Practical applications of multi-agent systems in electric power systems

V. M. Catterson, E. M. Davidson, S. D. J. McArthur

Abstract

The transformation of energy networks from passive to active systems requires the embedding of intelligence within the network. One suitable approach to integrating distributed intelligent systems is multi-agent systems technology, where components of functionality run as autonomous agents capable of interaction through messaging. This provides loose coupling between components that can benefit the complex systems envisioned for the smart grid. This paper reviews the key milestones of demonstrated agent systems in the power industry, and considers which aspects of agent design must still be addressed for widespread application of agent technology to occur.

1 Introduction

Energy networks are currently undergoing transformation, from the passive system with reactive protection and control of the last 100 years, to the active and proactive systems required to deliver smarter grid capabilities. This change requires a raft of technologies and data analysis

techniques, to fill what the European SmartGrids Technology Platform document calls "a toolbox of smart grid capabilities" [1].

The smart grid vision is of multiple, complex systems interacting safely and correctly, delivering new network functionality such as active network management and integrated condition monitoring, with the aim of more efficient use of power system resources and greater participation of end users. The end result will be a power network that flexibly allows the connection of more embedded and renewable generation, while handling changing load profiles due to electric vehicles and demand side management.

Delivery of smarter grid services will require a technology for integrating all of this functionality, in a way that does not negatively impact key metrics such as customer minutes lost in the UK, and system average interruption duration index (SAIDI) in the US. One such technology being proposed is multi-agent systems, where large, complex systems can be built from smaller, autonomous agents of functionality [2]. In essence, agents offer a platform for designing and delivering the complex smart grid application, from a set of constituent parts that are currently being researched and tested.

This paper examines the role that agent technology could play within the power engineering domain, by reviewing the key contributions within the field of agents in power, and highlighting where more research is required. Section 2 presents the key concepts and definitions, with consideration of some related areas of research. Section 3 reviews the state-of-the-art of agent applications by discussing some milestone industrial demonstrators and deployments of agent technology. The agent application areas discussed include post-fault analysis, condition monitoring, congestion management, and automatic restoration. From this, Section 5 draws out some challenges which remain for agent development in the power industry. Section 6 concludes the paper.

2 Concepts

Many concepts within the area of agent development mean different things to different people. This leads to obfuscation of the potential benefits that agents may offer, and thus can impede understanding.

The IEEE Power and Energy Society (PES) Multi-Agent Systems Working Group aims to offer leadership in this area by defining the terminology most used within the field. Through the publication of two papers—one specifically on concepts and approaches to developing agent systems [3], the other on technologies and tools for implementing agents [4]—the working group has provided some clarity on definitions that all in the field may examine, and decide whether or not the stated benefits of agents are achievable.

This section presents the key terms with discussion of the working group-agreed definitions, and provides discussion of the similarities and differences between agents and other intelligent system techniques and computing platforms.

2.1 Agent

The first task is to define an agent. The power engineering community has largely settled on the definition proposed by Wooldridge [5], that an agent is "a software (or hardware) entity that is situated in some *environment* and is able to *autonomously* react to changes in that *environment*". The environment here is the world surrounding the agent, able to be observed by the agent through sensors, and influenced by the agent through it taking some action in the world, but with a separation between the two.

That the agent is autonomous means that it is able to "exercise control over its actions" [6]. This is a somewhat vague definition, but essentially implies that no other program or entity

exerts direct control over the agent; unlike a programming subroutine, which must be called by a higher-level thread of execution in order to run, the agent itself schedules its behaviours to run or otherwise. It should be noted that the agent can perform an operation on behalf of another entity, and indeed, it is common for an agent to react to an incoming message by taking some action. But the important distinction is that the sending agent is not calling the receiver's functions, and the thread of execution does not pass from the message sender to the message receiver; rather, two separate threads of execution exist and it is possible for the receiving agent to refuse to service the sender's request. A legitimate example of this situation would be where the receiving agent does not have enough memory to complete an action, in which case it replies with a message refusing the request.

The definition of an agent acting autonomously within an environment is only one of many posited definitions. The computer science community has proposed a number over the years, including [7–10]. However, many of the alternative definitions are relatively vague and intentionally inclusive, allowing existing systems to be redefined as agents. This offers no tangible benefit to the engineering community, which is more concerned with the differentiating factors between agent systems and non-agent systems, in order to assess the potential benefits of this technology. As a result, Wooldridge's definition is the one of choice.

2.2 Intelligent Agent

Building on the definition of an agent, Wooldrige also defines the notion of an intelligent agent. Where an agent displays autonomy in an environment, an intelligent agent displays flexible autonomy as identified by three properties:

- Reactivity: the ability to respond to an event in a timely manner
- Pro-activeness: the ability to "take the initiative", as demonstrated through goal-directed behaviour

- Social ability: the ability to interact with other agents in the environment.

Any intelligent agent will display these three properties in some combination, and different intelligent agents will display varying levels of each. However, it is the goal-directed behaviour of an intelligent agent which truly sets it apart from other systems, making it more than just capable of responding to particular situations in a reactive manner.

2.3 Multi-Agent System

A collection of agents and intelligent agents interacting within the same environment comprises a multi-agent system. Since the system is made up of individual autonomous agents, each with their individual goals and reactive capabilities, there is no overall system goal or global goal. Instead, it is left to the system designer to ensure that the correct mix of agents is present within the system in order to achieve their desired outcomes.

For example, if two tasks are required to achieve the system designer's goal, the designer must ensure that agents within the system have the local goals appropriate to performing both tasks, either by giving both goals to one agent, or one goal each to two different agents.

This suggests the ability of agents to co-operate to meet their goals. Co-operation and co-ordination can be achieved due to the social ability of intelligent agents, which allows communication about tasks and goals through adherence to standards for inter-agent messaging.

2.4 Agent Standards

Within the power engineering community, the use of standards is becoming increasingly important. Utilities desire systems that conform to industry standards for data exchange, including

the Common Information Model (CIM) [11] and IEC 61850 [12]. As a result, any agent systems developed for this arena must adhere to appropriate standards, to give open, plug-and-play systems. Since agents interact via messaging, standards-adherence largely means conforming to open standards for inter-agent communication.

As with other communication systems, messaging occurs in different standardised layers. These are message transport (the delivery of a message), message format (the agent communication language), and message content (both grammar and vocabulary).

The Foundation for Intelligent Physical Agents (FIPA) have created a suite of standards that cover the full stack of agent communication. FIPA was accepted as a standards committee of the IEEE Computer Society in 2005, and the FIPA standards have become the de facto standards for agent development within the power domain.

2.4.1 The Agent Platform

Message transport and delivery to agents is specified through FIPA's Agent Management Reference model, which defines "the normative framework within which FIPA agents exist and operate. It establishes the logical reference model for the creation, registration, location, communication, migration and retirement of agents." [13]. This means FIPA agents reside on a platform which handles messaging using a transport protocol such as HTTP or IIOP, and handles addressing of agents by keeping track of those present on the platform.

This functionality is supplemented by a specified Directory Facilitator (DF) agent. Agents joining the platform can register their services with the DF, which provides a searchable directory of services and agents within the system. This allows a decoupling of tasks within the multi-agent system: agents need not be hard coded with the contact details of others whose services they need; instead, agents can search the DF for the currently-available providers of those services.

The searchable DF and seamless message transport and delivery offered by a FIPA-compliant agent platform are the foundations of an open, extensible multi-agent system.

2.4.2 Message Format

The agent platform provides the path for message transport between agents, but the format of the messages themselves must also be standardised for meaningful communication to occur. Some of the earliest agent systems defined proprietary communication languages (e.g. [14]), which were followed by interaction through blackboard architectures (e.g. [15]), before development of standardised communication languages such as Knowledge Query and Manipulation Language (KQML) [16].

Building on KQML, the FIPA Agent Communication Language (FIPA-ACL) standardises the structure and flow of messages between agents on a FIPA agent platform [17]. Message fields include receiving and sending agent addresses, message sequence information, and the message content. All these fields are optional; the only mandatory field is the message performative which indicates the intention behind the message, such as whether it is a request for an action, a query for information, or informing another agent of some information.

