. KHOSLA
BSKM Laboratory
School of Business
La Trobe University
Melbourne, Victoria – 3086
Fax: (613) 94795971, Tel: (613) 94793064
E-Mail: R.Khosla@latrobe.edu.au

J. KUMAR
Dept. of Computer Science and Computer Engineering
Latrobe University
Melbourne, Victoria – 3086
Fax: (613) 9479 3060, Tel: (613) 94792374
E-Mail: j.kumar@latrobe.edu.au

ABSTRACT

In this paper we outline a task-centered agent-oriented approach to modelling (or software engineering) of intelligent real-time systems for critical events like environmental disasters (e.g., bush fires, earthquakes), alarm processing in critical and emergency situations in power systems, and in war situations involving detection and destruction of enemy planes by fighter pilots. We model the user (e.g., operator or fighter pilot) at two distinct levels. These include the problem solving or the task level and performance optimisation level. Alarm processing is used as an application to illustrate the modelling at the task and performance optimisation levels respectively.

1 Introduction

The problems associated with alarm processing have been bothering the minds of power system researchers since the 1977 New York blackout. The social and economic consequences of a major interruption in the supply of electric power are so great that every effort should be made to reduce the impact of a disturbance (Electrical World 1990; Ewart 1978).

Real-time alarm processing and fault diagnosis requires a response time of 100ms or less to assist an operator (who suffers from information overload – see Figure 1) in a power system control centre to isolate the faulty components or parts of a power network. A typical power network (e.g., Thomastown Terminal Power Station (TTS) with 16 substations in Victoria) consist of more than 100 alarm related parameters and a power system operator can be faced with as many as 1500 alarm events/minute appearing on computer screen. In this emergency situation, in the process of isolating the fault (in an extremely tight time frame) suffers from information overload leading to increased stress levels and diminishing cognitive speed as shown in Figure 1. In order to alleviate the problem of information overload, conventional intelligent technology based approaches like neural networks, expert systems and conventional software engineering approaches cannot be applied directly. These approaches do not model and capture the operator’s problem solving behaviour in an emergency situation which is critical to the acceptability of any recommendations/predictions (e.g. fault location and isolation) made by these technologies.

Figure 1: Operator Speed vs Stress Level © IEEE 1992 (Kirschen and Wollenberg 1992)

In other words, it is essential to humanise these technologies in order to facilitate their acceptability by the operators faced with management of critical events. In this paper we propose humanisation of intelligent technologies at task level and performance optimisation level. For that matter, the paper is organised as follows. Section 2 outlines some existing work done in the area of real-time alarm processing systems. Section 3 describes the Task layer. Section 4 describes the performance optimisation layer and shows the order execution precedence of a real-time alarm processing system. Section 5 concludes the paper.

2 BACKGROUND

In this section we discuss about the work/research carried out in the application area of alarm information processing and fault diagnosis of electrical power systems. Sun 2004 illustrates the fault diagnosis model which is built using fuzzy Petri Nets to diagnose faults arising due to incomplete and uncertain alarm information of protective relays and circuit breakers. However, the application in real-time is limited by excessive computational needs of fuzzy Petri nets. Tan et al. 2001 propose a fuzzy expert system for online fault diagnosis on a transmission network.

Souza et al., 2001 proposed a methodology in which neural classifiers are employed for alarm processing and fault location in Electrical power system. The ideas presented by Souza et al., Jan 2001 and Meza, 2001 are extended by
proposing a methodology called Nero Fuzzy Approach which combines the use of artificial neural networks and fuzzy logic for alarm processing and identification of faulted components in electrical power systems by the authors (Souza, et al, April 2004). Fuzzy relations are established between alarm patterns and possibly faulted system components and these alarm patterns form the inputs to artificial neural networks. As can be seen, most of the existing work on alarm processing is modelled phenomena of alarms and the technologies to model the phenomena rather than around the information processing behaviour of the operator in a power system control centre who is faced with an emergency situation of isolating the faulty part of the network based on information provided by the alarms.

3 Human-Centered Distributed Multi-Agent Soft Computing Architecture Critical Events

The layered multi-agent architecture is shown in Figure 2. It consists of five layers, namely, the object layer, which

![Figure 2: Human-Centered Layered Multi-Agent Soft Computing Architecture for Image Processing](image)

defines the data architecture or structural content. The distributed processing layer helps to define the distributed processing and communication constructs used for receiving and real-time processing of alarm data in a distributed environment. Intelligent technology layer defines the agents used for alarm processing for various statistical, morphological and intelligent tools. The optimization agent layer defines constructs for performance measurement, and performance optimization technologies which are used for measuring and optimizing the quality of solution (e.g., predictive accuracy of the neural network prediction agent). Finally, the problem solving ontology (task) agent layer defines the constructs related to the problem solving adapters1 namely,

