Developing Industrial Multi-Agent Systems†

N. R. Jennings
Dept. Electronic Engineering,
Queen Mary & Westfield College,
Mile End Road, London E1 4NS, UK.
N.R.Jennings@qmw.ac.uk

J. M. Corera
IBERDROLA S. A.
c/ Gardoqui 8,
48008 Bilbao, Spain.
jose.corera@iberdrola.es

I. Laresgoiti
LABEIN,
Parque Technologico de Zamudio 101
48016 Zamudio, Spain.
lares@labein.es

(INVITED PAPER)

Abstract

The development and deployment of multi-agent systems in real world settings raises a number of important research issues and problems which must be overcome if Distributed AI (DAI) is to become a widespread solution technology. Work undertaken in the context of the ARCHON project has provided a number of important insights into these issues. By providing an in depth analysis of ARCHON's electricity transportation management application, this paper draws together many of the experiences obtained when building one of the world's first operational DAI systems.

Introduction

In many industrial applications a substantial amount of time, effort and finance has been devoted to developing complex and sophisticated software systems. These systems are often viewed in a piecemeal manner as isolated islands of automation, when, in reality, they should be seen as components of a much larger business function (Jennings, 1994a). The main benefit of taking a holistic perspective is that the partial subsystems can be integrated into a coherent and consistent super-system in which they work together to better meet the needs of the entire application. By the very fact that they are integrated, the finite budgets available for information technology development can be made to go further - consistent and up-to-date versions of the data can be shared by all the problem solvers, basic functionalities need only be implemented in one place, problem solving can make use of timely information which might not otherwise be available, and so on.

Two components are required to develop a well-structured DAI system: a software framework which provides assistance for interaction between the constituent subcomponents and a design methodology which provides a means of structuring these interactions. ARCHON addresses both of these facets: providing a decentralised software platform which offers the necessary control and level of integration to help the subcomponents to work together and devising a concomitant methodology which offers guidance on how to decompose the overall application and how to distribute the constituent tasks throughout the community to make best use of the capabilities of the ARCHON framework. Both of these facets have been applied to a number of real world industrial applications (Jennings, 1994b) - however here the electricity transportation management application developed and run on-line at Iberdrola[1] is the main focus.

ARCHON's individual problem solving entities are called *agents*; these agents have the ability to control their own problem solving and to interact with other community members. The interactions typically involve agents cooperating and communicating with one another in order to enhance their individual problem solving and to better solve the overall application problem. Each agent consists of an *ARCHON Layer* (AL) and an application program (known as an *Intelligent System* (IS)). Purpose-built ISs can make use of the ARCHON functionality to enhance their problem solving and to improve their robustness. However pre-existing ISs can also be incorporated, with a little adaptation, and can experience similar benefits (Jennings *et al.*, 1993). This latter point is important because in many cases developing the entire application afresh would be considered too expensive or too large a change away from proven technology (Jennings and Wittig, 1992).

To successfully incorporate both purpose-built and pre-existing systems, community design must be carried out from two different perspectives simultaneously. A top down approach is needed to look at the overall needs of the application and a bottom up approach is needed to look at the capabilities of the existing systems. Once the gap between what is required and what is available has been identified, the system designer can choose to provide the additional functionality through new systems, through additions to the existing systems, or through the ARCHON software itself. This methodology, which is described more thoroughly in Varga *et al*. (1994), shapes the design process by providing guidelines for problem decomposition and distribution which reduce inefficiencies.

This paper is organised along the following lines: section two provides an overview of the ARCHON architecture. Section three describes the Iberdrola application - it involves seven heterogeneous agents, a substantial number of which were purpose built, which perform three main types of activity (data acquisition, fault diagnosis, and service restoration) and cooperate in a number of different styles. Finally, section four highlights some key experiences which impact upon the design and implementation of future DAI systems.

The Archon Framework

The ARCHON software has been used to integrate a wide variety of application program types under the general assumption that the ensuing agents will be loosely coupled and semi-autonomous. The ISs themselves can be heterogeneous - in terms of their programming language, their

† The work described in this paper was supported by the ESPRIT II project P2256 (ARCHON)

1. Iberdrola is a large Spanish electric utility. Their transport network, on which this application operates, is controlled by the North Dispatch Control Room (DCR) located in Bilbao.

algorithm, their problem solving paradigm, and their hardware platform - as their differences are masked by a standard AL-IS interface. An AL views its IS in a purely functional manner, it expects to invoke functions (*tasks*) which return results, and there is a fixed language (Cockburn and Jennings, 1995) for managing this interaction.

In an ARCHON community there is no centrally located global authority, each agent controls its own IS and mediates its own interactions with other agents (*acquaintances*). The system's overall objectives are expressed in the separate local goals of each community member. Because the agents' goals are often interrelated, social interactions are required to meet global constraints and to provide the necessary services and information. Such interactions are controlled by the agent's AL.

In more detail, an agent's AL needs to: control tasks within its local IS (monitor), decide when to interact with other agents (planning and coordination module) (for which it needs to model the capabilities of its own IS and the ISs of the other agents - agent information management (AIM) module), and communicate with its acquaintances (high level communication module) (fig. 1).