FIPA-ACL messages are often sent as part of a sequence of interaction, such as an inform message being sent in response to a query. Some common interaction patterns are also specified by FIPA.

2.4.3 Message Content

While FIPA-ACL provides the headers for a message, the content structure is out of scope for that standard. The syntax and semantics, or grammar and lexicon of message content are provided by a content language and ontology, respectively.

The FIPA Semantic Language (FIPA-SL) has reached a stable standard version [18], providing a first order logic-style grammar for messages. Ontology is application-specific, making it impossible for a general standards-body such as FIPA to define a standard ontology. Therefore, agent system designers must design and build an ontology that suits the particular application their agents operate on.

The IEEE PES Multi-Agent Systems Working Group has defined an ontology for power engineering applications, largely based on CIM, which can be used as a basis for defining an application-specific ontology [19]. It is anticipated that agents from different designers will largely communicate about power systems concepts such as substations, voltages, and circuit breakers, and so this ontology of common terms will ease the creation of open agent systems.

2.5 Agents and Related Technologies

It is worth discussing the distinction between a theoretical engineering agent, which conforms to the given definitions and displays goal-directed behaviour and social ability, and a practical engineering agent, which is designed and built using the tools and approaches available today. While the above definitions indicate what an intelligent agent looks like, the reality is that such a theoretical agent is not fully realisable using current resources. For practical applications the intelligent agent will be implemented in such a way that it may not have the social ability that we may desire, or its behaviour will not be as goal-directed as the ideal.

This is because the concept of an intelligent agent builds on other parts of computer science which are not currently at the stage of delivering the task-decomposition and question-answering technologies that a theoretical agent requires. This section considers these related technologies in the context of agent systems, identifying the place within agent research for each of these fields.

2.5.1 Agents and Intelligent Systems

The heart of an intelligent agent is its ability to reason about its goals and the state of the world, and take actions to try to achieve its goals. The choice of reasoning technique employed is an open question, with many different solutions being proposed. These range from general purpose reasoning such as that offered by AI planners [20] or the belief-desire-intention framework [21], to application-specific solutions such as neural networks or other pattern recognition techniques trained to perform a specific operation [22].

In theory, an intelligent agent should display goal-directed behaviour. This strongly suggests that an 'ideal' agent should embody a planner, allowing it to assemble sequences of actions to achieve its goal. However, including a planner has disadvantages. One is purely practical: many agents will only have one or two potential actions at their disposal, designed to achieve one specific task. AI planning is pointless for organising such a simple set of actions, and would not improve the ability of the agent. Implementing planners within each agent would require significant developmental effort, for no gain in the general case.

However, even in the specific cases where an agent does have the potential to meet its goals in multiple ways, including a planner is not always helpful. Consider a situation where an agent A requires an agent B to perform a service in order to meet its goal. If agent B fails to complete the service, agent A must find an alternative solution. It may be the case that two other agents, C and D, can each perform subtasks that together meet the same objective as agent B's service. For agent A to be able to reason about this situation and replan to request services of agents C and D, it must be able to perform *task decomposition*, breaking down the one service provided by B into the two subtasks offered by C and D.

Task decomposition has been under extensive research in the field of semantic web services [23–25]. To be able to perform task decomposition on the service offered by agent B, the designer of agent A would need to describe the service to such a fine level of detail that it would be

equivalent to implementing the service itself.

Consider the service of performing a Discrete Fourier Transform ($Y_k = \sum_{n=0}^{N-1} X_n e^{-\frac{2\pi i}{N} nk}$) on a dataset. Given the definition of the DFT, this service could be decomposed into sub-services including sum, multiply, power, and so on. However, understanding the combination in which these sub-services should be used requires as much information as implementing the DFT itself. At this point, the only benefit to service decomposition is that it allows the spreading of computational load, which changes the topic of discussion to distributed computing for optimisation, rather than software organisation and systems design.

This raises the question of whether the theory of an ideal agent is faulty. If we cannot reach the state of goal-driven co-operating agents, then why use an agent approach for system design? In practice, the benefits of this approach are actually coincidental to the theory of agents. The value offered by multi-agent systems design is due to the decomposition of the system into reusable, redeployable, communicating components, each capable of performing its task autonomously. This allows a system to be reconfigured for use in multiple scenarios, and eases the merging of multiple systems by enforcing the use of common agent communications technologies. Additionally, because each autonomous agent performs a single clearly-defined task, it can be thoroughly designed and tested before deployment in the system.

This is not to say that planning has no place in a multi-agent system. While a practical agent cannot achieve goal-directed behaviour through co-operation, it can do so autonomously provided that it has sufficient reasoning capability to allow pro-active behaviour. Examples of this may range from using neural networks for parameter prediction, or rule based reasoning for operator decision support, through to AI planning for network reconfiguration.

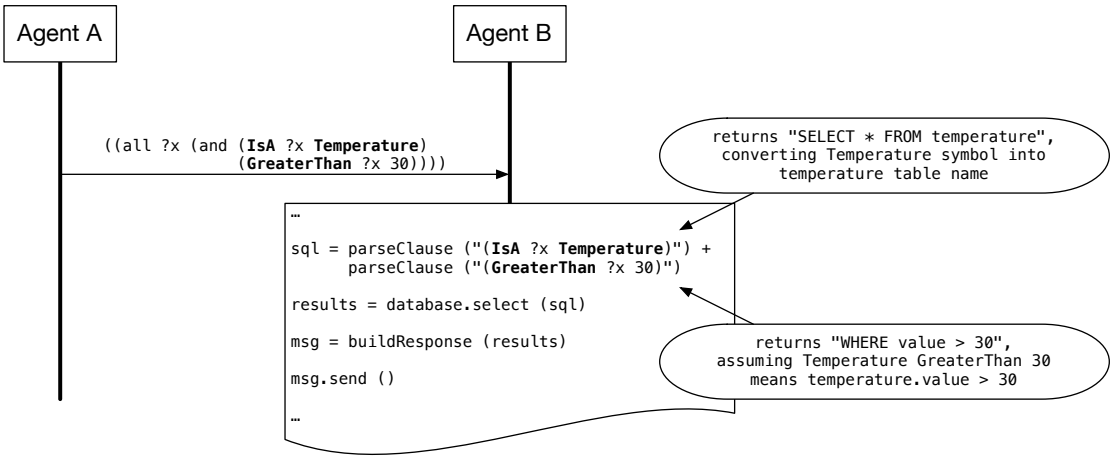

Figure 1: When an agent receives a message, it must translate content terms into method calls and database queries according to its internal knowledge representation.

2.5.2 Agents and Service-Oriented Architectures

The social ability of an intelligent agent theoretically allows for an agent to receive any standards-conforming query about information it knows and respond with an appropriate answer. In practice, agents are largely hard-coded to recognise and respond to set messages on topics they know, making the practice very different from the theory [26].

The problem is one of knowledge-representation, and is related to the choice of reasoning technique that gives an agent its intelligence. When a message is received, the agent must perform some conversion from terms within the content field into actions to take, methods to call, databases to consult, etc., before performing a second conversion from action results, method return values, and database results into ontology and content language terms for sending a response (see Fig. 1).

One way around the problem is to design the agent so that it uses the ontology for knowledge representation, and all its knowledge and state is already stored in ontology terms. However, in almost all situations a practical agent is going to hold more detailed knowledge than can be represented using the ontology. Say an agent can respond to queries for temperature data, taken

from a particular sensor. The ontology will provide the ability to describe a temperature, giving it a value, a unit, and the time it was measured. The agent must also know how to address the sensor and how to take readings, which is low-level detail that should not be covered by the ontology. The knowledge representation capabilities of the agent will necessarily be broader than the ontology alone, meaning that at some point of the agent's reasoning process there will be translation between ontology terms and specific methods or actions the agent can take.

This problem of restricted social ability has been recognised, and the most advanced solution is currently the JADE Semantics Add-On (JSA) [26]. Designed for the JADE agent platform, the JSA simplifies the task of linking ontology terms with particular agent actions, separating out the hard-coded elements to one particular place and thus simplifying maintainability issues.

