Chapter 15
Multi-Agent Technology for Power System Control

Robin Roche[1], Fabrice Lauri, Benjamin Blunier, Abdellatif Miraoui, Abderrafiâa Koukam.

Abstract – The electric grid is evolving towards what has been defined as the "smart grid paradigm." The development of communication infrastructures provides power electronics interfaces with the ability to control complex power systems in efficient and scalable ways and in real-time. Multi-Agent Systems (MAS) are based on distributing information and computing algorithms for complex networks, and are an excellent technological solution for this application. This chapter focuses on applications of MAS in power systems and describes how they can be used with other artificial intelligence techniques in order to make the grid smarter and more flexible. In addition to presenting the basics of multi-agent theory, this chapter covers some design procedures and provides several examples, as well as perspectives for future developments of MAS in power systems control.

15.1 Introduction

With the recent developments in smart grid technology, communication plays an increasingly central role in power systems, particularly at the distribution level. Although SCADA (Supervisory Control And Data Acquisition) systems have long been used for monitoring and remote control at the transmission and sub-transmission levels, they are expected to be utilized for further control in a more encompassing way than before, especially due to the new possibilities offered by power electronic interfaces. As shown in Chap. 2, power converters, for example, can accommodate various levels of current and voltage, manage power flows, and directly interact with renewable or alternative energy sources, storage units and sophisticated loads.

As energy prices and demand keep rising with depletion of fossil fuel resources and growing environmental concerns, the necessity to integrate distributed energy resources (renewable or alternative) has become of paramount importance. The increasing penetration of sources such as solar panels and wind turbines contributes to the intermittent nature of generation, requiring storage compensation that needs to be controlled adequately. The concept of smart grid control originated from such dispersed generation model, with the requirement of demand-side management (for example by shedding or shifting the use of some loads when required), while taking into account additional parameters such as customer behavioral expectations.

Three clear trends can be observed in contemporary power systems: they are becoming at the same time more complex, more distributed, and less predictable. Therefore, control systems have to evolve from their monolithic and centralized structures and adapt to more decentralized ones, where the intelligence of the system can be achieved through cooperation of multiple entities and components. Multi-agents systems (MAS), which have been applied in computer science studies for years, have characteristics that make them suitable for acting as

[1] R. Roche (🖂), F. Lauri, B. Blunier, A. Miraoui, A. Koukam
Université de Technologie de Belfort-Montbéliard, IRTES-SET, 90010 Belfort, France
e-mail: robin.roche@utbm.fr

a basis for building modern distributed control systems. In addition, artificial intelligence techniques can be embedded in some agents with smart features, particularly in automating tasks traditionally performed by human operators.

The next sections discuss the following important questions that are related to how multi-agent systems can be developed for controlling power systems:

- What do modern control systems require?
- What are MAS and why are they relevant in this field?
- How to design a MAS and what choices do designer have to face?
- What choices need to be made in the design process, among which alternatives?
- For which applications are MAS used or could MAS be used?
- How may MAS uses evolve in the future?

15.2 Distributed Control Systems

Control systems are at the core of all power systems, in that they allow them to operate efficiently and safely. Control is a term related to a wide variety of processes with different objectives and constraints. The following discussion provides some useful analysis in how such systems may evolve towards distributed, intelligent and autonomous paradigms, especially relevant to smart grid control needs.

15.2.1 Control Systems Objectives

The main objectives of control systems can be sorted according to these priorities:

1. Safety: The first one is the overall safety of the system, for example in avoiding a catastrophe such as the ones of Chernobyl or Fukushima or even a simple blackout that requires distributed generation to be disconnected from distribution lines.
2. Protection: A control system must protect equipment from self-inflicted or external damages.
3. Stability and reliability: The system must operate correctly (e.g., regulate voltage and reactive power, manage storage in conditions that do not degrade the performance, etc.) as long as possible while providing its customers with power of satisfactory quality. Therefore, stability and reliability must be taken in consideration in the overall control system design.
4. Economics: All other system priorities must be fulfilled while minimizing costs. Traditional energy management control primarily focuses on economics, but a broader paradigm should be considered in taking all technical specifications and economics as well.

From the above requirements, several complementary (sometimes overlapping) control system functionalities must be understood over several timescales (Fig. 15.1), related to the following physical constraints:

- *Protection* (milliseconds to seconds): in order to meet the first two priorities, control systems for utility-connected equipments must react very quickly to disturbances, such as short-circuits and overvoltages.
- *Regulation* (seconds): systems need to regulate voltage levels and reactive power in order to maintain stability and reliability.
- *Load following* (minutes to hours): generation needs to meet demand, resulting in frequency control. Such characteristic overlaps with further economic constraints.
- *Scheduling* (hours to days): unit commitment or maintenance scheduling are required functionalities for operating power systems, typically affecting all the above priorities.
- *Investment/expansion planning* (months to years): when and how the power system needs to evolve (by building new power plants, expanding transmission lines, etc.) is typically the slowest deciding priority.

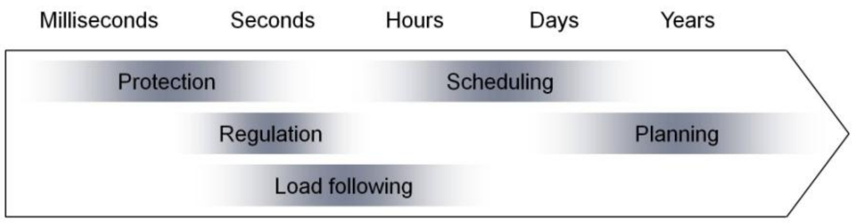

Fig. 15.1 Control systems can be divided into several complementary applications, each with its prevailing timescale, from milliseconds to years.

From these requirements and functionalities, a four level hierarchy in which several complementary layers can be defined for control systems:

- A physical layer uses actuators and sensors to monitor multiple information,
- A grid operation level with control strategies related to sources, load shedding, and so on.
- A decision-making layer considers energy management supporting scheduling and forecasting.
- And a user level, in which operators interact with the system.

15.2.2 Modern Architectures for Power Systems Control

Modern control systems used by utilities are based on supervisory control and data acquisition (SCADA) systems. SCADAs are information systems used for monitoring and supervising power systems or industrial processes, but without control functionalities. However, with recent popularization of the term, SCADAs are now associated with extensive systems, sometimes performing control actions. Fig. 15.2 shows the structure of a basic SCADA, which is a fully centralized architecture, where remote terminal units (RTUs) provide communication interfaces with physical components such as sources, loads, and so on.

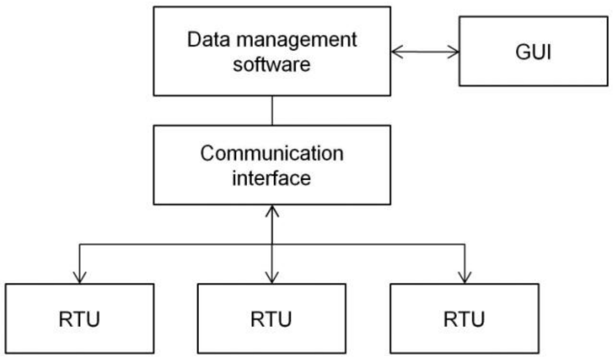

Fig. 15.2 A typical SCADA system consists in several remote terminal units (RTUs) connected to a central controller the operator can interact with through a GUI.

With the development of communication interfaces, distributed control systems (DCS) have emerged. They have many similarities with SCADAs but the intelligence is not centralized but rather distributed in several subsystems called intelligent electronic devices (IED). IEDs are usually implemented with microcontrollers and are similar to RTUs. However, they have extended capabilities such as the ability to autonomously issue control commands; an IED can command a circuit breaker by detecting an abnormal voltage, current or frequency, for example.

SCADAs only provide communication interfaces between a control system and physical end-points. Further control capabilities can be integrated, including (Fig. 15.3):

- Distribution management systems (DMS) achieve or facilitate numerous functionalities, such as substation automation, condition monitoring, fault location, voltage control and load flow calculations.
- Energy management systems (EMS) include computer-aided tools that enable controlling and scheduling the operation of a power system under given constraints (e.g. at a minimal cost), without affecting the capability to meet their technical requirements and quality standards.
- Outage management systems (OMS) are used to assist operators for system restoration after a failure.
- Geographical information systems (GIS) contain geographically referenced data, for visualizing events in a distribution grid.
- Additional systems for network analysis, demand response management, business management, billing, customer information, etc.

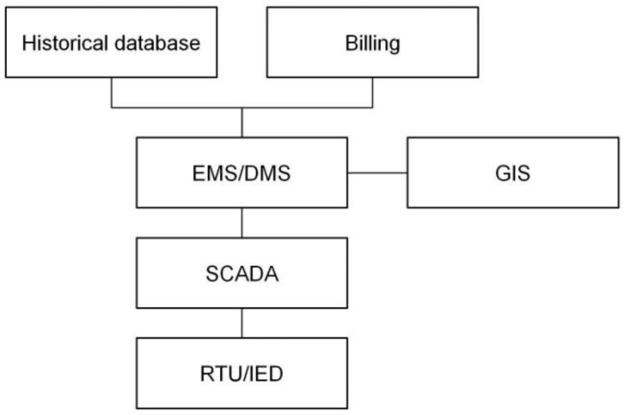

Fig. 15.3 An integrated control system includes several interconnected systems such as SCADA, EMS/DMS, etc.

15.2.3 Distributed Control Systems

The SCADA and associated control systems described in the previous section have been utilized for monitoring and controlling power systems relying on a traditional centralized structure, with a limited number of large power plants used for generation. However, distribution and transmission systems are under transformation towards further incorporation of distributed energy resources, and monitoring and control systems must therefore undertake changes such as:

- Distributing decision-making, by moving from centralized analysis and decision systems to distributed approaches, where several components (instead of one) are able to reach a goal through cooperative tasks. This process begun with the deployment of IEDs, but further features are required due to the evolution of the grid and associated communication infrastructures.
- Automating complex decision processes, such as scheduling and planning, today often conducted by human operators. There is a need for advanced computational resources, with complex decision-making and decision-support algorithms.