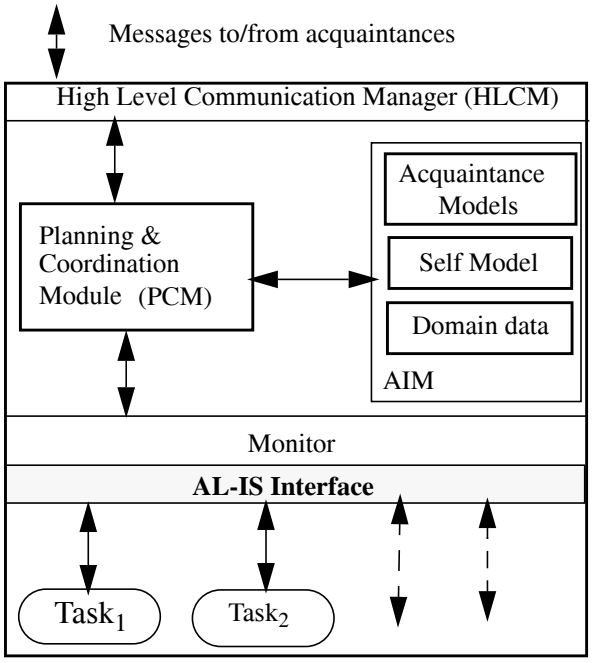

Figure 1: ARCHON Agent Architecture

Monitor

The Monitor is responsible for controlling the local IS. Each IS task is represented in the Monitor by a **monitoring unit** (MU). MUs present a standard interface to the Monitor whatever the host programming language and hardware platform of the underlying IS. Figure 2 shows a graphical representation of a MU called `UpdateTopologyWithAlarms` which takes `ALARM-MESSAGES` and `DISTURBANCE-ID` as inputs and produces `UPDATED-TOPOLOGY` as an output. The IS task associated with this MU is called `UpdateTopologyWithAlarms` in the AL and `topology_update` in the IS.

MUs can send and receive messages (directives, confirmations and requests) to and from the IS. All messages have to pass through the AL-IS interface which performs the translation and interpretation required for the IS to

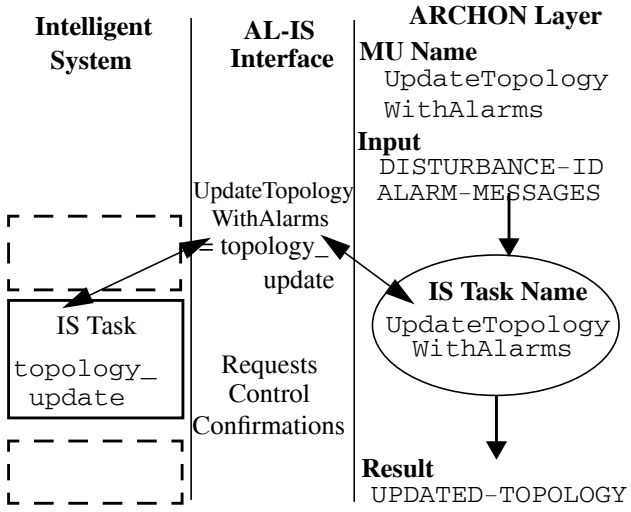

Figure 2: UpdateTopologyWithAlarms Monitoring Unit

understand the AL directives and for the AL to understand the IS messages.

MUs represent the finest level of control in the AL, at the next level of granularity there are **plans**. Plans are pre-specified acyclic OR-graphs in which the nodes are MUs and the arcs are conditions. These conditions can: be dependant on data already available from previously executed MUs in the plan, be dependant on data input to the plan when it started, make use of the locking mechanism for critical sections of the plan, or be used to return intermediate results before a plan has completed.

A sample plan which starts the fault diagnosis activity is shown in figure 3. Firstly, `ALARM-MESSAGES` are used as an input to the `SetNewFault` MU which notes that there is a new fault in the network and generates a new identifier (`DISTURBANCE-ID`) for it - this identifier is returned as one of the plan's intermediate results. The alarm messages and the disturbance identifier are then used as inputs to the previously described `UpdateTopologyWithAlarms` MU. When this MU is complete, the model of the network on which the diagnosis will be based is up to date and hence the process of identifying a potential list of faults can commence. There are two ways this activity can proceed: firstly, it is checked whether a list of generated hypotheses has already been provided (and stored in the domain data component of AIM) by an agent called BRS in which case these should form the start point (the `HypothesisGeneration-FromForeignSource` MU should be executed). If no pertinent information is available then the list should be generated from scratch (the `HypothesisGeneration` MU should be executed). In the latter case, the plan returns the list of generated hypotheses as an intermediate result so that they can be used elsewhere within the agent or even disseminated to relevant acquaintances.

The plan mechanism has an inbuilt backtracking facility which can be used to express preferences and deal with complex alternatives. Consider the plan `ReceiveAlarms` (figure 4) which determines what course of action an agent should take when it receives alarm messages. Here there are three cases: (i) see whether the alarms correspond to an ongoing fault which is known about; (ii) see whether the messages have been generated by planned maintenance (manœuvres) on the network; or (iii) see whether the alarms correspond to a new fault. The Monitor first tries the leftmost