However the general problem still stands, that the ontology lists a set of terms that an agent can employ in messages, while the implementation of what those terms mean is dependent on each agent's designer making an explicit link between an ontology term and the agent's actions.

Service oriented architectures (SOA) must also address this problem when exposing interfaces for web services. Web services are similar to agent services, in that heterogeneous systems operate together using a service model. The standards governing web services include the Web Service Description Language (WSDL) [27] and Universal Description, Discovery, and Integration (UDDI) protocol [28].

WSDL is an XML-format language used to give information about services. Contrary to its name, WSDL describes how to access and use the service rather than describing the service itself, with parameters such as the address, message template, and data types. It can be considered a language for service syntax, without semantic information.

UDDI is a service registry, similar to the FIPA DF. Service providers can register their WSDL service description with a UDDI node, advertising how to access that service. UDDI is a passive matchmaker, storing information generally referred to as white pages, yellow pages, and green

pages information. White pages information involves general contact details for the service provider; yellow pages information uses a taxonomy of business or industry categories to describe a service; and green pages information includes interfaces and addresses for service access (in WSDL). Searching can be performed on individual parameters.

The semantics of a web service are therefore described in the yellow pages information, based on categories defined in published taxonomies. This is exactly analogous to an agent system's ontology: the meaning of terms must be inferred by the system designer, and explicit links made between service terms and software actions. The main difference is that web service providers tend to explicitly limit the interface of calls or messages a service will respond to, while creators of agents may aim to reach the ideal of general social ability for their agents.

The reason that general social ability is considered desirable for agents is to future-proof the system. Designers hope that by giving an agent the ability to respond to messages with content not yet in use, future extensions to the system will be able to ask questions and request services of the agent without needing to update the agent. A trivial example is for the temperature sensor agent described above to respond to queries for the temperature at a given date to cover the current use case, but also being given the ability to respond to requests for all temperatures in a time range in case this is needed in the future.

There is a trade-off to be made between development time now versus development time in the future given the likelihood of such functionality being needed. Much time could be invested in allowing an agent to respond to many conceivable queries, but without much chance of recouping the benefit through many system extensions. The "future-proof-ness" of a system cannot be measured, and so there is little way of gauging the concrete benefits of extending agent social abilities. The most practical course is to design the agents to be as flexible as possible in handling the current use cases, such as by using the JSA, and trust that future extensions will be able to manage with the agent interfaces this provides.

2.5.3 The Practical Agent

In summary, the theoretical idea of an intelligent agent would take much more development time than is practical for an engineering system, and would not provide substantially improved capabilities over the practical implementation suggested in this paper. The goal-directed behaviour that seems a clear distinguisher between agent and non-agent systems is in practice difficult to achieve, since most agents have a limited set of potential actions they can take, and the state-of-the-art in task decomposition does not allow for detailed reasoning about the capabilities of other agents in the system. Social ability is also considered to set agent systems apart from others, but in practice is limited by the designer's interpretation of the ontology for a given agent.

With these considerations in mind, the definition of a practical intelligent agent is one which embodies some intelligent system technique, and displays autonomy through two features:

- It maintains a thread of execution that is distinct from other programs, i.e. no program or external system makes calls to the agent's functions or methods, where execution transfers from the external system to the agent code and back;
- It interfaces with the external world through messaging and taking action in the environment, i.e. other agents and external systems can affect the agent's behaviour only by submitting messages, or by altering the environment in a way intended to alter the agent's behaviour (e.g. leaving a pheromone marker in an ant colony simulation).

The degree to which the agent maintains autonomy from the agent platform is an implementation detail, and does not affect the first point. The JADE platform, for example, does not instantiate separate agents in separate operating system threads. However, it switches between agents as a mini scheduler, meaning that the agents within the platform are kept distinct (as operating system threads are kept distinct within a process). The agent can make calls to platform functions, such as for sending and receiving messages from the transport layer, but the platform can only

affect the agent by suspending, migrating, destroying, etc. through requests from the engineer. The platform supports the agent, rather than controlling the agent.

The capability of an agent to act autonomously has certain practical benefits for power engineering applications. Separate applications and algorithms can be implemented and tested as isolated systems, in order to ensure correct operation; and then deployed together in the field to gain the benefits of information-sharing across systems. This also means that stand-alone legacy systems (which, by their nature, run autonomously) can be integrated into new systems and interfaced with new data sources.

The following section considers particular examples of practical agent systems in the literature, and assesses the contributions they make to the field.

3 Review

A 2007 review of the field [3] identified four application areas where agent research was ongoing. One significant area of activity was in agent-based simulation of energy markets, for example [29, 30]. In this paper, we focus on "operational" use of agents rather than simulation: agent technology being used as an integration platform for in-field systems deployment. Taking this definition, this section identifies the state-of-the-art in agent system design, by discussing some of the key implementations of power engineering applications.

3.1 ARCHON

One of the earliest examples of an agent-based system in the power domain is ARCHON (ARchitecture for Co-operative Heterogeneous ON-line systems) [31]. The design intention was to integrate a set of pre-existing expert systems for different aspects of distribution network fault

diagnosis. Each expert system originally operated in isolation, with an engineer drawing conclusions about the state of the network from the output of each system.

The out-of-sequence SCADA alarms gathered from the network are first processed by the Alarm Analysis Agent, which tries to identify the faulted component. At the same time, a Breakers Supervisor agent analyses the correctly-sequenced SCADA data from individual substations. A third agent identifies the extent of the network that was blacked-out, and a fourth agent acts as an interface between the agent system and the control room computer system.

Integration of the systems allows automatic corroboration of diagnoses. If analysis of the network-wide, out-of-order alarms identifies the same faulted component as analysis of the local, chronological alarms, confidence in the diagnosis is higher. The Black-out Area Identifier agent provides extra information for Alarm Analysis, which can confirm or alter the fault diagnosis.

This system explicitly employed loose coupling of agent messaging interfaces in order to integrate legacy systems in a flexible, alterable way. Each agent was designed to keep the intelligence layer distinct from the social layer, with explicit mappings between them. The aim was to allow changes to the social interactions among agents without having to alter the intelligence layer. In effect, the legacy systems can remain the same regardless of the deployment scenario, while the agents can come together in varying ways by altering the messaging layers.

This capability was exploited in subsequent implementations, where further agents were added to the system for switching schedule production and weather monitoring for lightning strike location [14].

ARCHON used bespoke message formats and protocols in addition to a blackboard-like distributed database for information-sharing between agents. The latter avoids the need for transferring large datasets between agents over the message transport channel, with point-to-point and broadcast messaging allowing agents to alert others to updates in the datasets available. While this approach met the needs of the system at the time, the use of standards for messaging

allows more openness of the architecture, which was one of the drivers for using the agents' loose coupling.

ARCHON was supported by Iberdrola, and has been deployed in a control room in Bilbao [32]. It was pioneering by showing the potential of multiple autonomous systems linked by loosely coupled interfaces: benefiting the application by corroborating multiple diagnoses, and benefiting the system by flexibly allowing deployment of different sets of agents. While it aimed to be an open architecture that agents from any designer could join, in practice the lack of standardised messaging and the use of a distributed database would hinder this.

3.2 PEDA

The Protection Engineering Diagnostic Agents (PEDA) system was created for post-fault analysis, using not only SCADA data but also Digital Fault Recorder (DFR) data to diagnose network faults and validate that the protection system operated correctly [33]. Like ARCHON, PEDA integrates a number of legacy intelligent systems in order to improve the diagnostic conclusions of the system, while additionally collecting data from and corroborating across multiple sources of data via agents that use FIPA-conforming messaging.

SCADA data is analysed by an Incident and Event Identification agent, which uses an expert system to classify the type of incident and identify relevant alarms. The information from this agent is used to prioritise retrieval of fault records by the Fault Record Retrieval agent. Fault records are passed to the Fault Record Interpretation agent, which uses an expert system to identify key information such as fault type and clearance times. A Protection Validation and Diagnosis agent wraps a model-based reasoning system to check whether the protection system operated correctly for this fault. Information generated by all these agents is archived by a Collation Agent, and an Engineering Assistant Agent presents system conclusions to the engineer.