Multi-agent systems (MAS) have been showing a great promise to fulfill those needs. The following section explains their importance and how MAS can contribute in making control systems deeply distributed, further autonomous and more intelligent.

15.3 MAS for Power Systems

Although MAS have been used for over two decades, their application in power systems have appeared in the last decade. This section explain what are MAS and why they are so important as a distributed paradigm for strategies used in modern power systems control.

15.3.1 Principles of MAS

Multi-agent systems support a framework of modeling and control of multiple structures that can be decomposed into several interacting entities. Formal definitions for MAS have been proposed by Wooldridge and Weiss [1] and Ferber [2]. The following definition provides a simple overview of the MAS concept:

A multi-agent system is a system composed of a collection of autonomous and interacting entities called agents, evolving in an environment where they can autonomously perceive and act to satisfy their needs and objectives (Fig. 15.4).

Based on such definition, many existing systems from various domains can be classified as MAS, such as:

- A society or a group, in which agents are people, and can communicate with each other, cooperate, compete, and so on.
- A network of computers, where agents are software algorithms, interacting with each other by exchanging messages and data, for example to solve a problem faster than a single computer.
- Robots of a production line, where they need to cooperate and coordinate themselves to perform a given task.
- A power system, where the components of the grid are agents and must interact with each other to serve consumers reliably and at least cost.

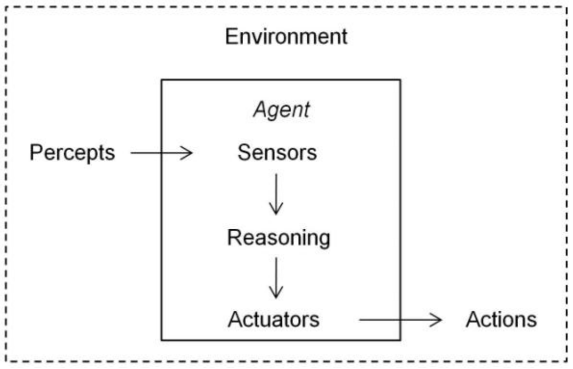

Fig. 15.4 An agent perceives its environment through sensors and acts on it using its actuators.

There are multiple definitions for what can be defined as “the environment” of an agent. It can be defined as to all external entities and resources the MAS can interact with. From a power system point-of-view, the environment of an agent can be considered as everything but itself. The agent can be (geographically) situated in this environment, and other agents may be part of it. There are various means of interacting with their environment, through communication, by controlling actuators, or by evaluating a particular index of performance, for instance. Agents receive data from their environment, called percepts, and take decisions on the basis of current and possibly past percepts.

Agents can exhibit different behaviors and properties which give them a certain degree of autonomy with various degrees of intelligence. The most common ones are:

- *Reactive agents*: they only show some simple reactions to excitations (stimuli). They are useful when fast response times are needed. Their representation of the world, i.e. the environment, is minimal. But by interacting with each other, such agents can together lead to important results, difficult to achieve if the system had been modeled as a single agent. Protection systems for utility distribution can be considered as reactive agents.
- *Cognitive or intelligent agents* : they have extended intellectual capabilities and can use their resources and skills to reach their goals. Those agents can have beliefs, desires and intentions, as in the BDI model (Beliefs, Desires, Intentions) [3]. Fully automated building energy management systems, and gas turbine control systems are examples of such agents.

- *Learning agents* : they can gain knowledge, for example by analyzing the results of their actions and usually have a much better knowledge of the environment, which is required to take complex decisions. A building energy management system with the ability to learn the habits of its occupants and take decisions accordingly would be an example.

Additional explanations are given in section 15.4.5.

Therefore, MAS have the following properties:

- They are inherently *distributed*, due the separation between agents and their environment. An agent may not be handicapped by a change of environment or else they will react differently depending on their surroundings. Agents do not need to have a global knowledge of the whole environment, since their perception is local.
- They are *pro-active* and follow their own objectives, or the ones prescribed for the whole system. These objectives can be reached by using their own resources and skills, autonomously, or by interacting with other agents.
- They have a *social* ability, i.e., by interacting with each other and their environment, agents can form groups and cooperate or compete, depending on their goals. Together, agents are thus a form of distributed or collective intelligence.

Therefore, the methodology of formulating a MAS-based modeling and control system design requires formalizing how the agents coordinate themselves, cooperate, take decisions, plan their actions, and build their representations of the environment.

15.3.2 Relevance of MAS for Power Systems

In order to discuss the advantages of MAS based control versus classical methodologies for applications in power systems, it is important to understand how the power grid evolved over the last decades. Classical control methods implemented in SCADAs are fully functional in today's grid. But the contemporary developments towards the future smart grid will require to integrate millions of devices such as distributed storage, intelligent loads and distributed energy resources. Control systems will then have to operate efficiently on a large scale system, despite very disperse faults that may occur. In order to analyze practical advantages of MAS for tackling these conditions, the following properties (distributed, pro-active and social) will be discussed in more details.

Distributed Architecture

MAS are distributed by design with three main attributes: (i) local knowledge, (ii) flexible interactions, and (iii) bottom-up control approach. Local knowledge means that agents' view of the environment are local, and as a consequence, their knowledge is limited to only what they can or need to know. The perception of agents can be limited to their neighbors, which enables reducing data transfer. For example, for a microgrid, an agent for a distributed generator does not need to receive information about a small load, which can be several miles away. Therefore, a distributed MAS architecture contributes to a scalable distribution grid.

A flexible designed MAS incorporates plug-and-play, robust and fault-tolerant procedures when required by changes in the environment. For example, if a generator or load agent is disconnected, i.e., turned on or off (independently of whether it has been scheduled or not) or loses communication, the MAS will acknowledge this modification and adapt to it. It will take it into account when taking decisions towards reaching its objectives (e.g., maintaining the system stable) after the event has been detected. This is property is an evolution when compared to most analytical control methods, where all possible events, changes and faulty conditions have to be predicted when designing the control system in order to command the system for corrective reaction. In addition, a flexible MAS can add or remove new functionalities without requiring to completely redesigning the system, which could help lower development and maintenance costs. This characteristic is similar to how computer systems use their plug-and-play internal operations. The operation of electric vehicles with the distribution grid for charging could for example benefit from this property (Fig. 15.5).

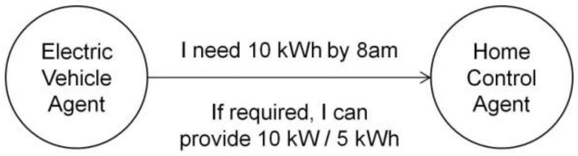

Fig. 15.5 When an agent joins an existing MAS, it can announce its name and services to the other agents so that they can interact with him. This figure illustrates an example with an electric vehicle.

The previous two features enable the third one, i.e., a bottom-up control approach. This feature is well-suited for complex and distributed problems. Agents can operate autonomously, and cooperate or compete with each other if required. The complexity of the control system can be reduced by distributing tasks among communicating agents. This property could play an important role in a MAS designed for a smart grid with a high penetration of distributed energy resources. The grid would be divided into many microgrids containing local generators, loads and storage devices (Fig. 15.6). Intermediate layers, consisting of groups of microgrids for example, could also be added. When compared to control methodologies applied in current power systems, this would correspond to a bottom-up approach, where decisions would be taken locally, in a decentralized way.

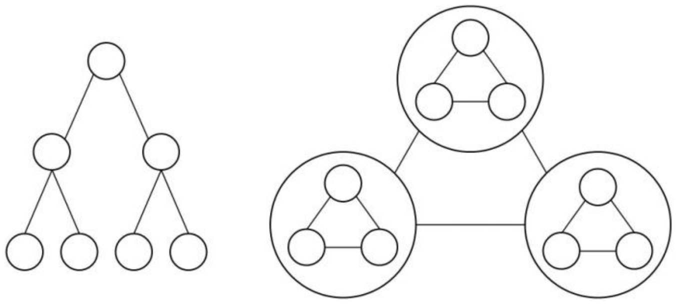

Fig. 15.6 One of the expected evolutions of future power grids is a change from a centralized and radial architecture (left) to a decentralized network of microgrids (right). Circles represent energy sources and loads, but also the corresponding agents.

Proactivity

Proactive agents have goals which can be local and/or global. A single agent usually has local goals while a group of agents may have global goals. For example, maintaining a steady voltage is mainly a local goal for a power source, while maintaining balance between generation and supply is a global objective and cannot be reached by a single agent, hence requiring cooperation. Such pro-activity might be enabled by autonomous intelligence with information based on knowledge about the environment (e.g., the grid), and when appropriate with further information by asking other agents, and knowledge of required actions requested by other agents through communication that may help achieving global goals. Agents can then take decisions based on such on-line knowledge and their goals, plan actions to perform, and finally execute them for achieving the required actions.

Figures 15.7 and 15.8 illustrate an example where a storage manager schedules a distribution procedure: if the system knows (based on forecasting) that demand is going to peak and conventional generation sources will not be sufficient to match it, the battery agent may take preemptive actions by charging the battery to its maximum state-of-charge. This would enable the system to meet future demand during such peak. Other examples could be procedures to start, synchronize and reconnect a turbine to the grid, and planning of required reactive power for such connection.

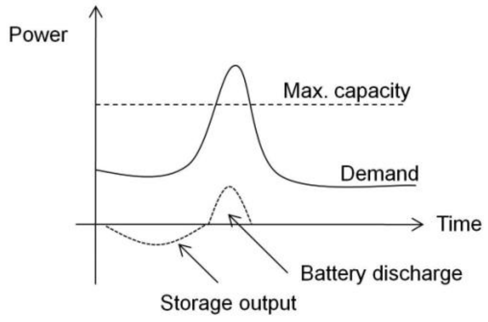

Fig. 15.7 In this fictitious example, the total generation capacity is not sufficient to meet demand during a peak. Storage enables to meet this demand, even during the peak, but requires charging before discharging.