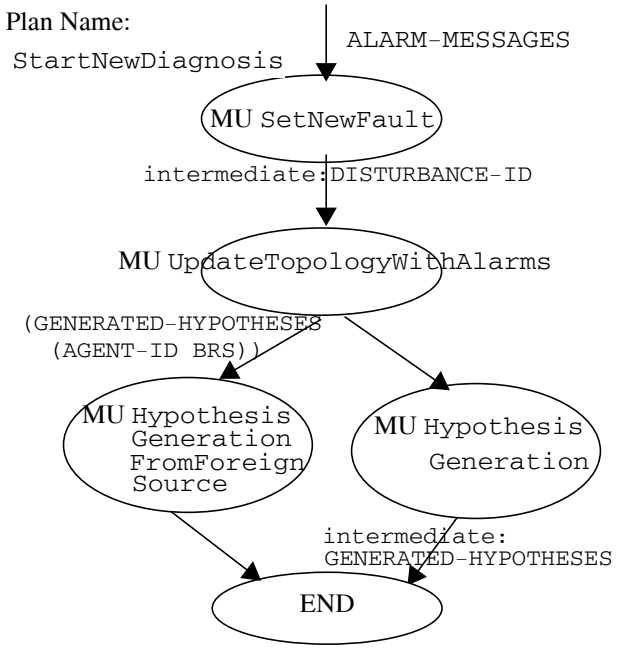

Figure 3: StartNewDiagnosis Plan

branch and executes the OngoingFault MU - if this is unable to give a disturbance identifier for the alarms then they cannot correspond to a known fault and so this branch fails; if, on the other hand, an identifier is found then the rest of the branch is traversed. In the case of failure, the plan mechanism backtracks up to the last successful execution (MU CollectAlarms) and tries the next branch (the Manoeuvres MU) - this branch fails if the disturbance identifier associated with the alarms is not given the tag "MANOEUVRES". Finally, if the alarms are not generated by manœuvres then they correspond to a new fault and so the StartNewDiagnosis branch, as described in figure 3, is invoked (this branch never fails and so the ReceiveAlarms plan will successfully terminate when the plan has been completed).

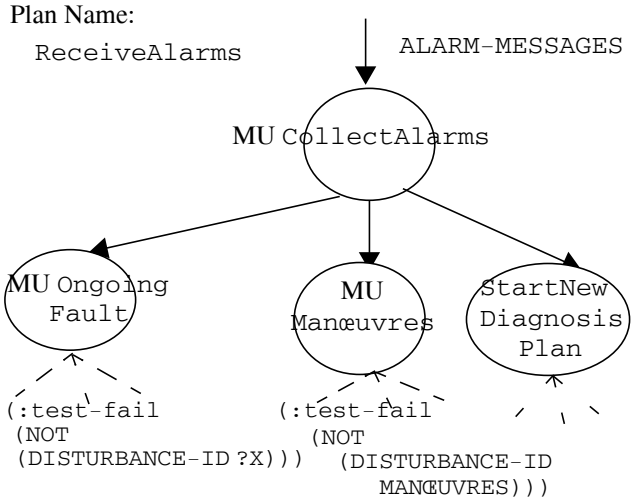

Figure 4: ReceiveAlarms Plan (with backtracking)

The highest level at which the IS's activities are represented is the behaviour level. **Behaviours** contain a plan, a trigger condition for activating the behaviour, descriptions of the inputs needed by the activity and the results which will be produced, and any children of the behaviour. There are two types of behaviour: those that are visible to the PCM (and the other AL components) and those that are purely internal to the Monitor. The former type are called **skills** and they may be triggered by new data (either arriving from other agents or which the agent has generated itself) or by direct requests from other agents.

Planning and Coordination Module (PCM)

The PCM is the reflective part of the AL, reasoning about the agent's role in terms of the wider cooperating community (Jennings and Pople, 1993). This module has to assess the agent's current status and decide which actions should be taken in order to exploit interactions with others whilst ensuring that the agent contributes to the community's overall well being. Specific examples of the PCM's functionality include: deciding which skills should be executed locally and which should be delegated to others, directing requests for cooperation to appropriate agents, determining how to respond to requests from other agents, and identifying when to disseminate timely information to acquaintances who would benefit from receiving it.

The PCM is composed of generic rules about cooperation and situation assessment which are applicable in all industrial applications - all the domain specific information needed to define individual behaviour is stored in the self and acquaintance models. The former contains information about the local IS and the latter contain information about the other agents in the system with which the modelling agent will interact. For example, in order to determine how to obtain information which is needed to execute a behaviour but which is not currently available, the PCM will make reference to its self model to see if the information can be provided locally by executing an appropriate skill. If the information cannot be provided locally then the acquaintance models are checked to see if another community member can provide it. The final major role of the PCM is to deal with requests arriving from other agents. By reference to its self model, it will decide whether to honour the request and will then activate the necessary skill to provide the requested data; when the information is available it will ensure that a reply is directed to the source of the request.

Agent Information Management Module

The AIM module is a distributed object management system which was designed to provide information management services to cooperating agents (Tuijnman and Afsarmanesh, 1993). Within ARCHON, it is used to store both the agent models and the domain level data.