A number of the PEDA agents were deployed in the UK with SP PowerSystems [34], where certain practical issues provided an instructive case study for the use of multi-agent systems in robust industrial applications. Specifically, the stability of the agent platform used for development was under question, which meant the agents deployed upon it were consequently unable to run without crashes. Secondly, the FIPA standards themselves had undergone revision, and some of the messaging did not conform to the new versions of the standards. Finally, the user requirements for the system had changed over time, meaning the engineers wanted robust archival of system output.

In short, the deployment of this system led to significant advances in understanding of the software engineering challenges of a multi-agent system, by requiring techniques and approaches to achieve the stability and maintainability required by an industrial system. The agent system can only be as robust as the tools it employs, and the importance of platform selection and long term data storage were highlighted. With the use of FIPA standards for messaging, this system was more open to agents with different designers than ARCHON, but the lack of standards versioning information in messages meant the agents had to be updated to conform to latest releases of the standards.

3.3 COMMAS

In addition to network fault diagnosis, agents have been used for diagnosis of faults in items of plant. The COndition Monitoring Multi-Agent System (COMMAS) was originally applied to gas turbine start-up sequences, before being implemented for transformer condition monitoring [22]. Implementation and use of this system has continued over many years, and similarly to PEDA, the requirements have changed over this period of time. This section considers the first practical implementation and the shift to a second practical implementation deployed with National Grid.

The original implementation aimed to diagnose defects causing partial discharge (PD) behaviour

within a transformer. PD data was gathered from sensors, then a Feature Extraction agent calculated feature vectors from the data. Three pattern recognition techniques were wrapped as Diagnosis Agents, which used the feature vectors for defect classification. The three diagnoses were collected by a Corroboration Agent, which made the final diagnosis. An Information Agent presented an interface for the engineer.

An agent approach was used for extensible system development: since the agents located others of the appropriate type through the DF, extra Diagnosis Agents could be added to the system and their diagnoses were seamlessly collected by the Corroboration Agent. Development of new Diagnosis Agents was also simplified, since all the messaging behaviours of locating feature vectors and providing diagnoses were identical. In the terminology of ARCHON, the intelligence layer differs between different Diagnosis Agents, while the messaging layer is identical.

This architecture suited PD monitoring in the laboratory, but had some disadvantages that were revealed as requirements changed. The first issue was that the diagnosis workflow is set by each type of agent looking for the previous type in the chain, making it difficult to insert steps into the workflow. For example, later work inserted anomaly detection capabilities between data gathering and feature extraction [35], so that diagnosis of a defect would only be performed if the transformer's behaviour changed. This required substantial alteration of the messaging layer of the Feature Extraction agent, to stop it from seeking sensor data until it received a start signal from the anomaly detector.

The second issue arose when new sources of data were available. In a project with the UK's National Grid, two transformers were instrumented with temperature, vibration, and current sensors with the aim of monitoring for changes in behaviour. The intelligent system technique chosen for this was Conditional Anomaly Detection [36], which operated on raw data instead of feature vectors, and did not require a separate stage for corroboration. Since the workflow for this data was so different from the PD data analysis, a whole new set of agents were required to interpret this data.

Taking these changes in requirements into account, the agent architecture was radically redesigned. Five different agent roles were defined: data provider, service provider, data director, archive, and interface [37]. Data provider agents make data available to the system, by connecting to sensors, databases, historical files, or web resources. Service providers perform data interpretation, such as calculating feature vectors or classifying defects. The archive provides long term storage of information generated by agents. Data directors contain task-specific knowledge allowing them to make links between data providers and service providers, and ensure information is archived. Interface agents provide ways of allowing information out of the system, such as to a user interface or a second system.

This architecture allows the diagnosis workflow to alter based on a given data director's knowledge. For the situation of adding anomaly detection to PD monitoring, it would be the data director that is altered from directing data straight to the feature extraction agent, to sending it for anomaly detection prior to feature extraction. This is a significant difference: in the first system design the feature extraction agent had to change even although the process of extracting features remained the same, whereas with the second design the data director changes because the flow of data (the sphere of this agent's knowledge) has to change.

The change in architecture reflects a change in design intention. The new implementation focusses on handling data from different sources using different techniques in a flexible and extensible way, by imposing the five roles required for many condition monitoring applications. This architecture was implemented for the transformer monitoring application described above, and monitored the transformers on-site for 12 months before decommissioning [36]. It has also been implemented for dissolved gas analysis of historical data and PD analysis of lab samples [37]; deployed on-site for HVDC reactor monitoring [38]; and deployed on a wireless sensor network for transformer monitoring [39].

While condition monitoring is traditionally treated as a distinct application from network monitoring, diagnosis of any faults or disturbances on the network can give extra information about diagnoses of plant health. The integration of the PEDA and COMMAS systems has been dis-

cussed [40], with the aim of using knowledge of network events such as high loading to inhibit the raising of alarms for transformer high temperatures or increased PD. Both systems conform to the FIPA standards for agent platform and agent messaging, but since ontology is not standardised, messages generated by one system are unintelligible to the other.

To date, this problem has not been fully overcome. The IEEE PES Multi-Agent Systems Working Group recognises it as a barrier to interoperation of agent systems [4], and has developed an upper ontology of common terms within the power engineering domain [19]. This can be used as the basis for application-specific ontology development, which simplifies system integration to some extent. Systems will likely communicate about high-level concepts such as transformers, substations, and faults, and the upper ontology provides a data model for such terms.

3.4 SPID

Certain events within the power system can lead to cascade tripping, which can rapidly black-out large portions of the network unnecessarily. The Strategic Power Infrastructure Defence (SPID) System was created to identify and resolve the types of hidden failure of components that can leave the system vulnerable to cascading events [41].

Agents are organised into three layers. The first set react quickly to events, by sending inhibiting signals to protection devices. At the next level, co-ordination agents use heuristic knowledge to decide if events at the reactive layer need further analysis. Finally the deliberative layer contains agents which do longer-term analysis, such as identifying potential hidden failures of components, reconfiguring the network, or generating a restoration plan after an incident.

The benefits of an agent approach are derived from the autonomy of agents in each layer [41]. Reactive agents can make local decisions and react quickly, compared to deliberative agents that gather wide area data before drawing conclusions. At the same time, such a complex system

maintains a level of robustness from the independence of each agent: if one ceases to function, it will affect others but not necessarily the whole system.

Later work identified the need for robust tools for implementing such a system, in order to deploy a practical industrial solution [42]. The choice of such tools was presented as an open question. Since rapid communication is vital for this application, it is unlikely agent messaging is FIPA-compliant due to the size of FIPA messages, although the choice of technology is not explicitly stated. This suggests that the architecture is not particularly open, but the application of security threat analysis could be used as an argument against an open system.

3.5 IntelliTEAM II

S&C Electric Company produce the IntelliTEAM II Automatic Restoration System [43], which decides and executes a restoration plan after isolation of a faulted line section. This is achieved by communication between agents responsible for adjacent lines. Each agent can control components such as switches and reclosers on its line, but action can only be taken after agreement with neighbouring agents. Each agent checks that its constraints will still be met before agreeing to neighbouring actions.

The system has been deployed with the ENMAX Power Corporation in Canada on key circuits. This shows the robustness required of industrial applications has been met. However, little information has been published on the workings of the agents themselves, other than the radio system employed for communication. A second trial on the Isle of Wight sees monitoring and restoration of an 11kV network, which is susceptible to interruptions due to weather conditions [44].

3.6 PowerMatcher

A more recent commercial product is PowerMatcher, an agent-based solution to demand management in congested or constrained networks [45]. The design intention for this system is to dynamically match supply and demand through an open market for trading in a local area.