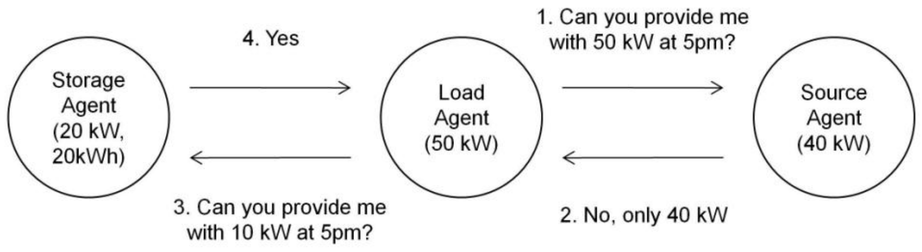
Storage Agent (20 kW, 20kWh)
Load Agent (50 kW)
Source Agent (40 kW)
1. Can you provide me with 50 kW at 5pm?
2. No, only 40 kW
3. Can you provide me with 10 kW at 5pm?
4. Yes

Fig. 15.8 A simplified example of how three agents could interact to find a solution to an insufficient capacity problem.

Social Behavior

Agents need to have a social behavior compatible with other fellow agents, expressed under various forms. Their social organization can vary from one system to another and with time, as well as the way they interact with each other and take decisions. In other words, agents can coordinate themselves and cooperate for reaching goals that may not be reachable by a single agent. Agents can influence the actions of others or act as interfaces for negotiations, requests and contracts, as described in section 15.4.8.

Continuing the previous example on peak demand with storage, before the MAS decides that the battery should absorb the peak, a "discussion" with other agents (such as power system brokers) may happen about whether the load can be taken by another source and maybe a better solution would arise from economics and technical points-of-view. Another scenario is when a small dc microgrid has a bus voltage maintained by a battery. If such battery tends to run towards a low state-of-charge, the battery agent can ask other agents to replace it in order to control the bus voltage.

Practical Implications

MAS also enable to specify communication aspects, that are not commonly considered in power systems studies, and are an important feature of smart grids. This property enables MAS-based prototype systems to be closer to implementation, as the system is already distributed and is easier to deploy. Defining what each system entity (i.e., agent) does also facilitates specifying the required hardware.

15.3.3 Applications of MAS for Power System Control

Smart-grid control requires flexibility and extensibility, and those are inherent features of MAS. Fig. 15.9 shows how MAS enable communication and decision-making for power systems, where agents are associated to sensors, actuators, operators and other physical or virtual entities. MAS have been showing promises as an efficient control framework for developing the technology of smart-grids, including voltage/VAR control, restoration, energy management, monitoring and fault analysis (several examples are described in section 15.5). A comprehensive state-of-the-art review can be found in McArthur et al. [4,5].

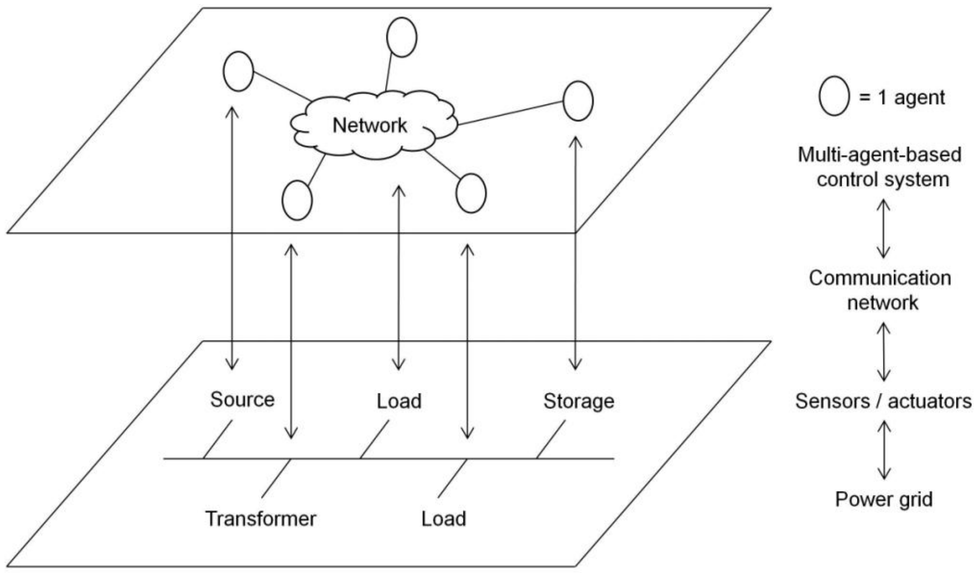

Fig. 15.9 MAS are well suited for monitoring and controlling power grids, and especially for smart grids where communication and distributed assets are intensively present.

15.4 MAS Design

This section gives the fundamentals on how MAS can be applied to power systems. It is described how existing standards, methodologies, languages, tools, architectures, and choices to make can support design specifications such as when and how should which agents interact (cooperate and/or compete) to successfully meet their design objectives. The following items are discussed:

- MAS design methodologies and their characteristics and protocols.
- Standards that MAS should adhere for power systems control.
- Communication protocols between agents.
- Types of services in agent platforms.
- Reactive and intelligent agents structures.
- Use of learning techniques for MAS.
- MAS for planning actions.
- MAS development platforms to facilitate MAS design and prototyping.

15.4.1 Development Methodologies

With a growing number of applications of MAS in various disciplinary fields, several methodologies have been created to enable developers to follow a formal process when designing MAS. These methodologies describe the tasks and activities that take place during the development process.

Similarly to most standard software development methodologies, or life cycles, most MAS development methodologies follow the same steps, sometimes referred to as the waterfall model: requirements specification, architecture definition and design, implementation, testing/verification, deployment and maintenance (Fig. 15.10). However, due to the distributed nature of MAS, such methods require some adaptations, which are directly related to how agents are structured and how they interact with each other. Most MAS methodologies rely on similar basic steps and abstractions.

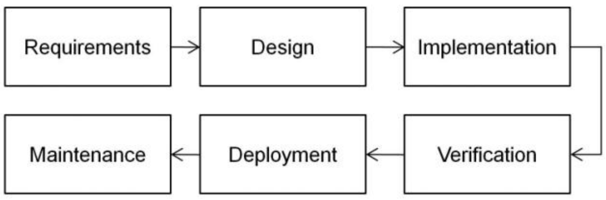

Fig. 15.10 The waterfall model of software development

Two abstraction levels are distinguished for specifying the system and its objectives in order to enable designing MAS:

- The individual agent level, for defining the autonomy, pro-activity, and cognitive, deliberation and control abilities of each agent.
- The multi-agent level is split into the four following processes:
  - ✓ The environment, to identify what the MAS can influence.

- ✓ The roles and interactions of agents.
- ✓ The organizational rules, to identify how roles and protocols depend on and interact with each other.
- ✓ The organizational structures, to identify the topology of the MAS.

Several development methodologies can be used for MAS design, such a GAIA [7]. Table 15.1 contains a summary of the analysis and design steps of this methodology. Other possibilities include SODA [8], MaSE [9], MESSAGE [10], TROPOS [11] or ADELFE [12]. MAS-CommonKADS [13] and DESIRE [14] are alternatives based on knowledge engineering. Comparisons of those methodologies can be found in [15,16,17].

Table 15.1 A summary of the main steps of the GAIA methodology

1. Analysis	*2. Architectural design*	*3. Detailed design*
1.1 Division into sub-organizations	2.1 Organizational structure	3.1 Agent models
1.2 Environmental model	2.2 Final role models	3.2 Service models
1.3 Initial role models	2.3 Final interaction models	
1.4 Initial interaction models		
1.5 Organization rules		

These methodologies use top-down approaches, but when designing a MAS with an expected emergent behavior, a bottom-up approach (where specific agent-level capabilities can result in coherent multi-agent-level behaviors), should be preferred [18]. It has been observed that, at the time of writing, such methodologies are rarely used in designing MAS for power systems, and that standards are rarely considered.

15.4.2 Standards

The on-going development of the "Internet of Things" is expected to enable all kinds of products (including energy sources and loads) to be interconnected. These products will need to use a common language to interact coherently. Interoperability between products from different designers and vendors is an important concern if the designed system needs to communicate with other devices to be used or maintained on a regular basis, which is usually the case for power systems applications. The benefits of using standards are important: they enable lower development costs, faster development and better flexibility, interoperability and integration with existing systems. As a consequence, several standards and specifications have been developed by various groups such as FIPA [19] and IEC [20].

Three main categories of standards are important for considering MAS applied to power system control: (i) FIPA standards for agents, (ii) standards for communication between systems, and (iii) data structuring for power system applications.

The first set of standards were designed by the Foundation for Intelligent Physical Agents (FIPA), an IEEE Computer Society organization. It was specifically developed for interoperability of agents and MAS and contains five main sections. The first one deals with particular application domains, such as entertainment or travel assistance, while the second section refers to the abstract architecture of MAS, i.e. the entities required to build agent services and environment. The third part describes agent communication with interaction protocols, communicative acts and content languages. The fourth section deals with agent management specifications and the fifth part presents the agent message transport protocol, used for transport and representation of messages. Two of these specifications are widely adopted and will be described in detail in the subsequent subsections: Agent Management (SC00023), and Agent Communication Language (ACL) (SC00061).

The second category of standards deals with how systems such as substations and other power systems should communicate. As communication is one of the most important tools of MAS, they need to be integrated in the design process of any industrial MAS. The authors selected three main standards to be discussed in this chapter: IEC 61850, IEC 60870 and DNP3.

- The International Electrotechnical Committee (IEC) has several technical committees, such as the TC57 [21], that is responsible for the development of standards for information exchange in power systems, including SCADA, EMS and distribution. One of the main standards published by this committee is IEC 61850 [22], originally a standard for the design of electrical substation automation, with a focus on communication. As adaptations and extensions of the standard were also published for particular applications such as hydroelectric power plants, distributed energy resources or wind turbines, the standard now applies to a variety of power systems, including energy storage and demand response. IEC 61850 includes several communication protocols based on Ethernet.
- IEC 60870 is another IEC standard for SCADAs, which has six parts. Its fifth and sixth parts define the communication profiles and protocols for interaction between systems.
- The distributed network protocol DNP3 [23] is a set of communication protocols for process automation, commonly used by utilities in SCADA systems. Its main focus is on reliable communications between master stations and IEDs/RTUs. DNP3 was adopted as an IEEE standard (1815-2010).