As an illustration of the agent models, consider an agent which is capable of producing information about ALARM-MESSAGES. The interest slots of its acquaintance models contain those agents who are interested in receiving this information and the conditions under which they are interested. The following portion of the acquaintance model specifies that an agent called BRS is interested in ALARM-MESSAGES which contain chronological information, that an agent called AAA is interested only in non-chronological alarm messages, and that an agent called BAI is only interested in non-chronological alarm messages which have the string INT within their ALARMS field:

INTEREST-DESCRIPTOR
INFORMATION-NAME: ALARM-MESSAGES

```
INFORMATION-CONDITION:
[   ("BRS", (CONTAIN (ALARM-MESSAGES "CHRONO "YES"")));
    ("AAA", (CONTAIN (ALARM-MESSAGES "CHRONO "NO"")));
    ("BAI", (AND  (CONTAIN (ALARM-MESSAGES "CHRONO "NO""))
                  (CONTAIN (ALARM-MESSAGES.ALARMS "INT")));]
```

In many industrial applications the domain level data which the agents need to exchange has a complex internal structure. In ARCHON, this structure is specified and maintained by AIM. For example, the information type `ALARM-MESSAGES` is defined in the following manner:

```
ALARM-MESSAGES
    AGENT-ID: AGENT              DISTURBANCE-ID: SOURCE
    CHRONO: Y-N-FLAG             BLOCK-TYPE: BLOCK-TYPE
    BLOCK-ID: ID-TYPE            ALARMS: LIST-OF-ALARMS
```

where each of the following types has the following set or permissable values: agent (CSI | AAA), source (yymmddhhmmss | MANOEUVRES | UNKNOWN), y-n-flag (YES | NO), block-type (UNIQUE | UNKNOWN | MIXED), id-type (yymmddhhmmss), and list-of-alarms (alarm1, ... alarmn).

High Level Communication Module

The High Level Communication Module (HLCM) allows agents to communicate with one another using services based on the TCP/IP protocol. The HLCM incorporates the functionality of the ISO/OSI Session Layer which continuously checks communication links and provides automatic recovery of connection breaks when possible.

Electricity Transport Management

Energy management is the process of monitoring and controlling the cycle of generating, transporting and distributing electrical energy to industrial and domestic customers. Generation transforms raw energy into a more accessible form. Energy is then *transported* from its generation site to the consumer. To minimise losses during transportation, the electrical voltage is made high (132 kV or above) before it is placed on a *transport network* and sent over many hundreds of kilometres. Finally, the voltage is lowered and electricity is delivered to the consumers using a *distribution network*.

To ensure the transportation network remains within the desired safety and economical constraints, it is equipped with a sophisticated data acquisition system (SCADA) and several conventional application programs which help the operator (*control engineer*) to analyse it (these programs are primarily designed for normal operating conditions). The network's operation is monitored from a DCR and whenever an unexpected event occurs hundreds of alarms are automatically sent to it by the SCADA system. Under these circumstances the operator has to rely on experiential knowledge to analyse the information, diagnose the situation, and take appropriate remedial actions to return the network to a safe state. To reduce the operators' cognitive load in such circumstances, and to help them make better decisions faster, Iberdrola decided that a number of decision support systems should be developed. These systems were then interconnected and subsequently extended using ARCHON technology - for a more detailed analysis refer to Jennings *et al.* (1995).

Why use DAI techniques?

When Iberdrola decided, in 1988, to implement decision support tools to ease the workload of their control engineers during disturbances, several technical factors affected their design choices. Firstly, the control system itself was a proprietary product from a control systems supply company - thus it was considered too risky and too difficult to embed the additional functionality directly within it. Secondly, the state of the art for commercial systems in this domain meant that the diverse support functions could only be realised through a number of standalone systems. Consequently, Iberdrola built separate decision support systems to assist with different aspects of the control engineer's job - the one which is most relevant to the subsequent discussion is the alarms analysis expert system which diagnosed faults produced in the network based on the alarm messages which arrived at the DCR. Their decision support systems were unconnected apart from the fact that they retrieved information about the network from the same source (the control system's real time database). To make this information available to the non-proprietary software products a number of interfaces to the control system had to be written - as well as providing access, these interfaces could filter and pre-process network information.

By 1991, however, some important changes had occurred: (i) the evolution of IT hardware and software had significantly increased the quantity and quality of the data which could be acquired from the transport network; (ii) distributed computing had become commercially viable because of improvements in local area network technology; and (iii) the prices of computers had decreased significantly so that powerful machines were no longer prohibitively expensive. Taken together, these changes meant that better and more powerful tools could be built to assist the control engineer. In particular it was considered important to be able to actually perform and dynamically monitor the service restoration process and also to exploit the new data sources, such as chronological information or faster rate snapshots, which became available. However the tried and tested decision support tools were still needed. Thus it was decided to adopt a system upgrading strategy which enabled the previously operational components to be used in conjunction with the new functionality. Two means of realising this strategy were considered: extend the existing systems to cover the new features or follow a distributed approach and allow the new functionality to be expressed as distinct computational entities which could interact with the pre-existing systems through a common distribution platform. The second option was chosen because it was considered to be the most effective means of:

(i) Permitting reasoning based on information of different granularity. Two types of alarm, chronological and non-chronological, now needed to be dealt with. In non-chronological alarms, the time stamped is coincident with the time of acquisition by the control system, whereas in chronological alarms the time stamped is coincident with the actual occurrence of the event. As chronological alarms represent a more accurate picture of events in the network they generally lead to a swifter diagnosis, however they have the disadvantage that chronological information has a low priority in Iberdrola's communication channels. Thus when the channels are saturated (as can happen during a disturbance) their availability time is unpredictable. For these reasons it was decided to build a new alarm analysis expert system which utilised chronological information and could subsequently integrate its results with those of the pre-existing system, rather than construct a monolithic system which received both types of data and had to embody both types of diagnostic knowledge. A similar situation occurs when considering service restoration. Two types of information are relevant to this activity: *snapshots* (which provide a comprehensive pic-

ture of the current state of all the components in the network) and *alarm messages* (which show how the state of the components has changed over a period of time). The former can be produced relatively quickly and give a complete picture of the system's state, whereas the latter may take several minutes for a large disturbance but are needed to indicate the type of fault from which the system must be restored. Rather than trying to place both types of information and reasoning in a single system it seemed more natural to develop a service restoration subsystem which dealt mainly with snapshots and received the necessary high-level information about the equipment at fault from a diagnosis subsystem (rather than trying to deal with the raw alarm messages itself).