Each device has a control agent with knowledge of how they produce or consume power. This leads to three types of device agent: stochastic operation agents (such as solar or wind energy converters), shiftable operation agents (such as ventilation or air conditioning systems), and user action devices (such as lighting and computers) [45]. In addition, there are three other types of agent for operating the market: an auctioneer agent, which finds the equilibrium price point; a concentrator agent, which can represent a cluster of device agents in the auction; and an objective agent, which can maintain objectives other than simply balancing supply and demand to, for example, operate a virtual power plant [46]. Through the use of concentrator agents, small local markets can trade with wider area markets, allowing the computational complexity of the market to scale logarithmically [46].

Agent technology was chosen for this application for reasons of scalability and openness [46]. As the number of devices using PowerMatcher grows, the appropriate agents can be instantiated from the same six agent templates described above, with no additional developer overhead. Since all agents are autonomous, the deployment of new devices does not affect the current device agents, other than by altering prices fixed through auction. Scalability of computation has been designed in to the architecture through the concentrator agents.

While the design and development of the system is considered scalable, in practice it may be limited by implementation of the agent platform. One of the most widely used platforms is JADE [26], a Java-based, FIPA-compliant platform that is generally held to produce reliable, scalable systems. However, the scalability and robustness of a JADE-based agent system is significantly impacted by deployment choices, such as whether to deploy agents on a single

platform or whether to federate multiple platforms containing subsets of agents [47].

PowerMatcher does not use the JADE platform, but instead agents run on a FIPA-compliant platform created by the PowerMatcher developers [48]. Little detail of this platform is published, but it is likely that deployment choices and platform implementation will have an effect on scalability. The extent of this effect has not been reported.

Openness of the PowerMatcher system is a design goal, in order to allow devices with different creators to integrate with the system. Messaging between agents does not currently comply with FIPA standards, instead being a proprietary messaging format tailored to the current application use cases [49]. However, it has been recognised that standards-compliant messaging would open up the architecture [48], and FIPA-compliant messaging is a goal for future development.

PowerMatcher is in advanced stages of field testing, with three demonstrator sites of varying sizes confirming that PowerMatcher can perform its required tasks [50]. One test deployed control agents with five devices for some months in 2006, with the objective of reducing imbalances between supply and demand. The second field test grouped five micro-CHP devices as a virtual power plant for one month in 2007, with the aim of reducing the peak load of the local network. The third and largest test is called PowerMatcher City, where agents are deployed with 100 controllable devices, split across 30 households and two laboratories. This third test is still ongoing.

3.7 AuRA-NMS

Taking a very different approach to congestion management, the Autonomous Regional Active Network Management System (AuRA-NMS) was designed to encompass multiple techniques for operating distribution networks more efficiently. This involved automating the decision-making required for power flow management, voltage control, and network optimisation, using agent

technology as an integration platform for some or all of these techniques.

While other network control research has suggested the use of agents, AuRA-NMS takes a significantly different design approach [51]. Instead of allocating an agent to each device in the network, AuRA-NMS splits agents along functional boundaries, giving each agent in the system autonomous capabilities for tackling different tasks such as voltage control or power flow management. The specific techniques tested are constraint programming, current tracing, and optimal power flow for power flow management; and case-based reasoning and constraint programming for voltage control [52, 53].

As with ARCHON, the loose coupling of the messaging interface is the key agent feature that gives tangible benefit to this application, rather than the scalable development offered by instantiating multiple agents of the same type that is exploited by PowerMatcher. This has the interesting effect of allowing flexible geographical deployment of agents where computing resources allow it and where the utility desires.

For example, one proposed field trial would deploy power flow management functionality distributed across three substations, while a second field trial would retain all computation within the utility's control room [53]. The location of agents for these trials is simply a choice, not a requirement, as the agents are not tied to specific items of plant. The first field trial is currently ongoing.

Similarly to the condition monitoring application of COMMAS, openness and future integration with other systems is highly desirable for AuRA-NMS. As such, the same issues of standards-compliance and ontology choice will affect the level to which it is practically possible to create an open architecture. Any progress which is made in one of these areas should be directly applicable to the other.

3.8 INTEGRAL

INTEGRAL is an EU programme which aims to develop an ICT platform for distributed control of electrical networks based on commonly available ICT components, standards, and technologies [54]. A key element of the INTEGRAL programme is demonstration of the proposed platform under a number of different network conditions: normal conditions (managing DER to minimise cost); critical conditions (maintaining stability); and emergency conditions (restoration of supply). The programme has three different pilot deployments, each deployment focusing on a particular condition. Agent technology features in all three pilots.

The pilot for normal network conditions, in the Netherlands, uses PowerMatcher, which we have already discussed in section 3.6.

For critical network conditions, a site trial in Spain uses MAS technology for the control of a microgrid [54]. Based on earlier work [55], agents are used to match supply and demand with the microgrid in response to grid events. The microgrid is connected to a UPS which is used to simulate a grid connection. Loads and generators within the microgrids are managed from a central PC which communicates via Zigbee with embedded controllers on each of the loads/generators. The central PC hosts a multi-agent system, implemented using JADE, which manages devices on the microgrid. The architecture of the MAS is such that each controllable device is assigned a local control agent. The local control agent manages the control of that individual device. A single microgrid central controller agent is responsible for the co-ordination of the local control agents and thus the whole system.

Agent concepts also play a role, allbeit more limited, within the emergency conditions pilot of INTEGRAL. The third pilot aims to demonstrate self-healing in response to faults, i.e. automatic supply restoration [56]. In terms of MAS design and implementation, a single intelligent agent is responsible for restoration within an area of network known as a cell. Performance of the agent and the underlying ICT infrastructure on which it relies is being tested on a low voltage

laboratory-based test network, which has been sized to represent a real 20 kV area of distribution network in France [57].

4 Interest in Agents and Smart Grids

As networks become "smarter", more flexible and automated, utilities and researchers are looking for platforms and technologies that can support a system of complex, interacting, sometimes competing functionality. The requirement is to allow different objectives, such as power flow management, voltage regulation, and automated restoration, to co-exist harmoniously within the smart grid management system. Data and measurements should be available to all functionality that requires it, while actions taken to meet one objective should not negatively impact another unless a negotiated agreement has been reached.

Various bodies have recognised the need for standards and platforms to support this vision. EPRI's IntelliGrid architecture [58] and the US Department of Energy's GridWise Architecture [59] are two such industrially-backed efforts at standardisation. These architectures are explicitly technology-agnostic, specifying the required functionality of a smart grid platform without suggesting particular implementations. At a high level, both architectures require platforms to allow interoperability of functionality, extensibility, and reliability.

Multi-agent systems aim to embody these features, making agent technology a strong contender for delivery of smart grid capability. Interoperability through standards-conformance, extensibility through deployment of newer agents, and reliability through distributed deployment of agents offering redundant functionality are accepted strengths of agent-based system design [3]. Many researchers are investigating the design of agent systems to address smart grid questions, as evidenced by a wealth of recent publications [51, 53, 60–86].

While it is early for physical demonstrators and field trials of these applications, it is anticipated

that agent technology will continue to grow in popularity, as it offers the means of creating the flexible, extensible, and robust systems required for the smart grid.

5 Challenges

The progress of these systems towards industrial implementations and demonstrators shows that agent technology can be used to build robust systems from autonomous components. However, there still remain some practical challenges to wide-spread deployment of multi-agent systems within power engineering. These can be summarised as ensuring a system design will deliver the required functionality, whether that functionality is a technical capability or an industry requirement.

The first aspect of this is how to design an agent system. Many methodologies exist (some listed in [4]), but the choice of methodology will greatly influence the final structure of the architecture. Most methodologies design from a top-down perspective, where an engineering task such as condition monitoring or post fault analysis is broken down into successive subtasks until they seem appropriate for agent behaviours. This will produce a system that meets the required functional capability for the current use case, but may be limited in extensibility to future expansions. Specifically, the social ability of the agents will be limited to creating and understanding the messages required to perform the high level task.

An alternative is to design from a bottom-up perspective, considering what actions and messaging capabilities an agent should have given its knowledge of the world. A fully bottom-up approach would fall into the trap described earlier, of investing much development effort into aspects that are not required for the current use case with the aim of future-proofing the system, but may never be required in the future.