The third and last set of standards defines how data should be structured, mainly for SCADA applications. These specifications impact how agents may structure their representation of the environment, and how they interact with each other.

- In addition to its specifications for communication, IEC 61850 also defines naming and object modeling schemes. It specifies a substation configuration description language (SCL) for the configuration of substations, based on XML, as well as a representation of modeled data and communication services.
- The Common Information Model (CIM) [24,25] was developed by the electric power industry for generation, transmission and distribution. Its aim is to define how application software can exchange information about the configuration and status of an electrical network. This is achieved through a common vocabulary and ontology. The CIM is based on other standards, such as IEC 61970 for interfaces with energy management systems and IEC 61968 for interfacing the main applications for electrical distribution in a utility. Although CIM and SCL share some similarities, they are distinct standards and should not be mixed. A distinction is the focus of SCL on substations, whereas CIM is applied to various other systems.
- MultiSpeak [26] is another standard designed for interfacing software applications of utilities. It defines common data semantics (in XML form), the message structure for data exchange, and the messages required to support specific business process steps. Here too, although CIM and MultiSpeak have similarities, the latter is rather oriented toward distribution and US-based electric cooperatives.

Due to the multiplicity of these standards, there is a lot of harmonization work underway between MultiSpeak, IEC 61968 and IEC 61970, and CIM and SCL.

Using a MAS for operating a power system requires combining these three types of standards. Achieving a coherent system respecting these specifications is difficult and has been the subject of some works, as in[27,28]. Moreover, there are additional standards for other topics, such as security (IEC 62351) and automatic meter reading (DLMS/IEC 62056).

15.4.3 Communication, Languages and Ontologies

A common language and vocabulary is required for agents to communicate. Therefore languages such as ACL (Agent Communication Language) and KQML (Knowledge Query and Manipulation Language) were created and can be used for MAS development. ACL is a FIPA specification [29] and is usually preferred. Similarly to protocols such as TCP, each message is given several attributes, including the content of the message, and information about the participants and the ownership of the conversation. The structure of an ACL message should contain the following parameters:

- A parameter defining the type of communication, called "performative"; this field indicates whether the message is a request, a reply, an information, etc.
- Participants in the conversation, with information on the sender, receiver(s), and reply-to fields, including the names of the corresponding agents.
- The content of the message.
- A description of the content, with the used language, encoding and vocabulary, called "ontology".
- Conversation control parameters, such as a conversation identifier and protocol.

Table 15.2 shows the structure of an ACL message. The performative is the only mandatory parameter in the specification, but others should also be included when necessary.

Table 15.2 Sample ACL message fields

Field	*Value*	*Description*
(inform		Performative
:sender	agent1	Sender name
:receiver	agent2	Receiver name
:content	How are you?	Content of the message
:in-reply-to	hello-round-0	Other parameters
:reply-with	hello-round-1	
:language	sl	
:ontology	hello-ontology	
)		

For complex conversations between heterogeneous agents, ontologies may be needed to define the vocabulary agents use in their conversations. As for humans, using a common language may indeed not be sufficient. An ontology is a formal representation of knowledge, under the form of a set of concepts and of relationships between these concepts. FIPA-SL is an ontology standard adopted by FIPA that is implemented in several MAS development tools (see section 15.4.10). Its includes three types of elements:

- Concepts can include all lot of different elements from physical components to data sets. Concept examples can include components, data types, etc.
- Agent actions are a particular type of concept corresponding to communication acts. For example, the action *connect()* would require the subject of the action to connect to the rest of the system after having been isolated.
- Predicates specify relationships between concepts which result in binary values (true or false). An example could be the predicate *isStorageUnitCharged()*, to determine whether a storage unit is fully charged or not.

In large systems, where several subsystems have to interact, may arise the need for multiple ontologies. An ontology agent can then be used to handle these multiple ontologies, or an upper ontologies may be developed to serve as a general models for designing more specific ontologies. As described in McArthur et al. [5], standards specifying data models, such as CIM, can be used as a basis to build ontologies. A CIM-based ontology was for example made available by the IEEE PES Multi-Agent Systems Working Group [30].

15.4.4 Agent Management

FIPA specification SC00023 [31] defines two levels in how agents should be managed, exist and operate. The first level, called "agent level", corresponds to each agent itself, while the second relates to groups of agents and how they interact with each other, called the "MAS level". At the agent level, the specification defines the life cycle of the agent, and at the MAS level, it proposes agent management services and a message transport system. These services are essential in enabling the MAS to operate in a distributed and flexible manner.

The first level defines how agents have their life cycles, from their creation to their end. During their lifetime, agents can be in five possible states:

- Initiated, just after their creation. In this state, the agent executes a series of instructions, defined by the user and run once as an initialization procedure.
- Active, that is the "normal" state, in which the agent operates.
- Waiting, when it is pooling for an external event and has not been woken up.
- Suspended, when it has been halted from the active state.
- In transit, when the agent is physically moving from one agent platform to another (e.g., from a computer to another).

Fig. 15.11 shows a finite state automaton representing all the possible states in an agent.

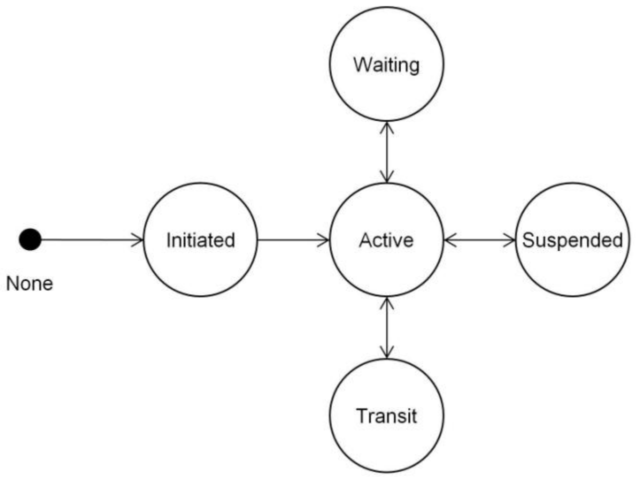

Fig. 15.11 The life cycle of an agent starts with its creation and its transition to the initiated state. It can then switch to four other states, the most common one being the active mode. Destroying the agent means it does not exist anymore [31].

At the MAS level, each agent platform (AP) contains several entities and at least one agent. A single platform can consist of agents located in different physical locations, such as on remote computers. It should implement the agent management reference model (Fig. 15.12), which defines service entities including:

- The agent itself, which communicates with other agents using the ACL language and may have access to external software.
- A directory facilitator (DF) that provides a yellow-pages service to agents, i.e., a list of agents with their respective capabilities. Each agent can register its services in the DF and then query it for knowing if other agents have a certain service registered. Implementing a DF is optional, and multiple DFs can exist in a single AP.
- A unique agent management system (AMS), which supervises the AP and maintains a list of all agents and their addresses in the AP. This functionality is similar to a white-pages service, i.e., a list of agents and their name and address, to which each agent must register.
- A message transport system (MTS), which enables messages to be transported from one agent to the other. A message transport protocol (MTP) is used for physically transferring messages between agents on possibly different platforms; all messages exchanged between platforms go through the MTP.

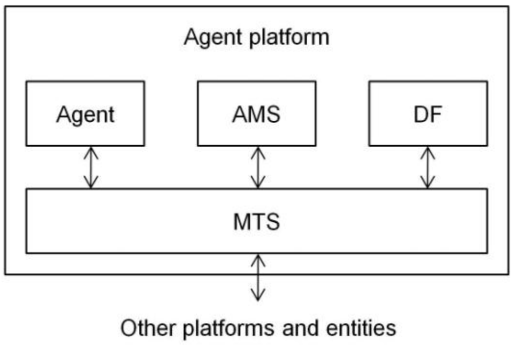

Fig. 15.12 FIPA's agent management reference model [31]

For enabling the previous services to operate properly, each agent is assigned a unique agent identifier (AID). Each AID is composed of three parameters:

- A unique name, usually consisting of a local name and the address of the AP.
- An address list, to which the messages should be delivered.
- A resolver, used by the AMS for resolving the transport address of the agent.

15.4.5 Agent Anatomies

As described by Russell and Norvig [32], the internal architecture (or anatomy) of an agent defines its behavior, i.e., there is a function mapping every sequence of percepts (what the agent perceives) into actions. This function can be simple, as in reactive agents, or quite complex as in cognitive agents. An architecture such as subsumption may be used for reactive agents [33]. Cognitive agents may be based on the Soar architecture [34], while the work of Albus on the reference model architecture [35] is applicable for hybrid types. There are four main classes of agents classified in accordance to their level of autonomy, the way percepts are used, and how they can be modeled:

- *Reflex agents* perform simple actions based on current percepts (Fig. 15.13) and their behavior is based on an action selection module that receives percepts from the environment and consults a database of condition-action rules, similar to *if-then* rules, to take a decision. Such agents can be useful when fast response times are needed, e.g., for protection. Their representation of the world (the environment) is minimal, but they may support emergent behaviors. Emergence occurs when new characteristics appear at a certain level of complexity.

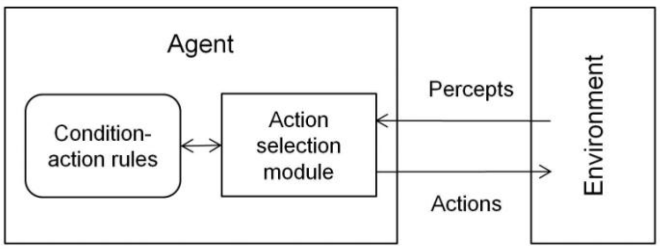

Fig. 15.13 Structure of a simple reflex agent

- *Goal-based agents* are directed by goals set *a priori* by the user. They have an internal representation of their environment and can memorize previous percepts so as to take more elaborate decisions. More precisely, once a percept is received by the agent, the memory manager stores the information in the percept memory. A sequence of percepts is then built in order to be used subsequently by the action selection module to select suitable actions in order to reach given goals (Fig. 15.14).