(ii) Allowing different network models to be included within the same system. Some of the problem solvers needed to work on the SCADA model of the network, while others needed the applications network model (a model which permits network equations to be solved and takes the physical characteristics of all its components into account). Rather than trying to combine and harmonise these complex and disparate models at design time, it was decided that each subsystem should work on whichever model was most appropriate for its task. Then the various components should be able to interact at runtime to resolve any inconsistencies which arise from their use of different network models.

(iii) Enabling a number of different problem solving paradigms to be utilised. The diverse range of activities which needed to be performed in this application meant that there was no universally best problem solving paradigm: procedural techniques were required for algorithmic calculations like connectivity (to know which component is connected to which other) and load-flow analysis (solution of the network equations), whereas symbolic reasoning based on heuristic search was the best approach to diagnosis. A distributed approach enabled each component to be encoded in the most appropriate method.

(iv) Meeting the application's performance criteria. Transportation management is a time-critical application and as many different types of information can be processed in parallel, with only a small synchronisation overhead, the response time of the overall system can be improved through the use of a number of interconnected machines.

Having decided upon a distributed approach, a choice had to be made between using more conventional distributed processing techniques or DAI techniques - here the latter was adopted for the following reasons: (Barandiaran *et al.*, 1991; Abel *et al.*, 1993)

(i) Economy: The alarms analysis expert system was already operational, however new functionality needed to be added and new information needed to be treated. It was estimated that the cost of modifying the extant system was significantly larger than that of implementing a new one. However it was also judged that as the new functionalities and data were so diverse that it would be an extremely expensive activity to put them within a single system. Therefore it turned out to be more economical to build smaller systems, and re-use the existing alarm analysis expert system, and allow them to be integrated through a DAI framework. A DAI framework was needed because the interactions between these subsystems were both sophisticated and context dependent, therefore run time reasoning based on dynamic data needed to be performed.

(ii) Robustness: As the subsystems have overlapping domains of expertise, the failure of one of them to produce an answer does not necessarily mean that no solution will be forthcoming because one of the other systems may be able to produce at least a partial solution. However to achieve this back-up functionality in a flexible manner, the different problem solving components need to be intelligently coordinated - a task beyond present generation distributed processing systems.

(iii) Reliability: The solutions of the systems that overlap can be cross-referenced to enable the operator to be presented with more reliable information. Again, however, this cross-referencing functionality needs to be properly managed according to the prevailing circumstances and so requires dynamic and flexible reasoning to take place.

(iv) Natural representation of the domain: A DAI approach accurately represents the way the control engineers work when a large disturbance occurs. They specialize their roles - one looks after restoration, another tries to diagnose the problem based on different sources of information, and so on - and they then communicate relevant information to one another to ensure they are following a coherent course of action towards the overall objective of restoring the service (Jennings and Wittig, 1992).

Specification of the Agents

During normal working conditions, management of the network by the operator in the DCR consists mainly of topology changes (operation on breakers and switches), generation scheduling, and control of the energy interchange with other utilities (Corera *et al.*, 1993). However, during emergency situations management becomes considerably more difficult because of the large number of constraints which have to be taken into consideration and the insufficient quality of the information which is available to make these decisions. Emergency situations typically originate from a short circuit in a line, bus-bar or transformer. They can be exacerbated by equipment malfunctioning (eg a breaker failing to open) or subsequent overloads (a domino effect can cause one line to fail because of an overload, this in turn increases the load on neighbouring lines so they become overloaded and subsequently fail, and so on). The situation can become even worse if power stations become disconnected as this will cause an imbalance in the network's power. Consequently, actions to restore service must be taken rapidly and accurately, so that what starts as a relatively minor problem does not escalate into a major disaster. In these circumstances, the actions which the operator can perform consist mainly of breaker operations, topology changes, and activation/deactivation of automatisms and protective relays. For larger disturbances, however, actions on power plants may also be required.

From this description of the control engineer's job, a top-down analysis identified that a comprehensive decision support system should cover the following activities: (i) Detect the existence of disturbances; sometimes the operation of protective relays and breakers can be caused by routine maintenance and this should not be confused with a genuine disturbance situation; (ii) Determine the cause, location and type of the disturbance; including identifying if any equipment is permanently damaged; (iii) Analyse the situation of the network once it arrives at a steady state; and (iv) Prepare a restoration plan to return the network to its original operational state

Allying this top-down analysis with the bottom-up perspective of examining the extant systems, it was decided to encapsulate the following pre-existing systems as agents - the alarms analysis expert system and the interface to the control system. As discussed earlier, the availability of

chronological alarm messages necessitated a new diagnosis system which it was decided to make available as an agent. Finally, it was always known that information about the initial area out of service (the *black out area*) could help constrain the search of the faulty equipment, however it was never deemed cost effective to develop a dedicated stand alone system for this purpose since the original alarm analysis expert system's performance was considered satisfactory (if somewhat slow). However through the use of DAI technology much of the basic infrastructure to implement this functionality was now available from other agents and so it was considered economically viable to develop a system capable of producing this information (in terms of the ARCHON methodology, this decision corresponds to providing additional functionality through the development of new systems).