Therefore, a balance between top-down and bottom-up design seems appropriate, with the design

emphasis more on a top-down approach to meeting current use cases. However, the current use cases can be quite complex, for example by allowing the autonomy of agents to flexibly change at run-time. Research into degrees of autonomy has considered the circumstances in which it is appropriate for robots within a team to act independently versus co-ordinating behaviour with others [87, 88]. This work presented various use cases in which robots had to act autonomously for timely response to environmental events, but in other situations take orders from others in order to meet the team's objectives. Choosing the appropriate degrees of autonomy for agents within the power system is an important question not yet explored within the industry.

Finally, designers must have ways of verifying that once implemented, agents will behave in the expected ways in order to robustly continue to meet its design objectives. Validation and verification of agent code is in many ways like assessing the correctness of any software system, with the possible interactions of various agents both simplifying and adding complexity to the task.

Since agents operate as autonomous components they can be thoroughly tested as stand-alone units, entirely analogously to unit testing of software classes. External systems such as other agents can be simulated through mocking [89], where the intended responses are hard-coded into a mock agent, thus decoupling the behaviour of agents within the system for test purposes. Since many tools and frameworks exist for unit testing and mocking, agent testing is no more difficult than standard unit testing, relying as it does on defining a test set with appropriate coverage to ensure all behaviours are exercised.

However, by using FIPA standards for messaging, the number of legitimate messages that could potentially be sent to an agent is practically infinite. This complicates the task of ensuring an agent cannot be damaged by unexpected input, as the test set of messages may be much larger and more complex than a test set of parameters for standard unit testing of a software function. Further, the asynchronous interaction of agents means that they are potentially more sensitive to the timing and order of messages received from multiple sources, which can require test sets of the same messages sent at varying intervals to get full coverage.

Software verification is not a new area, and there is nothing inherent about agent design that requires more or less effort expended on testing. However, the testing procedure for an agent system is something rarely discussed in publications on the use of agents in industry, and if agent-based systems are to be deployed for safety critical or safety related applications in the power industry, it is an area that must be given consideration.

6 Conclusion

This paper has discussed the theoretical and practical benefits that multi-agent systems technology can bring to power engineering applications. Through a review of some of the key agent-based systems in the literature, the progress of this technology towards robust industrial installations has been shown, culminating in ongoing field trials and commercial products currently available. Each of these demonstrators covers a different strand of the functionality envisioned for the smart grid concept, showing through example that multi-agent systems can provide the integrating technology for such a complex system.

However, there are still some aspects of agent design and implementation that could be improved. Design methodology, degrees of autonomy, and verification and testing have not been fully explored in the literature, and further research would benefit engineering agent systems. With this in mind, further demonstration systems and industrial deployments are required, to continue advancing the practice of multi-agent design.

References

[1] SmartGrids, "Vision and strategy for Europe's electricity networks of the future," 2006, available http://www.smartgrids.eu/.

[2] V. M. Catterson, E. M. Davidson, and S. D. J. McArthur, "Agents for Active Network Management and Condition Monitoring in the Smart Grid," in *9th Int. Conf. on Autonomous Agents and Multiagent Systems*, Toronto, Canada, May 2010.

[3] S. D. J. McArthur, E. M. Davidson, V. M. Catterson, A. L. Dimeas, N. D. Hatziargyriou, F. Ponci, and T. Funabashi, "Multi-Agent Systems for Power Engineering Applications—Part 1: Concepts, Approaches and Technical Challenges," *IEEE Trans. Power Systems*, vol. 22, no. 4, pp. 1743–1752, Nov. 2007.

[4] ——, "Multi-Agent Systems for Power Engineering Applications—Part 2: Technologies, Standards and Tools for Building Multi-Agent Systems," *IEEE Trans. Power Systems*, vol. 22, no. 4, pp. 1753–1759, Nov. 2007.

[5] M. Wooldridge, "Intelligent Agents," in *Multi-agent Systems*, G. Weiss, Ed. The MIT Press, Apr. 1999, pp. 3–51.

[6] S. Franklin and A. Graesser, "Is it an agent or just a program?" in *Proc. 3rd International Workshop on Agent Theories, Architectures, and Languages.* Springer-Verlag, 1996.

[7] S. Russell and P. Norvig, *Artificial Intelligence: A Modern Approach.* Prentice-Hall Inc., 1995.

[8] P. Maes, "Artificial life meets entertainment: life-like autonomous agents," *Communications of the ACM*, vol. 38, no. 11, pp. 108–114, 1995.

[9] L. N. Foner, "Entertaining agents: a sociological case study," in *Proc. 1st International Conference on Autonomous Agents*, 1997.

[10] B. Hayes-Roth, "An architecture for adaptive intelligent systems," *Artificial Intelligence: Special Issue on Agents and Interactivity*, vol. 72, pp. 329–365, 1995.

[11] IEC, "Energy Management System Application Program Interface (EMS-API) - Part 301: Common Information Model (CIM) base," 2005, document IEC 61970-301.

[12] ——, "Communications Networks and Systems in Substations," 2005, document IEC 61850.

[13] Foundation for Intelligent Physical Agents (FIPA), "Agent Management Specification," 2002, http://www.fipa.org/specs/fipa00023/SC00023J.html.

[14] D. Cockburn and N. R. Jennings, "ARCHON: A Distributed Artificial Intelligence System for Industrial Applications," in *Foundations of Distributed Artificial Intelligence*, G. M. P. O'Hare and N. R. Jennings, Eds. Wiley, Apr. 1996, pp. 319–344.

[15] S. Talukdar, "Asynchronous teams: Cooperation schemes for autonomous agents," *Journal of Heuristics*, vol. 4, no. 4, pp. 295–321, 1998.

[16] T. Finin, Y. Labrou, and J. Mayfield, "KQML as an agent communication language," in *Software Agents*, J. Bradshaw, Ed. The MIT Press, 1997.

[17] Foundation for Intelligent Physical Agents (FIPA), "FIPA ACL Message Structure Specification," 2002, http://www.fipa.org/specs/fipa00061/SC00061G.html.

[18] ——, "FIPA SL Content Language Specification," 2002, http://fipa.org/specs/fipa00008.

[19] V. M. Catterson, P. C. Baker, E. M. Davidson, and S. D. J. McArthur, "An upper ontology for power engineering applications," 2010, available http://ewh.ieee.org/mu/pes-mas/.

[20] N. Muscettola, P. P. Nayak, B. Pell, and B. C. Williams, "Remote Agent: to boldly go where no AI system has gone before," *Artificial Intelligence*, vol. 103, no. 1-2, Aug. 1998.

[21] A. S. Rao and M. P. Georgeff, "BDI agents: From theory to practice," in *1st International Conference on Multi-Agent Systems (ICMAS-95)*, 1995.

[22] S. D. J. McArthur, S. M. Strachan, and G. Jahn, "The design of a multi-agent transformer condition monitoring system," *IEEE Trans. Power Systems*, vol. 19, no. 4, pp. 1845–1852, Nov. 2004.

[23] S. A. McIlraith, T. C. Son, and H. Zeng, "Semantic Web Services," *IEEE Intelligent Systems*, vol. 16, no. 2, pp. 46–53, Mar./Apr. 2001.

[24] A. Ankolekar, M. Burstein, J. R. Hobbs, O. Lassila, D. Martin, D. McDermott, S. A. McIlraith, S. Narayanan, M. Paolucci, T. Payne, and K. Sycara, "DAML-S: Web Service

Description for the Semantic Web," in *Proc. 1st Int. Semantic Web Conf. (ISWC)*, Sardinia, Italy, 2002.

[25] A. Gómez-Pérez, R. González-Cabero, and M. Lama, "ODE SWS: A Framework for Designing and Composing Semantic Web Services," *IEEE Intelligent Systems*, vol. 19, no. 4, pp. 24–31, Jul./Aug. 2004.

[26] F. Bellifemine, G. Caire, and D. Greenwood, *Developing Multi-Agent Systems with JADE*. John Wiley and Sons Ltd, 2007.