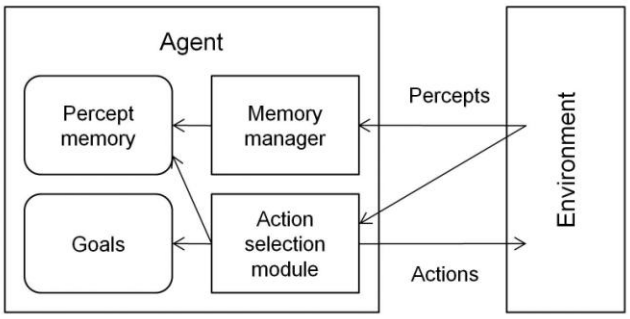

Fig. 15.14 Structure of a goal-based agent

The BDI agent model [3] is a type of goal-based agent that enables a designer to build agents having beliefs about the world, with desires, i.e., objectives and goals, plus intentions, i.e., plans. The BDI architecture enables a separation between plan (actions) selection and execution. Such agents can therefore be used for planning, which consists in finding a sequence of actions in order to reach a given goal. The concept of planning will be further explained in section 15.4.9.

- *Utility-based agents* use a performance measurement index, commonly called utility function, in order to evaluate their behavior. Suppose a goal-based agent has to search for a sequence of towns that have to be visited. Some paths may be shorter, safer or cheaper than others. A goal-based agent will select any path among the possible ones. On the other hand, an utility-based agent can select a specific path that optimizes a given utility function, taking into account some compromise. Fig. 15.15 summarizes the processes used by a utility-based agent. A utility-based agent that always optimizes its utility is called a *rational agent*, as it behaves in an efficient manner, given its prior knowledge of the environment. An agent running an economic dispatch algorithm is an example.

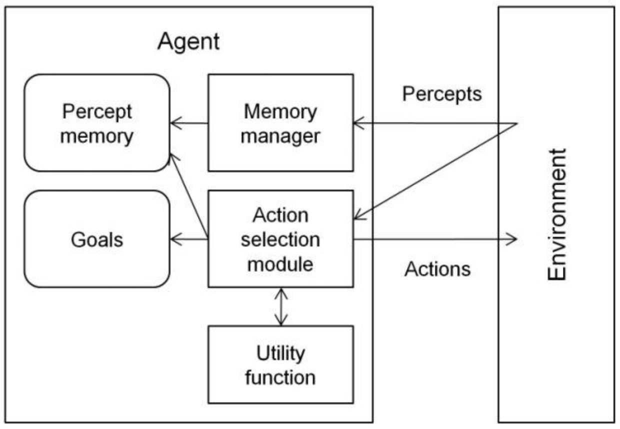

Fig. 15.15 Structure of a utility-based agent

- *Learning agents* belong to one of the previous classes and in addition can learn to perform a given task more effectively. They are able to modify the function that codes their behavior while interacting with their environment, so as to be more confident at performing a given task. An agent running a load forecasting algorithm would be a typical example. These types of agents are described with more details in the next section.

15.4.6 Learning agents

Since the beginning of artificial intelligence in the 1950's, scientists have been developing computer programs with the purpose of *learning* how to perform a specific task. Machine learning became a very exciting branch of artificial intelligence, for it aims at designing machines that can improve their performance without having to actually program them explicitly.

Instead of defining algorithms as functions that map inputs into correct outputs, with the risk of forgetting some inputs when defining the function, machine learning advocates the design of algorithms that enable a machine to learn a function using empirical data. Such data can be either gathered prior to the learning phase of the task at hand, or collected by the machine as it interacts with its environment, i.e., within a network or a physical or virtual world.

Machine learning is traditionally divided into three main branches:

- *Supervised learning* deals with problems of prediction or classification. It consists in determining the underlying function that has generated a set of data. This function is assumed to map inputs to desired outputs. Artificial neural networks, support vector machines and decision-trees are some of the techniques widely used in supervised learning. An application of supervised learning in power systems is for load forecasting, where historical and weather data is used to predict the load.

- *Unsupervised learning* consists in identifying some regularities in the given data, in order to constitute classes. K-means, mixture models, principal component analysis, independent component analysis or some artificial neural networks like the self-organizing map are some of the most widely used approaches for solving unsupervised learning problems. Such techniques can be used for blind source separation (that is the separation of a set of signals from a set of mixed signals), text clustering (grouping documents that share similar contents) and dimensionality reduction (which consists in reducing the quantity of data under consideration when taking a decision). In power systems, an algorithm analyzing large fault data sets to detect the origin of an issue could use such techniques.

- *Reinforcement learning* tackles sequential decision-making problems, that is problems where a rational agent has to determine the best sequence of actions that enables it to efficiently perform a given task. For example, a chess program willing to win should perform the best sequence of moves at every encountered game configuration. A reinforcement learning agent usually interacts with its environment using three kinds of signals: percept, action and reward signals. Rewards are typically generated by a reward function defined by the system designer. This reward function must formalize what the agent has to do, i.e., its task, and not how it should do it. An agent bidding on power markets could integrate such capability to improve its strategy over time.

Each time the agent receives a percept, its action selection module consults a memory constituted by information about its interactions with its environment and then selects an action that is performed by the agent in the environment. A new percept and reward are sent to the memory manager of the agent to update its interaction memory (Fig. 15.16). These steps of interaction and learning of the agent are continuously repeated. The figure below assumes the agent has a global perception of its environment.

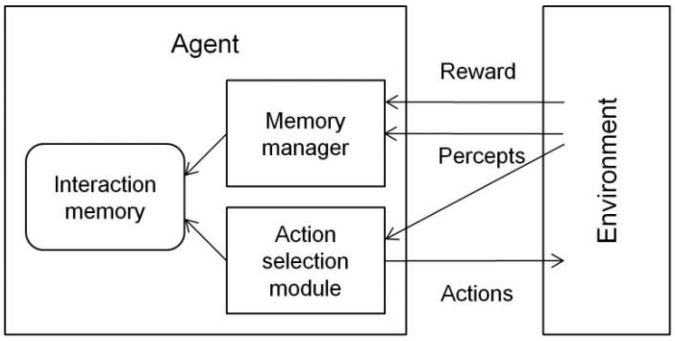

Fig. 15.16 Structure of a simple learning agent

Machine learning techniques have been applied successfully to solve complex problems in various applications. Their success can be explained by the fact that they can intrinsically handle partial observability and uncertainties coming from different sources, for example due to measurements by imperfect sensors. Several free software suites are available for experimentation on a variety of machine learning algorithms such as RapidMiner [36], KNIME [37], Weka [38] or Shogun toolbox [39].

In the fields of control and energy management, machine learning techniques can be used advantageously for carrying out different important tasks. Supervised learning can be applied to handle renewable generation forecasting [40,41] as well as demand forecasting [42,43]. Renewable generation from photovoltaic panels and wind generators can be predicted from data mapping time-stamped meteorological conditions (like wind speed, solar irradiation or humidity) to the optimized corresponding generation of different types of renewable devices. Supervised learning techniques take advantage of these data to determine underlying functions (one for each possible renewable energy source) that can precisely predict generation from information about meteorological conditions. Similarly, demand forecasting can be handled by using supervised learning techniques that uses time-stamped data extracted from historical and meteorological measurement databases.

Reinforcement learning techniques can for example be used to reliably damp electromechanical oscillations in FACTS (flexible AC transmission systems) [44]. Reinforcement learning could also be valuable for supporting the behavior control of energy trading agents. Such techniques can be based on nonlinear, stochastic and non stationary assumptions, so that agents can continuously learn to act rationally in environments whose dynamics or the operating conditions can possibly change. Moreover, they could be used in combination with some classical approaches and enable them to operate faster by suggesting solutions based on historical data. For example, principal component analysis (PCA) might be used from an initial set of information to extract the important features on which some agents could rely in order to learn to act intelligently.

15.4.7 Organizational Topologies

The collection of roles, authority relationships, data flow, resource allocation and coordination patterns that guide the behaviors of all agents is defined as an organizational topology. The major topologies used in MAS include hierarchies, holarchies, coalitions, teams, congregations, societies, federations, markets and matrix organizations [45]. Each has its own strengths and weaknesses, and some topologies are more appropriate than others depending on the application:

- A *hierarchy* (Fig. 15.17) is the earliest and the most widely used topology, in which agents are arranged in a tree-like structure. Agents higher in the tree have a more global view than agents below them. Lower-level agents transmit the information perceived locally to higher-level agents, which provide directions to those below them, on the basis of a more complete amount of information. This topology is typically used in most current control systems.