In more detail, the operational DAI system consists of seven agents running on five different machines.

• **BAI** (Black-out Area Identifier) When a fault occurs, the network's protective relays and breakers automatically try to isolate the minimum amount of equipment possible; in an ideal case only the element at fault would be isolated. The BAI's objective is to identify which elements of the network are initially out of service as the actual element at fault must be within this region. It uses non chronological alarm messages as its information source and cooperates with the BRS and the AAA to increase the efficiency of the overall diagnosis process.

• **CSI** (pre-existing Control System Interface) The CSI acts as the application's front end to the control system computers. Its objectives are to acquire and distribute network data to the other agents, to interface to the conventional management system application programs, and to monitor the restoration process to detect any unexpected deviations. It is split into two physical agents: **CSI-D** which detects the occurrence of disturbances and preprocesses the chronological and non chronological alarm messages which are used by the AAA, the BAI and the BRS; and **CSI-R** which detects and corrects inconsistencies in the snapshot data file of the network, calculates the power flowing through it and makes this information available to the SRA and the UIA. CSI-D is primarily concerned with the system's diagnosis activities and CSI-R with its restoration activities.

• **BRS** (Breakers and Relays Supervisor) The new alarms analysis expert system which detects the occurrence of a disturbance, determines the type of fault and its extent, generates an ordered list of fault hypotheses, validates hypotheses, and identifies malfunctioning equipment. In order to perform its analysis, it takes two types of inputs: chronological alarm messages and snapshots of the network which give the status of every breaker and switch.

• **AAA** (pre-existing, non chronological Alarms Analysis Agent expert system) This agent pursues similar goals to the BRS, however the quality of information it receives is inferior to that of the BRS. Although the alarm messages received by both systems relate to the same physical operations, those received by the AAA represent ±5 seconds accuracy, while those received by the BRS are precise. This means that if the data is error free, then the BRS performs a better diagnosis than the AAA. However if some of the chronological information is lost (a distinct possibility when the SCADA system is busy) then the BRS may perform worse than the AAA. Therefore whenever incomplete or erroneous information exists, which is in most interesting cases, there is a need for cooperation between the two systems to make the overall system more robust and reliable.

• **SRA** (Service Restoration Agent) This agent devises a service restoration plan to return the network to a steady state after a blackout has occurred. To do this it takes into account the constraints imposed by the damaged equipment, as identified by the diagnosis agents.

• **UIA** (User Interface Agent) This agent implements the interface between the users and the community of agents. It gives the user the facility to inspect the results produced by the diagnosis agents, display the alarms received, and browse through the log of analysed disturbances. From the point of view of restoration, the user can see the plan produced, modify it, run it in a simulated environment to see its predicted effect, and request the development of a new restoration plan which takes into account some actions which he deems pertinent. Through the use of a distributed windowing system, the UIA presents the appropriate information on the consoles of the various control engineers who are working on the system.

This system design ensures that all the tasks identified by the top-down analysis are performed by at least one agent. Robustness is achieved by having multiple agents that are able to provide the same (or at least some) overlapping results. Efficiency is obtained by the parallel activation of tasks. Reliability is increased because even if one of the agents breaks down the rest of the agents can often produce a result which, although not as good as the one provided by the complete system, is still of use to the operator.

Cooperative Scenarios

An important example of cooperation in this system involves the information interchange between the AAA, BRS and BAI agents. The AAA and the BRS produce the same result from different information sources, while the BAI applies different knowledge to produce a result that should be coherent with that of the AAA and the BRS.

Assume a block of non-chronological alarm messages has been provided by the SCADA system and these alarm messages have been identified as related to a disturbance by the CSI. Using the interest descriptors of its acquaintance models - see the AIM section - the CSI will realise that this information is relevant to the AAA and the BAI. Application of the appropriate PCM generic rule will result in this information voluntarily being sent to the specified agents as unsolicited data. Some time later, the same process will be repeated and the BRS will receive the corresponding chronological alarm messages. At this point, the AAA, BAI and BRS are all operating in parallel.

When the AAA receives the alarm messages, and the corresponding disturbance identifier marks them as being a consequence of a new fault (see figure 4), the `StartNew-Diagnosis` plan is executed and a preliminary set of hypotheses (`GENERATED-HYPOTHESES`) are produced. During this time, the BAI would also have received the alarm messages and would have started its skill for producing the `Initial-Black-Out-Area`. When this plan is complete, the BAI's PCM checks whether any other agents are interested in this information - it finds out that the AAA is and so it sends out the information.

Simultaneously, altough after a certain delay, the BRS agent starts working on the analysis of the chronological alarm messages. This will also result in a list of `GENERATED-HYPOTHESES` being produced. The BRS checks whether any agents are interested in this information - again the AAA is noted and the generated hypotheses are sent to it. The BRS then continues with its diagnosis to try and validate the cause of the fault.

