Explanation, training and decision support for process control

T M KHAN, *J KWAAN, **K E BROWN, *R MCKINNEL
School of Informatics and Multimedia Technology
University of North London
Holloway Road, London, N78DB, UK
*Alenia Marconi Systems
FIFE KY11 8JX, UK
**Department of Computing and Electrical Engineering
Heriot-Watt University
EDINBURGH EH14 4AS, UK

Abstract: - Industrial process plants present complex problems for operators to deal with. Usually an operator can rely on standard operating procedures to deal with most situations; however, occasionally very complex and unfamiliar situations arise in which an operator needs additional advice. The objective of the EXTRAS project is to provide advice through an intelligent explanation system. Additionally, training is an important component of an operators work so the EXTRAS project also examines intelligent training. In this paper an environment for model based training is presented, which incorporates an intelligent explanation capability. This architecture uses multiple domain models to support a dedicated explanation expert. The underlying methodology for the explanation system, and a specific industrial application are described.

Key-Words: - Knowledge-based systems, Decision support systems, Models, Explanations. Proc.pp.2171-2176

1 Introduction

The use of simulation based training in industrial contexts is widely evident. There is a need, however, to supplement simulations with executable models of the industrial plant. An earlier project, MOBIT [9][11] introduced the idea of using multiple sets of domain models to provide additional support in industrial training. Its main contribution was to demonstrate how several models of the system can be used in conjunction with models of expertise and a simulation of the system to support operators. However, one limitation that was evident is that the systems being used by operators tend to be complex and difficult to master.

A well known problem with simulations is the amount of data being given out, which is usually too much for a trainee operator to manage. Therefore, the original model based architecture of MOBIT has been extended to include a dedicated explanation component whose function is to mediate between the operator and the simulation. The type of mediation depends on the circumstances of the situation. An inexperienced trainee might choose to use the fully mediated course by continually using the EXTRAS explanation component whereas a more skilled operator might work directly with the simulation or even directly with the task. It is a required function of the training environment to support all different training modes.

The following section examines the requirement for explanations made by a specific industrial application: power generation.

2 The EXTRAS Project

The EXTRAS project is a collaboration of industrial utilities and Research and Development establishments from across Europe and Heriot-Watt University in the UK, whose aim is to construct a toolkit for developing training and decision support systems for industrial process control. The focus is on producing explanations by combining model based reasoning, simulation and knowledge-based systems in a hybrid architecture. This signifies a significant departure from existing methods of decision support.

3 Methodology

The EXTRAS methodology for building applications is based on several ideas ranging from classifications of explanations, classifications of tasks and classifications of errors. There is also a specification of knowledge types. Collectively these theoretical constructs allow any situation to be specified in terms of the information needs of the user so that a tailored and focused response can be made.

3.1 Primitive Tasks

The classification of tasks reported [7] as part of the ESPRIT Project P820 provides a domain independent description of tasks that an EXTRAS application must support. A classification of five tasks is used, which covers the situations a worker (e.g. a production plant operator) is likely to encounter within the domain of industrial process control: interpretation—*determine the current state by selecting parameters and constraints*, identification—*determine possible past and contributing events*; prediction—*hypothesize about future states*; decision making—*establish all desirable future states (goals) and identify all relevant actions to achieve goals*; and execution—*perform all actions in plan*.

An analysis of tasks is essential in the methodology because it provides the guidelines for identifying types of knowledge that must be supported by the system, and for relating this to the models that need to be developed in the multiple models domain system. The system decides what coordinate of knowledge is required for a given explanation and tries to find the appropriate knowledge within that locale. Several *dimensions* of knowledge are used.

3.2 Modeling Dimensions

The modeling dimensions introduced in [8] and detailed in the methodology are generality, scope and precision. Any piece of knowledge within the system is deemed to exist at a given coordinate within these dimensions. The idea of the dimensions is that they can be used by an explanation system to navigate through the knowledge available in a controlled way.