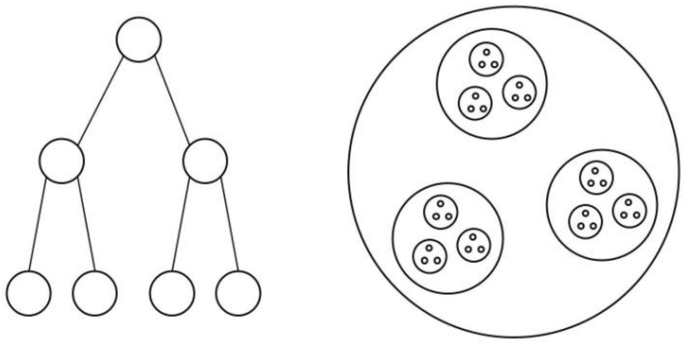

Fig. 15.17 A hierarchy (left) and an holarchy (right)

- *Holarchies* consist of agents (called holons) that are constituted by several entities and are at same time part of a larger entity. Biological species, individuals, cells and atoms can each be viewed as holons sharing this dual characteristic. Hierarchies and holarchies are more easily applied to problems where goals can be decomposed into tasks that can be assigned to those agents.
- C*oalitions* are dynamic and short-lived, and emerge as soon as a goal has to be fulfilled by a subset of an agent population. The coalition is destroyed when the constituent agents have managed to perform the task. The structure of a coalition is typically flat, although there can exist a leading agent that represents the coalition as a whole. In a coalition, agents are "selfish," i.e., they try to maximize their own profit. Such topology may for example be used for power restoration after a blackout.
- T*eams* are an "altruist" type of organization, as opposed to coalitions; they attempt to maximize the utility of the whole team, and coordinate their actions in order to efficiently fulfill a common task. Team agents have an explicit representation of the shared tasks and know the means by which cooperation should progress.
- *Congregations* are generally long-lived and are formed from heterogeneous agents that have great interest to get together. Some simple examples of congregations are clubs or academic departments. Congregating agents are expected to be rational, by maximizing their own long-term utility. Congregation are formed if agents want to increase information gain or to decrease commitment failure.
- *Societies* are inherently long-lived and open. Agents living in a society may have different goals, levels of rationality and heterogeneous capabilities. They meet and interact according to social laws (or norms), which dictate how they should coexist. Vehicular traffic laws are an example of social laws that minimize conflicts and encourage efficient solutions.

- *Federations* are arranged such that some agents delegate a part of their autonomy to a single agent that represents the group (Fig. 15.18). Group members interact only with this delegate (also called facilitator, mediator or broker), which acts as an interface between the group and the outside world. The delegate typically receives undirected messages from its group members and sends information to the delegates of other federations. Messages from group members include skill descriptions, task requirements or status information, whereas messages from or to other delegates include task requests or capability notifications. This structure could for example be used to interface several microgrids, where the central controller of each microgrid would be the delegate.

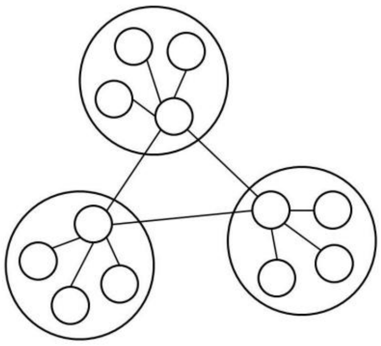

Fig. 15.18 A federation example

- *Marketplaces*, or market-based organizations, enable buyers and sellers to send and receive bids for a common set of items, such as shared resources or tasks. Like for federations, an individual or a group of individuals in a marketplace is responsible for coordinating the actions of other agents. Unlike a federation though, agents within a marketplace are competitive. This topology is already used in power markets, where independent power providers and utilities (among other players) bid to sell and buy power.
- *Matrix* organizations mimic how humans influence one another, i.e., the behavior of an agent or of an agent group may be influenced by multiple centers of authority. How the agent perceives these influences can influence other agents as well. Simple examples of such influences include someone receiving guidance or pressures from its boss, spouse, children or colleagues.

For power systems, no preferred topology is considered better than others. Several parameters may influence the choice of a topology:

- The structure of the real world organization of the system or the power grid.
- Needs for reliability, especially regarding communication.
- Whether decisions require negotiations, for example based on market mechanisms.
- The scale of the system and possible evolutions.
- The necessity to change of topology to reach goals.

Such choice is a tradeoff several parameters including computation, coordination simplicity and organizational rules, plus the simplicity of the architecture itself. An analysis according to these criteria defines the topology of the system and its control regime (the way agents interact, as described in the next section). It should also be noted that the architecture of a

MAS may change over time, depending on what it is trying to achieve and how the environment evolves.

Contemporary control systems are frequently based on hierarchical topologies because they have been the most intuitive to the human mind when designing control systems, especially in single layer applications. For layered architectures, there are other possibilities that enable dividing the global system into several interacting subsystems, such as the reference model architecture [35]. An example is the hierarchy of grid control centers in a country (Fig. 15.19). However, there are several drawbacks; the main one being that they are prone to single-point failures, for example if the central controller or communication fails. Their reliability and resilience are thus limited.

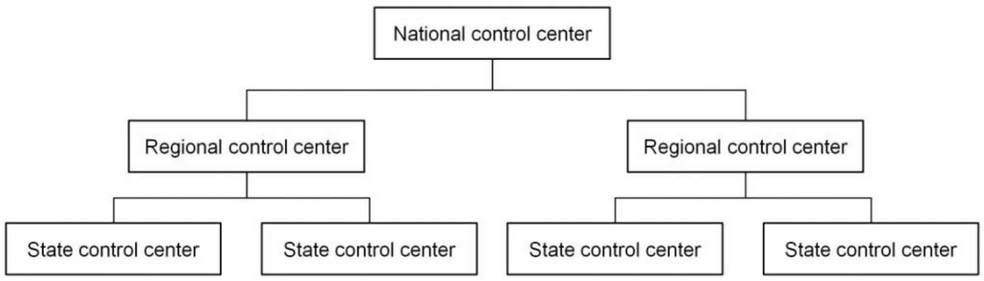

Fig. 15.19 Most of today's control systems are organized in a hierarchical topology based on geographical subdivisions.

More decentralized topologies are theoretically well suited for the needs of modernized, distributed and liberalized power systems. But they are also more complex to design as they impose a change of paradigm. A fully distributed architecture, where all agents would be "equal," would be ideal and would constitute the ultimate level of resilience to communication failures, as there would not be any centralization of data (the need for communication would certainly increase though). Until such topologies become feasible, hybrid ones, such as the ones listed earlier, are thus to privilege. The Internet is an example of an hybrid architecture, where there is no global controller, and decisions are taken at least partially locally, although some nodes are most important than others.

15.4.8 Inter-Agent Interaction

The described topologies define the interaction of one agent with another one, or another entity, but it is still needed to define how they interact. Interactions are paramount notions for defining multi-agent systems [2]. An interaction between agents can only take place if they can act or communicate and if there are situations where they can get together, such as the need to fulfill a common objective. Interactions may be conducted under the form of discussions between at least two agents and can occur in numerous situations: the rescue of an agent by others, a conversation between two agents, the implicit agreement when two agents have to decide which one goes first, the cooperation of several agents to fulfill a common task, and so on.

Interactions are usually required when agents have to satisfy a common objective while taking into account their limited resources and individual skills. Getting together to fulfill a common objective involves that some agents are part of a possibly emerging organization, as described in the previous section. Therefore, every agent organization is the result of these interactions and of the place where they take place. The dynamical characteristics of interactions then implies that new agent organizations are likely to be formed as new objectives have to be satisfied. Agents can for example form coalitions [46] if it helps them reach their goals.

Defining interactions between agents depends on answers to the following questions: What is the nature of their goals? What are their resources? Which skills do they possess to fulfill these goals? Several criteria can be used to classify interactions. These criteria include the nature of the goals pursued by the agents, their relationship to external resources, and their individual skills with respect to the task at hand:

- Typically, agents are engaged in competitive tasks when their goals are contradictory, whereas they cooperate or co-evolve when their individual goals are compatible (that is when the satisfaction of a goal by an agent does not interfere with the possibility of satisfying another goal by another agent). Maintaining the balance between electricity supply and generation is an example of problem where cooperation is essential.
- External resources include all environment elements that agents need to satisfy a goal, such as raw materials, energy, available amounts of space and time, etc. For example, every energy source or power line has a limited capacity. Limited resources can lead agents to conflicts as they will likely need the same resources at the same moment and at the same place than other agents. Such conflictual situations can generally be resolved by the coordination of agent actions.
- The last criterion relates to whether the task can be pursued by a single agent rather than by a group of agents, and if each one of them has the appropriate skills to perform its subtask. An energy source can for example not store energy, and neither does a load.

To enable interoperability, interactions are structured and follow common rules called protocols. Basic and common types of interactions are requests, queries, subscriptions and propositions. An example is the FIPA-Request protocol [47]: an agent can formulate a request, that other agents can accept or refuse (Fig. 15.20).

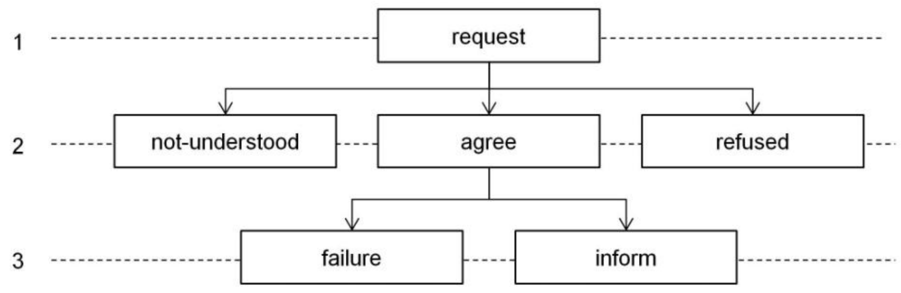

Fig. 15.20 The FIPA-Request protocol enables an agent to formulate a request that other agents may accept or refuse. The corresponding conversation is divided into three consecutive steps.

Those interactions arelimited, and combinations of them are often used instead. There are several types of such complex interaction protocols: contracting, auctions, bargaining, voting and brokering.

- The *contract-net protocol* is an example of contracting, defined in FIPA specification SC00029 [48]. This task-sharing protocol consists in a collection of agents forming a contract network. Two categories of agents are distinguished: the manager, and contractor agents. A typical round (see Table 15.3) starts with a call for proposals sent by the initiator. A deadline can be set to limit the duration spent waiting for answers. Contractors (participants) can then emit proposals (prices, time to execute an action, etc.) or refuse. The initiator evaluates the proposals and selects zero, one or several agents to perform the task to be fulfilled. The selected participants are free to accept or refuse this offer.

Table 15.3 A summary of the conversation between the initiator and the participants (contractors) in a contract-net protocol according to the FIPA specification.