[27] E. Christensen, F. Curbera, G. Meredith, and S. Weerawarana, "Web Services Description Language (WSDL) 1.1," 2001, http://www.w3.org/TR/wsdl.

[28] L. Clement, A. Hately, C. von Riegen, and T. Rogers, "Universal Description, Discovery and Integration v3.0.2 (UDDI)," 2005, http://www.oasis-open.org/committees/uddi-spec/doc/spec/v3/uddi-v3.0.2-20041019.htm.

[29] A. Somani and L. Tesfatsion, "An Agent-Based Test Bed Study of Wholesale Power Market Performance Measures," *IEEE Computational Intelligence Magazine*, vol. 3, no. 4, pp. 56–72, 2008.

[30] H. Li and L. Tesfatsion, "The AMES wholesale power market test bed: A computational laboratory for research, teaching, and training," in *IEEE Power Engineering Society General Meeting, 2009*, Jul. 2009.

[31] T. Wittig, N. R. Jennings, and E. M. Mandan, "ARCHON — A framework for intelligent cooperations," *IEE-BCS Journal of Intelligent Systems Engineering*, vol. 3, no. 3, pp. 168–179, 1994.

[32] N. R. Jennings, J. M. Corera, I. Laresgoiti, E. H. Mamdani, F. Perriollat, P. Skarek, and L. Z. Varga, "Using ARCHON to develop real-world DAI applications for electricity transportation management and particle accelerator control," *IEEE Expert*, vol. 11, pp. 64–70, 1996.

[33] J. A. Hossack, J. Menal, S. D. J. McArthur, and J. R. McDonald, "A multiagent architecture for protection engineering diagnostic assistance," *IEEE Trans. Power Systems*, vol. 18, no. 2, pp. 639–647, May 2003.

[34] E. M. Davidson, S. D. J. McArthur, J. R. McDonald, T. Cumming, and I. Watt, "Applying multi-agent system technology in practice: automated management and analysis of SCADA and digital fault recorder data," *IEEE Trans. Power Systems*, vol. 21, no. 2, pp. 559–567, May 2006.

[35] A. J. Brown, V. M. Catterson, M. Fox, D. Long, and S. D. J. McArthur, "Learning Models of Plant Behavior for Anomaly Detection and Condition Monitoring," *Engineering Intelligent Systems*, vol. 15, Jun. 2007.

[36] V. M. Catterson, S. D. J. McArthur, and G. Moss, "On-Line Conditional Anomaly Detection in Multivariate Data for Transformer Monitoring," *IEEE Trans. Power Delivery*, vol. 25, no. 4, pp. 2556–2564, Oct. 2010.

[37] V. M. Catterson, S. E. Rudd, S. D. J. McArthur, and G. Moss, "On-line Transformer Condition Monitoring through Diagnostics and Anomaly Detection," in *15th Int. Conf. Intelligent Systems Application to Power Systems (ISAP)*, Nov. 2009.

[38] V. M. Catterson, S. D. J. McArthur, M. D. Judd, and A. S. Zaher, "Managing Remote Online Partial Discharge Data," *IEEE Trans. Power Delivery*, vol. 23, no. 4, pp. 1754–1762, Oct. 2008.

[39] P. C. Baker, V. M. Catterson, and S. D. J. McArthur, "Integrating an Agent-based Wireless Sensor Network within an Existing Multi-agent Condition Monitoring System," in *IEEE International Conference on Intelligent Systems Application to Power Systems (ISAP)*, Curitiba, Brazil, Nov. 2009.

[40] V. M. Catterson, E. M. Davidson, and S. D. J. McArthur, "Issues in integrating existing multi-agent systems for power engineering applications," in *13th Int. Conf. Intelligent Systems Application to Power Systems (ISAP)*, Oct. 2005.

[41] C.-C. Liu, J. Jung, G. Heydt, V. Vittal, and A. G. Phadke, "The strategic power infrastructure defense (SPID) system: A conceptual design," *IEEE Control Systems Magazine*, vol. 20, no. 4, pp. 40–52, Aug. 2000.

[42] H. Li, G. W. Rosenwald, J. Jung, and C.-C. Liu, "Strategic Power Infrastructure Defense," *Proc. IEEE*, vol. 93, no. 5, pp. 918–933, May 2005.

[43] D. M. Staszesky, D. Craig, and C. Befus, "Advanced Feeder Automation is Here," *IEEE Power and Energy Magazine*, vol. 3, no. 5, pp. 56–63, Sep./Oct. 2005.

[44] D. MacLeman, W. Bik, and A. Jones, "Evaluation of a self healing distribution automation scheme on the Isle of Wight," in *20th International Conference and Exhibition on Electricity Distribution (CIRED 2009)*, Jun. 2009.

[45] K. Kok, C. Warmer, and R. Kamphuis, "The PowerMatcher: Multiagent Control of Electricity Demand and Supply," in *Agents in Industry: the best from the AAMAS 2005 Industry Track, IEEE Intelligent Systems*, Mar./Apr. 2006, vol. 21(2).

[46] J. K. Kok, M. J. J. Scheepers, and I. G. Kamphuis, "Intelligence in Electricity Networks for Embedding Renewables and Distributed Generation," in *Intelligent Infrastructures*, R. R. Negenborn, Z. Lukszo, and J. Hellendoorn, Eds. Springer, 2009.

[47] E. M. Davidson and S. D. J. McArthur, "Exploiting Multi-agent System Technology within an Autonomous Regional Active Network Management System," in *14th Int. Conf. Intelligent Systems Application to Power Systems (ISAP)*, Nov. 2007.

[48] M. P. F. Hommelberg, B. J. van der Velde, C. J. Warmer, I. G. Kamphuis, and J. K. Kok, "A novel architecture for real-time operation of multi-agent based coordination of demand and supply," in *IEEE Power Engineering Society General Meeting, 2008*, Jul. 2008.

[49] C. J. Warmer, I. G. Kamphuis, R. M. Hermans, J. Frunt, A. Jokić, and P. P. J. van den Bosch, "Balancing Services in Smart Electricity Grids Enabled by Market-Driven Software Agents," in *9th Int. Conf. on Autonomous Agents and Multiagent Systems*, Toronto, Canada, May 2010.

[50] J. K. Kok, "Multi-Agent Coordination in the Electricity Grid, from Concept towards Market Introduction," in *9th Int. Conf. on Autonomous Agents and Multiagent Systems*, Toronto, Canada, May 2010.

[51] E. M. Davidson, S. D. J. McArthur, C. Yuen, and M. Larsson, "AuRA-NMS: Towards the delivery of smarter distribution networks through the application of multi-agent systems technology," in *IEEE Power Engineering Society General Meeting, 2008*, Jul. 2008.

[52] E. M. Davidson, S. D. J. McArthur, M. J. Dolan, and J. R. McDonald, "Exploiting intelligent systems techniques within an autonomous regional active network management system," in *IEEE Power Engineering Society General Meeting, 2009*, Jun. 2009.

[53] E. M. Davidson, M. J. Dolan, G. W. Ault, and S. D. J. McArthur, "AuRA-NMS: An Autonomous Regional Active Network Management System for EDF Energy and SP Energy Networks," in *IEEE Power Engineering Society General Meeting, 2010*, Jul. 2010.

[54] G. Peppink, R. Kamphuis, K. Kok, A. Dimeas, E. Karfopoulos, N. Hatziargyriou, N. Hadjsaid, R. Caire, R. Gustavsson, J. M. Salas, H. Niesing, J. van der Velde, L. Tena, F. Bliek, M. Eijgelaar, L. Hamilton, and H. Akkermans, "INTEGRAL: ICT-platform based Distributed Control in electricity grids with a large share of Distributed Energy Resources and Renewable Energy Sources," in *1st International ICST Conference on E-Energy*, Athens, Greece, Oct. 2010.

[55] A. L. Dimeas and N. D. Hatziargyriou, "Operation of a multi-agent system for microgrid control," *IEEE Trans. Power Systems*, vol. 20, no. 3, pp. 1447–1455, Aug. 2005.