After producing its tentative list of hypotheses, the AAA proceeds with a detailed analysis to try and ascertain the precise cause of the fault (i.e. to produce VALIDATED-HYPOTHESES). The following situations may then occur: (i) the Initial-Black-Out-Area is available to the AAA, this triggers a refinement behaviour which may reduce the number of hypotheses to be validated because the BAI has given a focused view of the situation; (ii) the generated hypotheses provided by the BRS are available to the AAA, this triggers another refinement behaviour and obtains a better reordering of the hypotheses to be validated and a benefit in finding the element at fault; (iii) the validated hypotheses provided by the BRS are available to the AAA, this triggers yet another refinement behaviour, which has the same functionality as the previous one, but the reordering is based on validated hypotheses which are more accurate; (iv) if no information is available from the BAI or BRS, the AAA proceeds with its hypotheses validation as a standalone agent. Therefore, if the other agents are down or they are too slow to provide the information, the AAA will continue and find a faulty element although its diagnosis will be less reliable and will take longer.

The restoration process is activated whenever a disturbance is detected. Once the disturbance is identified, the disturbance identifier is sent to the CSI-R which acquires the snapshot of the network, corrects any inconsistencies which have arisen in its representation, and calculates the power flow solution of the current state. This information is then passed onto the SRA so that it can prepare for its restoration planning. The SRA waits until the diagnosis agents have informed it of the element suspected of being at fault (VALIDATED-HYPOTHESES) and then proceeds to prepare a restoration plan. If, during this plan preparation, the SRA is informed that the equipment at fault is different from that originally indicated by either the AAA or the BRS, then it replans the restoration taking this information into account.

The UIA is the interface through which the user accesses the results produced by the agent community. During the diagnosis phase, the user is presented with both the tentative (early) list of suspected hypotheses and the final (validated) list. During the restoration phase, the UIA supports a more participatory interaction between the user and the agent community. The user is presented with the restoration plan and can then decide to modify it, run a detailed simulation to see the effects of the plan on the state of the network or ask for a new plan to be devised taking into account new constraints which he specifies. The UIA also supports a reporting functionality in that the control engineer can ask for the logs of the disturbances to be presented and analysed.

Observations and Reflections

This application has been in operation in Iberdrola's North DCR since the beginning of 1994 and has afforded a number of benefits. Firstly, the agent system gives better results than its stand alone counterparts because it takes multiple types of knowledge and data into account and then integrates them in a consistent manner. Secondly, the agent system is more robust because there are overlapping functionalities which means partial results can be produced in the case of component (agent) failure. Thirdly, some results can be provided more quickly because cooperation provides a short cut (see previous section). Fourthly, the functionalities of the different domain systems can be increased independently which makes them easier to maintain (see, for example, the argument for developing the BAI). Fifthly, the control engineer is provided with an integrated view of the results he is interested in. Finally, the system has been designed to be open so that new agents can be added in an incremental manner.

One of the key features of this multi-agent system is the way it handles fault diagnosis by using two different types of data (the non-chronological alarms used by the AAA and the chronological alarms used by the BRS) and two different points of view (the typical diagnosis approach of hypothesis generation and validation used by the AAA and BRS, and the BAI's monitoring approach which provides a high level view of the status of the network). With this set-up, it is possible to dynamically select the solution method which is best suited to the current situation. For example, if the BRS is operational, but the AAA is not, the solution provided to the control engineer is the one created by the BRS; but if both the BRS and the AAA are running, the solution provided is the one which is mutually agreed between them. Also the fact that multiple agents are trying to generate the same results can be exploited to avoid repetition of certain tasks if it is deemed desirable in a particular context. For example both the AAA and the BRS can provide GENERATED-HYPOTHESES, consequently if the generated hypotheses that are provided by the BRS are available to the AAA before it starts its own generation task, then this task need not be executed and the hypotheses provided by the BRS can be used instead (see figure 3). This ability to flexibly manage, at runtime, multiple sources of data and multiple problem solving perspectives provides enormous robustness to the overall system because if one of the agents crashes the others will still be able to provide some form of solution.

As a consequence of the experience obtained during the development and installation of this multi-agent system, some important application design improvements are foreseen for the future. The first drawback of the current system is caused by the fact that the energy transport network covers a vast geographic area (meaning there is a huge amount of topological information) and that it encompasses a number of different voltage levels. As the network's behaviour depends both on the voltage level and the geographic location, the main problem solving agents (the AAA, BRS, BAI and SRA) have to contain and manage information about Iberdrola's entire transport network. This means the agents require a substantial amount of memory and computing resource because their searches are through such a large problem space. To combat this problem, the next version of the system will be designed so that the agents work with smaller portions of the network; this will make them easier to debug and maintain, faster in execution, and more cost effective in that they could run on PCs instead of workstations.

The second drawback of the current system is that all the agents have uniform knowledge of the network. For instance, the AAA applies virtually the same knowledge about protective relays to its 400 kV, 220 kV and 132 kV voltage levels. However, if there were one AAA (or BRS or SRA) per voltage level, it would be possible to customise their domain knowledge. For example, the following pieces of potentially useful knowledge could be reflected: the fact that protective relays on the 400 kV voltage level are more reliable than the ones at 132 kV voltage level, and the fact that the 400 kV network is more interconnected than the 132 kV network and has more complex breaker structures (like central breakers or rings of breakers). A further source of heterogeneity which is

currently masked is that the network itself is the result of the fusion of a number of smaller transport networks that were developed by different companies before coming under the overarching umbrella of Iberdrola. Thus, for example, the protective relays of the Northwestern Iberdrola network are different from the protective relays in the rest of the network. Again this information could be exploited if smaller and more specialised agents were developed.