Initiator	*Direction*	*Participants*
Call for proposals	→	
	←	Propose or refuse
Accept or reject proposals	→	
	←	Inform of the end/result of the round or of a failure

- Agents can interact and distribute tasks through *auctions*. Four main types of auctions are commonly used: English, Dutch, sealed first-price or Vickrey, and Walrasian auctions.
    - English auctions are the most common ones. Participants bid openly against each other, with each subsequent bid higher than the previous one. A reservation price (the minimum price) may be set by the auctioneer. The auction ends when no participant is willing to bid further. FIPA defines a specification for English auctions in XC00031 [49].
    - In Dutch auctions, the auctioneer starts with a high price which is lowered until some participant accepts the announced price. This type of auctions is also defined by FIPA in its XC00032 specification [50].
    - With sealed first-price auctions, all bidders simultaneously submit their bids, and the winner is the one with the highest bid. Bidders can only submit a single bid. Vickrey auctions are identical except that the winner pays the second highest submitted price [51].
    - Walrasian auctions are more complex and enable matching supply and demand in a market of perfect competition. A market clearing price is set so that the total demand equals the amount of sold goods, and leads to a general equilibrium [52]. This type of auctions was proposed for use on electricity markets using locational margin pricing [53].
- *Bargaining* is an alternative to auctions for pricing goods, i.e., when prices are not fixed and can be negotiated. The goods can for example be split into several parts and be themselves subject to bargaining. As humans, agents can employ various strategies to reach their goals.
- Interactions can also happen under the form of votes. *Voting protocols*, such as Robert's rules of order (RONR) [54], define procedures for conducting votes between agents. Voting can be used to take decisions when votes are very simple (e.g., yes or

no). Similarly to votes during elections, various rules can be adopted for selecting the winner(s).

- Another interaction type specified by FIPA is *brokering* [55]. An agent can for example request a broker to find other agents who can answer a query; the broker would then relay the answer back to the initiator.

Numerous other interaction protocols, sometimes derived from domains such as game theory, can be employed. Custom interactions can for example be based on Petri nets; consensus theory can be used for reaching a group agreement even in the presence of faults [56,57]; the Shapley value can help finding solutions to cooperative games [58,59]; etc.

15.4.9 Multi-Agent Planning

In order to attain their goals, agents are sometimes required to coordinate themselves to cooperate or compete with each other, and dispatch tasks using interaction protocols. In some cases, agents may be required to plan their actions before executing them. Actions can include rounds of interactions that must happen in a given order, for example if their outcomes are interdependent. As shown in Fig. 15.21, planning enables advanced agent cooperation, whereas negotiations are used for managing competing agents.

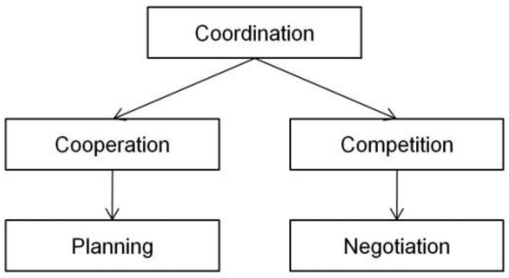

Fig. 15.21 Agents can coordinate themselves in two different manners: by cooperating, and planning is then required, or by competing, requiring negotiation.

Similarly to interactions, planning involves action selection, sequencing and resources handling. The established plans can be action sequences or action trees resulting from policies and strategies defined by the designer or by other agents. Fig. 15.7 provides a simple example: to be able to provide the required power during the demand peak, the storage unit needs to charge before the peak occurs.

Three main elements differentiate multi-agent planning techniques: where the plans are created, when coordination occurs, and how plans are coordinated. As for MAS architectures, planning can be centralized, or partially or fully distributed. In centralized planning, one agent plans for the others. It is potentially optimal and requires communication only before and after planning. On the other hand, in distributed planning, computation time is reduced, privacy is easier to ensure, the system is scalable and distributed control and execution are enabled. For partially distributed planning, only parts of agents' plans are shared. Regarding when coordination occurs, three main approaches exist: before, during and after planning (Fig. 15.22). Coordination can also happen during plan execution, especially when communication is unreliable.

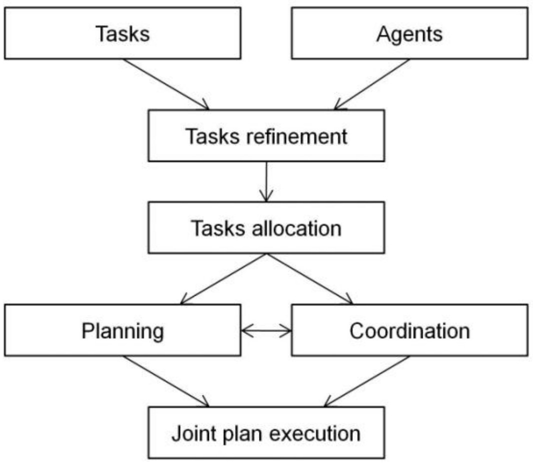

Fig. 15.22 Planning and coordination are complementary in establishing joint plans in MAS. Coordination can happen before, during (as pictured here) or after planning [60,61].

Existing planning techniques include STRIPS, ACT, partial plans, plan space refinement, HTN, or Petri nets [55]. Choosing a planning and coordination technique depends on the behavior of the agents and of the technical environment. Similarly to what was detailed in the previous sections, agents for planning can:

- Be strongly related or independent.
- Be cooperative or self-interested.
- Try to resolve conflicts or to exploit efficiency.
- Have no communication or reliable communication available.
- Have to bear with a low or high incident rate.

The interested reader may refer to [32,60,61,62] for more information on this topic.

15.4.10 Development Platforms

Although it is possible to develop a MAS from scratch, using a dedicated development platform (middleware) is, in most cases, a much simpler solution. Many different toolkits were created over the years, as shown in [63]. These platforms include tools and functionalities that facilitate the development of MAS. Some of them comply with FIPA standards, especially for messaging and agent management. A list of such platforms is available in [64].

In the field of MAS for power system applications, JADE (Java Agent DEvelopment framework [65]) is the most popular. It has extensive documentation, third-party plug-ins (e.g. for mobile devices, for BDI agents [66], etc.) and full FIPA compatibility. Among other existing platforms, some general-purpose examples include AgentBuilder, MadKit and ZEUS. Some others may be of interest for specific users: NetLogo enables beginners to get started with MAS programming, JANUS simplifies holonic agents building, Cougaar enables very large scale simulations, JADEX proposes a framework for developing intelligent agents according to the BDI model, JACK for autonomous agents, etc. Most of these platforms are developed in Java, which enables cross-platform (i.e., operating systems) compatibility, but other languages can be used.

Because of the multiple definitions and applications of MAS, most of these platforms are quite different, and a careful selection needs to be operated. This selection can be done according to the following criteria: documentation availability, compliance with standards, performance [67], scalability, built-in development tools, GUI friendliness, etc.

15.5 Applications of MAS to Power Systems

For all the reasons listed in the previous sections, MAS have been applied to solve several problems in power systems. The following four examples were published in the last decade. Based on the explanations from the previous sections, the reader is encouraged to analyze what choices the designers of the presented MAS have conducted. Other existing applications [4] also constitute interesting concepts.

15.5.1 Coordinated DC Bus Voltage Control

Lagorse et al. proposed a MAS-based coordinated DC bus voltage control scheme, described in [68]. The objective of this algorithm is to automatically maintain the bus voltage at a constant value, which can be considered as the DC equivalent of frequency control in AC networks: as soon as there is a difference between supply and demand, the voltage level (resp. the frequency) moves away from its reference value and compromises system stability.

The DC bus can be the central part of an islanded network where various loads, sources and storage units are connected. In the described test case, the microgrid includes a PV source, a battery, supercapacitors, a grid access point (to import/export energy from/to the distribution system), and an active load, all connected to a common DC bus (see Fig. 15.23).

Each source and storage unit is connected to a power electronic converter controlled by an agent (Fig. 15.24). The agents cooperate to ensure that the bus voltage remains constant, using a token system to select which inverter controls the bus voltage. A secondary objective is to minimize energy imported from the grid.

The inverter that has the token regulates the bus voltage, while the other inverters are current-controlled through PI controllers. The agents coordinate themselves to define two outputs: how each inverter should be controlled (voltage or current), and the corresponding voltage and current references.

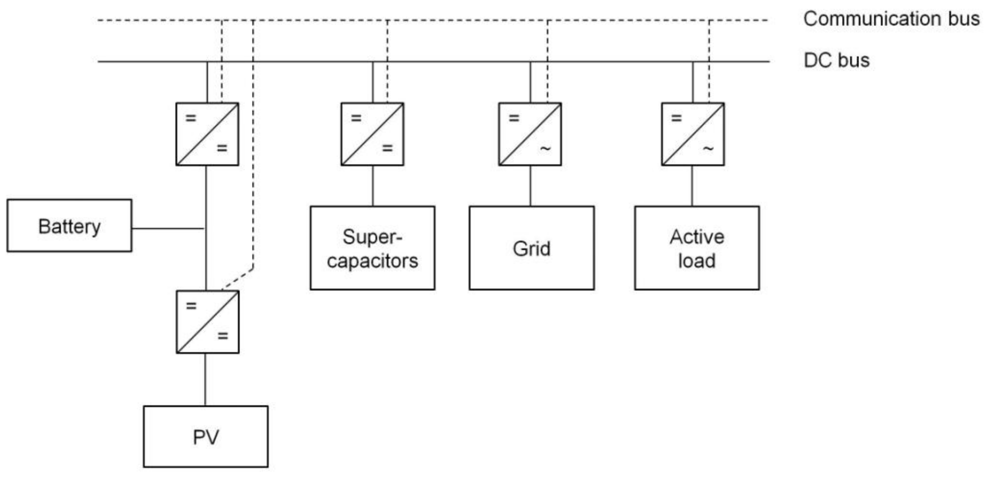

Fig. 15.23 A single-wire schema of the hardware architecture studied in this DC bus voltage control application. Agents directly control inverters by cooperating with each other [68].