3.2.1 Dimension of Generality

Generality is an enumerated scale relating to how general the type of knowledge in question is. In EXTRAS three levels of generality have been defined (1) Equations or principles, which contain theoretical and abstract knowledge. (2) Associations, which are relationships between a condition and a result. Usually represented by rules of an If–Then structure. (3) Procedures define how to achieve a given goal via a precise set of actions. Various action types can be used to produce very complex execution plans.

For the CCPPA application, mainly procedures and association are used. Procedures are used to encode the actions expected of an operator to achieve a given task including preconditions and postconditions. Associations are used to encode mainly alarm handling information. Here a given alarm is associated with a procedure for dealing with it. Stored procedures for alarms allow sharing of knowledge between operators so the company can share corporate knowledge.

3.2.2 Dimension of Scope

Scope represents a belonging to style relationship. In a plant this resolves to knowledge belonging to plant level, subsystem level or component level. Scope is hierarchical and one scope is quite often part of another, and scopes also often intersect. For these reasons, scope is defined much like set notation, where each scope is composed of other scopes and components.

Within the CCPPA, scope is used to define to which areas a particular piece of knowledge applies. For procedures, these are always associated with the plant, a subsystem and sometimes a component. Usually, component scope is used only at the procedure action level. The associations used tend to be at the subsystem or component level. Finally, the model used by CCPPA is at the plant scope. All models have an associated scope. The model for CCPPA is a full scope model whose simulation is performed by the full scope simulator. Within that model are the variables available through the interface to simulators, the procedures that apply to that model and the associations that apply to the model. Other models of the plant that could be used to answer questions like “How?” And “What If?” would exist at other scopes.

3.3 Request types

An analysis of user requirements has led to a list of question types that the explanation system should be able to cope with. The question types can be classified into four main classes: system states, operator actions, models, and results of reasoning as follows. Additionally there is a fifth class of ‘documentation’ which contains static information that is not task dependent.

3.3.1 System States

These include the qualitative evaluation or quantitative value of a parameter, component or system; and can be any of past, present, and future states. E.g. tell me what will happen if I do this; tell me about past malfunctions.

3.3.2 Operator Actions

These include mental and physical operational and planning actions, and can be any of: actions that were, could be or should be performed on the plant to control it (operational), as well as actions intended to reason out a solution prior to executing any controls on the plant, i.e., planning, reasoning.E.g. Tell me how past malfunctions were remedied; Tell me what I should do now.

3.3.3 System and Expertise Models

These include any elements of knowledge stored in databases and knowledge bases, and can be any of procedures, associations, and equations. Whereas requests of the first two types are the results of applying a knowledge representation, this request type seeks the knowledge representation itself. E.g. How do I operate it? Give me an operational procedure; Can you tell me all the rules of thumb associated with it; Tell me all procedures that this piece of equipment is used in; What is its general function? How does it work in this procedure?

3.3.4 Reasoning

This involves executing one or more of the available models, and can be any of simulating with mathematical equations, chaining rule-based experts, and running a sequence of actions on the plant (procedure). E.g. I can't get it to work at this stage of my procedure, what am I doing wrong? Why does it react like this in this scenario?

3.3.5 Documentation

This contains frequently accessed information and static knowledge, such as reference material. E.g. Where is it located on the plant? What does it look like?

4 Background to the application

The rest of the paper describes the CCPPA application developed using the EXTRAS methodology and toolkit. The application has primarily been designed to act as a proving ground for the EXTRAS generic explanation software components. In particular generic solutions have been produced for managing the storage, retrieval and communication of domain knowledge, editing and storage of procedures and alarms and inter application communication between multiple processes on heterogeneous hardware networks.

5 CCPPA

The Combined Cycle Power Generation Application (CCPGA) aims to build an Intelligent Simulation System (ISS) to be used for support of operators. The CCPPA is aimed at improving the off-line training of control room operators, which is currently provided by means of real time simulators. Simulators in this case replace the actual plant.