1 Problem solving adapter is a generic problem solving primitive used for modeling a subset of a problem under study. The term “adapter” has its underpinnings in design patterns in software engineering.
Table 1 Functions of five problem solving adapters

<table>
<thead>
<tr>
<th>Phase</th>
<th>Goal</th>
<th>Some Tasks</th>
</tr>
</thead>
<tbody>
<tr>
<td>Preprocessing</td>
<td>Improve data quality</td>
<td>Noise Filtering Input Conditioning</td>
</tr>
<tr>
<td>Decomposition</td>
<td>Restrict the context of the input from the environment at the global level, by defining a set of orthogonal concepts</td>
<td>Define orthogonal concepts</td>
</tr>
<tr>
<td>Control</td>
<td>Determine decision selection knowledge constructs within an orthogonal concept for the problem under study.</td>
<td>Define decision level concepts with in each orthogonal concept as identified by users Determine Conflict Resolution rules</td>
</tr>
<tr>
<td>Decision</td>
<td>Provide decision instance results in a user defined decision concept.</td>
<td>Define decision instance of interest to the user</td>
</tr>
<tr>
<td>Post-processing</td>
<td>Establish outcomes as desired outcomes</td>
<td>Concept validation Decision instance result validation</td>
</tr>
</tbody>
</table>

preprocessing, decomposition, control, decision and postprocessing. This layer models practitioner's tasks in the domain under study. This layer employs the services of the other

4 layers for accomplishing various tasks. The five layers facilitate a component based approach for intelligent agent based software design. The generic agent definition used for defining the agents in various layers is shown in Figure 3.

3.1. Human-Centered Problem Solving Layer

Figures 4, 5, 6 and 7 model the problem solving or information processing behaviour of operator in a Thomastown Terminal Station control center using the problem solving adapters as defined in Table 1. These figures also show the mapping/association between the problem solving agent of the problem solving agent layer and the alarm objects in the object layer. For example alarm objects like voltage dip, network, communication; emergency and routine shown in Figure 5 represents the orthogonal (uncorrelated) functional concepts employed by the operator for processing alarm related information in a critical or emergency situation. More information on the earlier versions of the architecture and agent definition can be found in Khosla (2004).
4 Optimisation Layer

In Figure 8 the performance optimisation agent is responsible for optimising parameters in the parameter belief base (e.g., weight matrix of a neural network used for predicting faults or faulty components in a power network). The optimised parameters are used to modify the weights/structure of the neural network prediction agent (e.g., neural network agent predicting faults in a power system network). The parameter belief base represents the parameters used by the NN prediction agent in a particular prediction cycle. These parameters may be changed by the performance optimisation agent to improve the performance of the prediction agent. Predictions produced from the prediction agent are, of course, based on the data from the database of historical alarm data. The prediction results can be assessed manually (by user/operator) or by performance measurement agent (once it has been trained on operators' feedback over a period of time). Initially, the feedback is provided by the operator to the performance measurement agent. It simply communicates the negative
feedback to the performance optimisation agent. Overtime/with enough training/learning based on user feedback, the performance monitoring agent will be comparable to the human agent and takes over most of the performance measurement/assessment jobs from human counterpart.

The goal of the optimisation agent is to enhance the performance of the data mining system. Its task may contain adjusting parameters for some data mining technologies like weights in a neural network or changing the structure of Bayesian network. Optimisation can also be implemented by configuration of hybrid approaches through fusion, combination and transformation (Khosla 2000 – Intelligent Multimedia-multi-agent Systems – a Human-centred approach). The goal of performance measurement agent is to bridge the gap between the optimised alarm processing system and the existing alarm processing system. The task of the performance measurement agent is to identify and quantify user feedback (either discrete or continuous) for the measurement of the quality of prediction of the artificial neural network fault prediction agent.

The performance optimisation agent can be triggered by the decision phase agent. The performance measurement agent and the performance optimisation agent work together, to realise optimisation as shown in Figure 8.

Figure 8 Configuration of Performance measurement agent & performance optimisation agent in the Optimisation layer

Figure 9 shows the order of execution precedence of various real time alarm processing problem solving agents based on the modelling outlined in section 3.1.

5 Conclusion

This paper describes the design vocabulary for modelling the problem solving behaviour users/operators involved in critical events as well as the architecture for optimising the performance of the intelligent system used in critical events on a continuous and ongoing basis based on user/operator feedback. An Real Time Alarm Processing System (RTAPS) is used as an example to illustrate the application of the problem solving agent layer and optimization layer of a multi-layered multi-agent architecture for critical events. The description and application of the problem solving layer and optimization layer help to highlight the specific software engineering aspects for design and acceptability of critical event systems in industry as against design of conventional systems.

REFERENCES

Figure 9 Order of Execution Precedence of the Real time Alarm Processing System

32