[56] L. Le-Thanh, R. Caire, B. Raison, S. Bacha, F. Blache, and G. Valla, "Test bench for self-healing functionalities applied on distribution network with distributed generators," in *IEEE Bucharest PowerTech 2009*, Jul. 2009.

[57] N. Hadjsaid, L. Le-Thanh, R. Caire, B. Raison, F. Blache, B. Ståhl, and R. Gustavsson, "Integrated ICT framework for distribution network with decentralized energy resources: Prototype, design and development," in *IEEE Power Engineering Society General Meeting, 2010*, Jul. 2010.

[58] EPRI, "Intelligrid Architecture Status Report: Technology Transfer Activities and Recommendations," Dec. 2006, available from http://intelligrid.epri.com/technical_results.html.

[59] GridWise Architecture Council, "Interoperability Consitution Whitepaper," Dec. 2006, available from http://www.gridwiseac.org/about/publications.aspx.

[60] M. Pipattanasomporn, H. Feroze, and S. Rahman, "Multi-agent systems in a distributed smart grid: Design and implementation," in *IEEE PES Power Systems Conference and Exposition (PSCE 2009)*, Mar. 2009.

[61] J. Ko, I.-H. Shin, G.-L. Park, H.-Y. Kwak, and K.-J. Ahn, "Design of a Multi-agent System for Personalized Service in the Smart Grid," *Security-Enriched Urban Computing and Smart Grid: Communications in Computer and Information Science*, vol. 78, pp. 267–273, 2010.

[62] M. P. F. Hommelberg, C. J. Warmer, I. G. Kamphuis, J. K. Kok, and G. J. Schaeffer, "Distributed Control Concepts using Multi-Agent technology and Automatic Markets: An indispensable feature of smart power grids," in *IEEE Power Engineering Society General Meeting, 2007*, Jun. 2007.

[63] R. Belkacemi and A. Feliachi, "Multi-agent design for power distribution system reconfiguration based on the artificial immune system algorithm," in *Proceedings of 2010 IEEE International Symposium on Circuits and Systems (ISCAS)*, May 2010, pp. 3461–3464.

[64] S. Chouhan, W. Hui, H. J. Lai, A. Feliachi, and M. A. Choudhry, "Intelligent reconfiguration of smart distribution network using multi-agent technology," in *IEEE Power Engineering Society General Meeting, 2009*, Jul. 2009.

[65] Q. Pang, H. Gao, and X. Minjiang, "Multi-agent based fault location algorithm for smart distribution grid," in *10th IET International Conference on Developments in Power System Protection (DPSP 2010)*, Mar. 2010.

[66] D. A. Cohen, "GridAgents: Intelligent agent applications for integration of distributed energy resources within distribution systems," in *IEEE Power Engineering Society General Meeting, 2008*, Jul. 2008.

[67] A. Saleem, K. Heussen, and M. Lind, "Agent services for situation aware control of power systems with distributed generation," in *IEEE Power Engineering Society General Meeting, 2009*, Jul. 2009.

[68] A. Saleem and M. Lind, "Requirement analysis for autonomous systems and intelligent agents in future Danish electric power systems," *International Journal of Engineering, Science, and Technology*, vol. 2, no. 3, pp. 60–68, 2010.

[69] J. M. Solanki, S. K. Solanki, and N. Schulz, "Multi-agent-based reconfiguration for restoration of distribution systems with distributed generators," *Integrated Computer-Aided Engineering*, vol. 17, no. 4, pp. 331–346, 2010.

[70] I. Zabet and M. Montazeri, "Implementing cooperative agent-based protection and outage management system for power distribution network control," in *4th International Power Engineering and Optimization Conference (PEOCO)*, Jun. 2010, pp. 318–324.

[71] ——, "Decentralized control and management systems for power industry via multiagent systems technology," in *4th International Power Engineering and Optimization Conference (PEOCO)*, Jun. 2010, pp. 549–556.

[72] Q. Ai, J.-H. Wu, and J. Zhang, "Strategies for optimal use of clean distributed energy in smart grid," *High Voltage Engineering*, vol. 35, no. 11, pp. 2813–2819, Nov. 2009.

[73] Z. Jiang, "Computational intelligence techniques for a smart electric grid of the future," *Advances in Neural Networks: Lecture Notes in Computer Science*, vol. 5551/2009, pp. 1191–1201, 2009.

[74] C. Xin, L. Feipeng, L. Yunkun, H. Yaping, and Z. Yunyong, "Protective relaying on-line setting calculation system," in *2010 Asia-Pacific Power and Energy Engineering Conference (APPEEC)*, Mar. 2010.

[75] S. Karnouskos and T. N. de Holanda, "Simulation of a smart grid city with software agents," in *2009 Third UKSim European Symposium on Computer Modeling and Simulation*, Nov. 2009.

[76] I.-H. Lim, M.-S. Choi, S.-J. Lee, and T. W. Kim, "Intelligent distributed restoration by multi-agent system concept in DAS," in *15th Int. Conf. Intelligent Systems Application to Power Systems (ISAP)*, Nov. 2009.

[77] I. Chao, O. Ardaiz, R. Sanguesa, L. Joita, and O. F. Rana, "Optimizing decentralized grid markets through group selection," in *International Conference on Complex, Intelligent and Software Intensive Systems*, Mar. 2008, pp. 951–956.

[78] A. Molderink, V. Bakker, M. G. C. Bosman, J. L. Hurink, and G. J. M. Smit, "Domestic energy management methodology for optimizing efficiency in smart grids," in *IEEE Bucharest PowerTech 2009*, Jul. 2009.

[79] N. Cai and J. Mitra, "A decentralized control architecture for a microgrid with power electronic interfaces," in *North American Power Symposium (NAPS)*, Sep. 2010.

[80] R. Bhuvaneswari, S. K. Srivastava, C. S. Edrington, D. A. Cartes, and S. Subramanian, "Intelligent agent based auction by economic generation scheduling for microgrid operation," in *Innovative Smart Grid Technologies (ISGT)*, Jan. 2010.

[81] H. F. Wedde, S. Lehnhoff, C. Rehtanz, and O. Krause, "Intelligent agents under collaborative control in emerging power systems," *International Journal of Engineering, Science, and Technology*, vol. 2, no. 3, pp. 45–59, 2010.

[82] Z. Vale, H. Morais, P. Faria, H. Khodr, J. Ferreira, and P. Kadar, "Distributed energy resources management with cyber-physical SCADA in the context of future smart grids," in *MELECON 2010: 15th IEEE Mediterranean Electrotechnical Conference*, Apr. 2010, pp. 431–436.

[83] H. Morais, Z. A. Vale, C. Ramos, and I. Praça, "Virtual power producers simulation—Negotiating renewable distributed generation in competitive electricity markets," in *IEEE PES/IAS Conference on Sustainable Alternative Energy (SAE)*, Sep. 2009.

[84] A. A. Aquino-Lugo and T. J. Overbye, "Distributed intelligent agents for service restoration and control applications," in *North American Power Symposium (NAPS)*, Sep. 2008.

[85] S. M. Amin, "For the Good of the Grid," *IEEE Power and Energy Magazine*, vol. 6, no. 6, pp. 48–59, Nov./Dec. 2008.

[86] S. Mullen and G. Onsongo, "Decentralized agent-based underfrequency load shedding," *Integrated Computer-Aided Engineering, IOS Press*, vol. 17, no. 4, pp. 321–329, 2010.

[87] J. Shah, P. Conrad, and B. C. Williams, "Fast Distributed Multi-agent Plan Execution with Dynamic Task Assignment and Scheduling," in *International Conference on Automated Planning and Scheduling (ICAPS 09)*, Thessaloniki, Greece, Sep. 2009.

[88] P. R. Conrad, J. Shah, and B. C. Williams, "Flexible Execution of Plans with Choice," in *International Conference on Automated Planning and Scheduling (ICAPS 09)*, Thessaloniki, Greece, Sep. 2009.

[89] K. Beck, *Test Driven Development: By Example*. Addison-Wesley Professional, 2002.