The simulator used by the application is a full scope real time simulator supplied by AMS to National Power. This provides an almost exact replica of the plant and consists of a number of operator stations and plant monitoring stations that run on a computer network. Almost every action that can be performed on the actual plant can be performed in the same way on the simulator. The simulator has been extended to allow the generic EXTRAS tools to inter-operate with the simulator: in particular setting/getting of information, notification of user actions & alarms and changing of mimic pages.

Within a training session the trainee is free to operate the simulator as if using the plant. The trainee will have a predefined task set in the form of a manoeuvre. A manoeuvre is defined as a network of procedures required to achieve a particular goal. The procedures that make up the manoeuvre are those required to achieve certain key tasks in the manoeuvre. Procedures are often carried out in parallel for complicated manoeuvres. As well as procedures, the manoeuvre also contains a series of alarms. These alarms are a selection of the total number of alarms available on the plant. When an alarm arises during the training, the trainee will be expected to follow the correct procedure for dealing with it. The alarm diagnosis information offers an opportunity for sharing information with other operators on the plant. EXTRAS allows operators to record their solutions to particular alarms; this allows less experienced operators to gain an insight into unfamiliar situations.

5.1 The Domain System

The domain system is the responsible for storing all knowledge used by the application. It is an object orientated database and access suite that allows applications to create, fetch, edit and store knowledge in an object orientated fashion.

Knowledge is stored via servers and can be stored and accessed from anywhere in the network. As well as storing static information, the domain is also used for representing transient knowledge. Transient knowledge is knowledge that exists for a short time to serve a purpose and then expires. In the application, transient knowledge is used primarily for communicating information between applications. Transient knowledge can contain links to static knowledge that makes communication of information very powerful. For example, an error message can be passed as transient information but contain links to static information such as a stored procedure or action that the error was about. The domain system is integrated to all components of the EXTRAS toolkit.

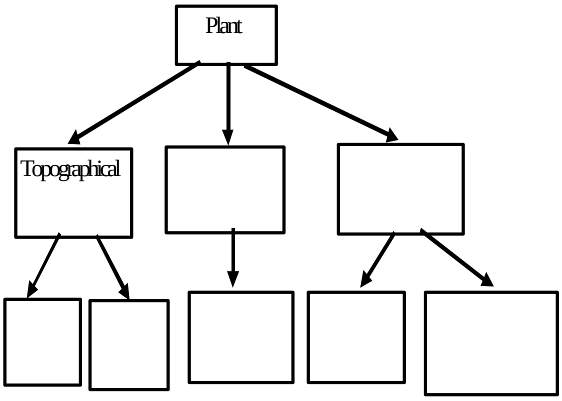

Fig. 1: Arrangement of Knowledge in Application

The domain knowledge has been represented as in Fig. 1. Examples of procedures and associations knowledge are given below. Procedures represent instructions that operators must perform to control the system, and the particular kind of associations described here are for alarm handling. A procedure is described below.

Step 1: In the MV/LV Switchroom CHECK with CCR that the Steam Turbine 415V Distribution Board JESS1E is Live, then CHECK locally that the CW Make-up Modulating Valve Cubicle (19PAR50GW001) supply conductors are isolated and racked in.
Step 2: At the Steam Turbine House Condenser Sub-basement CHECK that the Condenser waterboxes have been vacated, and all tools and lights have been removed.
Step 3: CLOSE and tighten down securely all waterbox access doors.
Step 4: CHECK that the CONDENSER INLET ISOLATING VALVES (84PAB10(20)AA001) (2 off) are OPEN.
Step 5: CHECK that the CONDENSER OUTLET ISOLATION VALVES (84PAB30(40)AA001) (2 off) are OPEN.

There are many more steps to a procedure and at least nine hundred procedures associated with the plant. Each procedure can have up to 200 steps. The natural language representation of procedures is parsed to create an action that can be monitored by the EXTRAS system. Monitoring is performed by comparing every action performed by the user with a step in a procedure [1].