On a more general note, although the ARCHON approach was influenced to a large extent by the need to incorporate pre-existing industrial control software into a multi-agent community, it is felt that cooperative interworking, heterogeneity, semi-autonomy, and loose coupling are attributes that are likely to be encountered when building most complex systems (even if they are built entirely from scratch and are not deliberately conceived as DAI systems). This belief is based on three main observations. Firstly, most large organisations, where the majority of complex systems reside, have departmental structures which need to be observed, but the individuals within these structures often need to work together in a coherent manner. Secondly, the components (including both the humans and the software) within an organisation, and even within a department, are likely to be heterogeneous simply because each one has to be based upon a different modelling paradigm in order to be effective. Finally, complexity is best handled by devolving responsibility for decisions to the level at which the actions are performed (hence the problem solvers will be loosely coupled and semi-autonomous).

In most cases the domain systems that already existed were each fairly complex and had been designed to encompass those aspects of the domain that could be expressed within a coherent modelling paradigm. They were designed in this manner because conventional wisdom dictates that when building a system using a particular modelling approach, one should include everything that is known about the system's world that can be expressed within that model. However, after due reflection and observation it is noted that once several such conventional systems are brought together into a cooperative framework, then completeness and comprehensiveness are no longer the key criteria for allocating agents. In a multi-agent system, questions of the efficiency of the overall system are likely to be paramount. This, in turn, may dictate that an agent is identified with the smallest possible coherent and autonomous entity. As a consequence the system may have a large number of such agents which may, in turn, have an implication on the performance of the overall system. The ARCHON experience as yet does not extend to systems with many (hundreds) of agents, however it is likely that in such situations the designer may need to consider if some of the smaller agents need to be coalesced again.

Other important experiences from this work are that: (i) speed of operation is a factor even where the real-time requirements of the underlying processes are longer than milliseconds (because a great many concurrent processes are active and their interactions are cumulative); (ii) passing of intermediate results (or progress reporting) is an effective means of increasing system parallelism; (iii) data which is exchanged between agents ought to have a degree of persistence so that troubleshooting can take place and audit records can be maintained; (iv) significant improvements in the overall application can be obtained through relatively straightforward cooperative interactions. Finally, it is important that a clear and detailed DAI methodology is worked out, especially in view of the above discussion about agent granularity - ARCHON's informal hybrid approach is an important first step in this direction but it needs to be made more rigourous and have an even clearer link to the software development process.

References

Abel, E., Laresgoiti, I., Perez, J., and Corera, J., and Echavarri, J., 1993. A multi-agent approach to analyse disturbances in electrical networks. Proc. Fourth Int. Conf. on Expert Systems Applications to Power Systems, Melbourne, Australia.

Barandiaran, J., Laresgoiti, I., Perez, J., Corera, J., and Echavarri, J., 1991. Diagnosing faults in electrical networks. Proc. EXPERSYS 91, Paris, France.

Corera, J., Echavarri, J., Laresgoiti, I., Lazaro, J. M., and Perez, J., 1993. On-line expert system for service restoration. Proc. Fourth Int. Conf. on Expert Systems Applications to Power Systems, Melbourne, Australia.

Cockburn, D., and Jennings, N. R., 1995. ARCHON: A DAI System for Industrial Applications. In Foundations of DAI (eds. G. M. P. O' Hare & N. R. Jennings), Wiley Interscience

Jennings, N. R., Corera, J., Laresgoiti, I., Mamdani, E. H., Perriolat, F., Skarek, P. and Varga. L. Z. 1995. Using ARCHON to develop real-word DAI applications for electricity transportation management and particle accelerator control. IEEE Expert.

Jennings, N. R. 1994a. Cooperation in Industrial Multi-Agent Systems. Series in Computer Science - Vol 43, World Scientific Press.

Jennings, N. R. 1994b. The ARCHON system and its Applications. Proc Second Int. Working Conf. on Cooperating Knowledge Based Systems, Keele, UK, 13-29

Jennings, N. R., and Pople, J. A., 1993. Design and Implementation of ARCHON's Coordination Module. Proc. Workshop on Cooperating Knowledge Based Systems, Keele, UK, 61-82.

Jennings, N. R., Varga, L. Z., Aarnts, R., Fuchs, J., and Skarek, P. 1993. Transforming Standalone Expert Systems into a Community of Cooperating Agents. Int. Journal of Engineering Applications of AI 6 (4) 317-331.

Jennings, N. R., and Wittig, T., 1992. ARCHON: Theory and Practice. In Distributed Artificial Intelligence: Theory and Praxis (eds. N. M. Avouris and L.Gasser), Kluwer Academic Press, 179-195.

Tuijnman, F., and Afsarmanesh, A., 1993. Distributed Objects in a Federation of Autonomous Cooperating Agents. Proc. Int. Conf. on Intelligent and Cooperative Information Systems, Rotterdam, Netherlands, 256-265.

Varga, L. Z., Jennings, N. R., and Cockburn, D. 1994. Integrating Intelligent Systems into a Cooperating Community for Electricity Distribution Management. Expert Systems with Applications (1994) 7 (4) 563-579.