An example of an alarm association is given below. The alarms are encoded into the domain system as associations so that when the alarm is detected the response can be immediately displayed to the operator:

Code: **GEN CB AIR SUPPLY**
FAULT: Common alarm for air supply to generator
Actions:
1. Check air supplies for correct pressure.
2. Circuit breaker and busbar. Check SF6 isolating gas for correct pressure.
3. Compressor for generator circuit breaker fault If pressure is too low, refill it.

Generator circuit breaker run time exceeded Check indications at indication panel.

5.1.1 Domain Knowledge Encoding

Within CCPPA, all domain knowledge is encoded within the AMS domain system. The domain system is an object-orientated database and has been used to implement the proposed structure of knowledge outlined in [10]. To do this database class definitions have been created for each knowledge type and container defined. In CCPPA, all knowledge is encoded using the domain system. This includes static domain knowledge for permanent information and transient information in the form of messages and information sent between the CCPPA application's programs. As far as static knowledge is concerned, mainly procedures and associations are encoded along with the necessary model container and index mechanisms defined in [10]. No equations are used at present. The real use for equations is seen to be in the other simulation models that could be created for preparing higher level explanations for the user. The static knowledge is encoded using a number of authoring tools, created specifically for the job.

5.2 The Messaging Layer

The AMS toolkit requires multiple processes on multiple computers to be able to communicate. The messaging layer provides a framework whereby any application can communicate with any other seamlessly. The messaging layer supports sending of byte streams between applications written in both Java and C++, running on Windows NT/95/98 or UNIX.

5.3 The Application Layer

All of the components in the AMS demonstrator are written using the application layer. It provides a high level interface that allows an application program to send and receive messages to and from others and to provide services for others to use. The key feature of the application layer is that all messages and services use transient domain objects to hold the message information, the parameters to services and the results from services. This is described in more detail later.

5.4 Alarm Editors

To facilitate the editing of some of the domain knowledge used by the application a number of editors have been produced. These editors have been written in Java and are thus able to run on any platform that the domain and application layer is ported to. Extensive use has been made of HTML for action descriptions and supporting information to allow links to other information to be easily added. For example, links to pages on the simulator, links to supporting information etc. Java is used so that some of the key presentation components used in the editors can be reused to provide explanation through embedded applets in the presentation system. This also means that all of the support links such as expert action demonstrations embedded in the knowledge become instantly available to the trainee.

5.5 Interface to Simulators

The interface to the simulator is separated into multiple parts. For obtaining user actions, alarm acknowledgements, alarms and requesting page changes, the application layer is used with predefined message types. The simulator automatically sends user actions etc. to anyone who is interested. To pick these up, an EXTRAS application simply has to register an interest. The actual getting and setting of information is performed using application layer services and the messaging layer. Services are used to order sequences of information to be prepared by the simulator. Control of the simulator and all EXTRAS applications is through a standard set of services they all provide.

5.6 Monitor User Actions

This application is used during a training session to monitor what the operator is doing with regards to actions on the control system or other systems simulated. These are then compared with what is expected for the manoeuvre being executed. Any discrepancies detected are sent out as messages in the application layer to any application that has registered an interest in discrepancies. For any discrepancy, the associated error is also classified.

5.7 Explanation and Presentation System

The explanation and presentation systems are combined into a single application. This application takes any errors generated within the application and presents an appropriate explanation. Explanations are generated in HTML and use plug in applets reused from the knowledge editors to present the more complex information. This allows follow up explanations and questions to be easily presented via HTML links of differing types. The approach used makes it possible for multiple explanation systems to be running in a network, which could be useful for training multiple operators on multiple stations.

6 Explanations

The explanation system is responsible for producing all explanations for the trainee. The explanation system takes inputs from monitor user action and other sources. When a situation arises that the system feels requires an explanation, it might or might not pass this on to the trainee. It may instead simply generate an explanation and store it to a log for viewing after the exercise is finished. In the current implementation, explanations are presented to the trainee via an HTML browser with imbedded applets for more complex explanations. Depending on the mode of operation, the explanation system will inform the trainee of all violations, only the more serious violations or only severe deviations from the planned exercise. In all cases, it will be possible for the explanations system to shut down the exercise. This is most likely only in severe cases, where the errors are too bad to warrant continuing the randomness of the trainee inputs.

The actual inputs to the explainer are mostly user action errors. These are deviations detected by Monitor User Actions from the planned actions encoded in the manoeuvre's procedures. The error types generated are as based on [2] and [3].

6.1 Explanation Generation

The explanation system is mostly reactive. That is it reacts to conditions and decides whether to present the information to the trainee. Some proactive explanation is provided to the trainee at most times through links on the explanation pages allowing them to find out about actions, procedures, associations and the manoeuvre they are working on. All explanations are logged as HTML files and an index HTML file is created to allow review. The decision about whether the feedback is proactive or reactive is configured depending on the experience of the user and can be configured by the trainee during the training session.

Explanations are generated from the methodology proposed in [4]. In this the error coming from the monitor or request from the user is converted to a particular type of task as outlined earlier. The decision making task, for example, relates to the act of trying to make a choice on how to achieve a given goal. There may be alternatives to this. In CCPPA, the trainee will have a very precise definition of the current goal they have to achieve. They can ask for an explanation of this goal and a demonstration via the explanation interface. For the application, there is almost never an alternative action recommended for any given situation. The course of action is always very precise. Association are used for alarms and can be used to determine how to deal with a given alarm situation. The Execution task is related to an attempt to perform a well-defined task. This is the primary task type associated with the CCPPA. CCPPA deals mainly with procedures and associations encoded in the manoeuvres. The explanations are generated using the fillers approach of [5] in which core information is mapped into templates.

7 Conclusion

A methodology for designing process control support systems and an implementation was described. The methodology contains classifications of situations that allow an explanation expert to select an appropriate kind of response. An architecture for a development toolkit was also described with a specific implementation of a power plant application.

References:

[1] Caimi, M and Lanza, C. Intelligent Simulation Systems: the EXTRAS Approach. *Proceedings of the Sixth European Conference on Cognitive Science Approaches to Process Control,* 1997, pp.103-106, CNR.

[2] Hill R W and Johnson, W L. Situated Plan Attribution, *Journal AI in Education*, 1995.

[3] Hollnagel E. The Phenotype of Erroneous Actions, *Human-Computer Interaction and Complex Systems*, Weir G R S and Alty J L (eds), Academic Press, 73-121. 1991.

[4] Khan T M, Mitchell J E M, Ravindranathan M D, Brown K E and Leitch R R. Methodology for Generation of Dynamic Process Explanations, *EXTRAS project BE-95-1245, Deliverable Report D4*, 1997.

[5] Khan T M, Brown K E and Leitch R R. Didactic and Informational Explanation in Simulations with Multiple Models, *AIED97, 1997,* pp.355-363.

[6] Khan, T.M., Paul, S.J., Brown, K.E., Leitch, R.R. Model based explanations in simulation based training. Proceedings of Intelligent. *4th International Conference on Tutoring Systems, ITS'98. San Antono, Texas,* 1998.

[7] Leitch R and Stefanini A. QUIC: a development environment for knowledge based systems in industrial automation, *ESPRIT 88,* vol. 1, 1988, 674-696.

[8] Leitch R R et al. Modeling Choices in Intelligent Systems, *AISB Quarterly* No 93, 1995, 54-60.

[9] Liddle, J., Leitch, R. and Brown.K. Generic Approaches to Developing Practical Intelligent Industrial Training Systems, Proceedings of Third International Conference, ITS '96, Montreal, Canada, 1996, pp. 512-520.

[10] McKinnel R. The EXTRAS Domain Knowledge Structure for Multiple Models and Explanation, *Project BE-95-1245, IR2-11*, 1998.

[11] Mitchell, J.E.M, Liddle, J., Brown, K. and Leitch, R.R. Integrating Simulations into Intelligent Training Systems, *EuroAI-ed '96*, Lisbon, 1996, pp. 80-87.

[12] Norman, D. Things that make us smart, 1993.