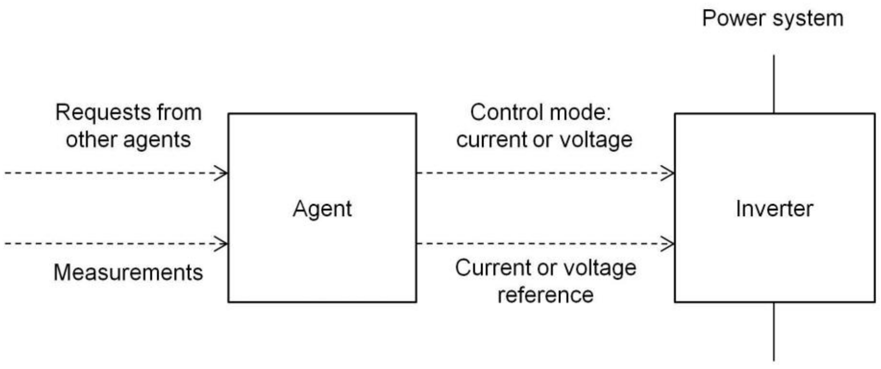
Power system
Requests from other agents
Measurements
Agent
Control mode: current or voltage
Current or voltage reference
Inverter

Fig. 15.24 Each agent receives requests from other agents and decides whether its inverter should be current- or voltage-controlled and with which reference [68].

For example, if a storage unit holds the token and its state-of-charge reaches its lower limit, the unit is not able to keep controlling the bus voltage anymore and needs to give the token to another agent. It then requests another agent to replace it. This request may or may not be accepted. If it is, then the other agent switches its inverter to voltage-controlled mode, while the agent that had the token switches it to current-controlled mode (Table 15.4). If the request is rejected, then the agent can ask another agent connected to the bus to replace it.

Table 15.4 An idealized discussion between two agents when transferring the token to control the voltage of the DC bus [68].

Agent 1	*Token*	*Direction*	*Token*	*Agent 2*
I cannot keep controlling the bus voltage. Can you take the token?	x	→		
	x	←		Yes
Here, take the token		→	x	
		←	x	Thank you

Each agent has a similar internal architecture, based on a finite-state-machine which gives the agent a different behavior depending on whether it has the token or not. Each agent has its own characteristics derived from the type of hardware it is connected to: source, load or storage unit. For example, as grid imports have to be minimized, the grid agent never asks to control the voltage, and can only receive requests. On the contrary, supercapacitors agents have the highest priority to regulate the bus voltage because of their high voltage and very shortresponse-time compared to other sources.

The main interest of this voltage control algorithm resides in its decentralized nature, its flexibility and its fault-tolerance. Other components can be added and/or deleted without requiring manual adaptation of the system, as long as the system is capable of knowing that these components are available and what are their main characteristics are. On the other hand, this scheme does not optimize how the resources are used, which would require a different approach.

15.5.2 Market-Based Coordination of Distributed Generation

Coordinating distributed energy sources in modernized power markets is another field where MAS-based cooperation can be employed. PowerMatcher, a concept developed by ECN in the Netherlands [52,69,70], is an example. The objective is to coordinate agents so as to balance supply and demand in an economically efficient way. To solve this problem, agents have the capability to competitively trade energy on a common market. These market-based negotiations provide a decision-making framework based on microeconomics.

The negotiation process relies on dynamic pricing schemes, in which prices can vary throughout the day depending on the balance between supply and demand: the higher the demand, the higher the prices, and vice-versa. Each agent buys or sells energy depending on its type (load, source, etc.), and commits to this bought or sold amount. In this competitive general equilibrium market, all agents have access to the same information on price, obtained by searching for the equilibrium between supply and demand.

The PowerMatcher architecture is based on a logical tree structure, as shown in Fig. 15.26. Each group of agents is coordinated by a concentrator agent, which is in turn coordinated by another concentrator at a higher level. Four types of agents are used:

- *Device agents* control the distributed sources, loads and storage units of the system, and are able to converse with concentrators. They have their own goals and properties, and issue bids based on what they are willing to pay (resp. be paid) for a given amount of energy to consume (resp. produce).
- *Concentrator agents* (Fig. 15.25) locally concentrate and aggregate the bids they receive, and back propagate the price chosen by the auctioneer to lower level agents (device agents and other concentrators).

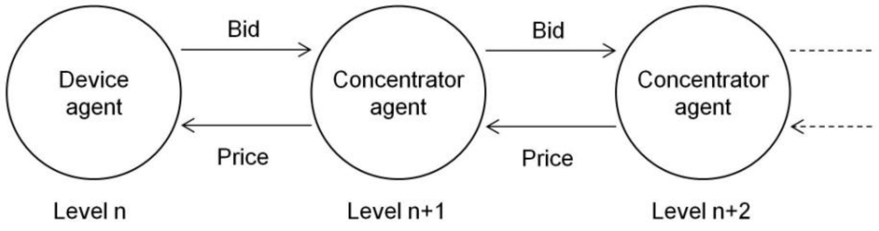

Fig. 15.25 Each device and concentrator agent emits a bid for the power it wants to sell or buy, and receives a price computed by the auctioneer in return. This price is back propagated through all layers of the system by the concentrator agents.

- The *auctioneer agent*, at the top of the tree, centralizes the bids of the agents connected to it and finds the equilibrium price based on the bids it has received to clear the market. The obtained price is then sent back to the agents if it has significantly changed from its earlier value.
- The *objective agent* is connected to the auctioneer agent and defines the objective of the system.

Each agent, whether it is a device or a concentrator, only knows its direct neighbors.

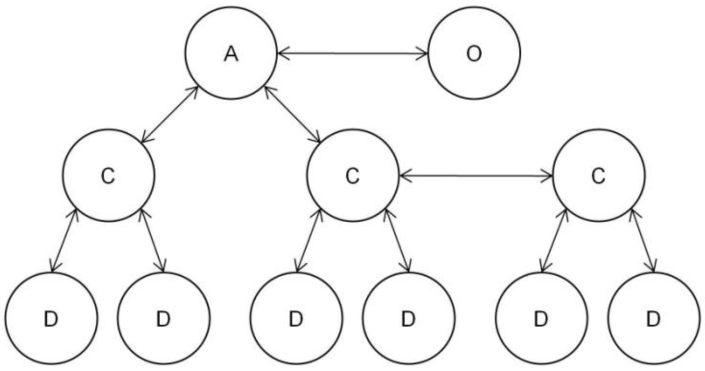

Fig. 15.26 The MAS proposed in PowerMatcher uses a hierarchical architecture, based on four types of agents: an auctioneer (*A*), an objective agent (*O*), concentrator agents (*C*) and device agents (*D*). The above diagram shows an example with a limited number of agents.

A negotiation round is initiated with a given frequency (of a few minutes) by the auctioneer agent. The agent starts by asking other agents to submit their bids, and computes the equilibrium price based on information it has gathered. Other events can trigger a round, such as a sudden change in demand. DF and AMS agents are used so that each agent can find other agents to negotiate with.

The main interests of this concept are its ability to operate in environments with a large share of renewable energy sources, where the energy sector has been fully liberalized, and consumers can also be producers. The selected bottom-up approach, combined with the decentralized decision-making process, enable a very flexible architecture, where a change in the structure, by adding or removing functionalities or agents, does not affect the overall operation of the system. This structure is plug-and-play and scalable: contrary to other architectures, this solution is not based on a central optimization algorithm that would hinder the ability to scale the system. By aggregating local bids, the need for intensive data communication is avoided. Similar concepts can also be applied to trade other commodities, such as heat in a microgrid where cogeneration is available. The system is being tested in a demonstration project called PowerMatching city[2], located in the Netherlands.

Comparable and interesting MAS- and market-based approaches were proposed by other researchers, such as DEZENT [71], MASCEM [72], work by Dimeas and Hatziargyriou [73,74], Pipattanasomporn et al. [75], and Funabashi et al. [76].

15.5.3 Restoration

One of the earliest applications of MAS in power systems is for restoration, as described in Nagata and Sasaki [77]. Restoration is needed after a partial or global blackout has occurred. Operators usually employ an outage management system (OMS) to automatically restore power, by sequentially reenergizing all the buses in the grid so as to serve the loads. At the end of the process, the grid is back to its normal operation mode, as before the event. The objective is to serve the maximum of loads connected to the buses. Several constraints have to

[2] http://www.powermatchingcity.nl. Last visited June 27, 2011.

be respected: the balance between supply and demand, the capacity of each source, and the voltage limits on each bus and branch.

In the proposed approach, two types of agents are used (Fig. 15.27): bus agents (BAG, one for each bus), and a facilitator agent (FAG). Each BAG tries to restore power to the load connected to its bus and can only communicate with its geographical neighbors and with the FAG. Its behavior follows simples rules:

- If the bus can be powered by several branches, then the agent chooses the branch with the highest power in order to restore power as quickly as possible.
- If the available power is not sufficient, the agent negotiates with its neighbors to find another solution and get a higher power to energize its load.
- If no satisfying solution can be found, the agent sheds its loads to its minimum power.

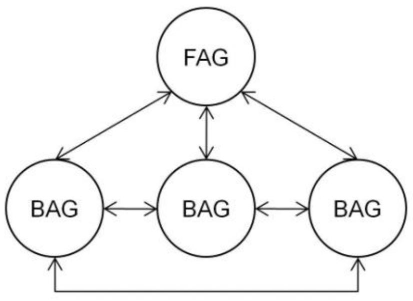

Fig. 15.27 The proposed MAS is made of two types of agents, each with their own roles: bus agents (BAG) and a facilitator agent (FAG) [77].

The facilitator agent FAG facilitates the negotiation by classifying buses according to their voltage level, and choosing one BAG at each voltage level to initiate the restoration process. Several restoration processes can be run in parallel at different voltage levels, which accelerates the overall restoration.

In a later paper [78], the authors proposed another version of this concept, where FAGs are feeder agents and act as managers for the decision-making process. The architecture shown in Fig. 15.27 is replicated at each feeder, and feeder agents can communicate with each other. This enables the system to operate on a much larger scale, with restoration “groups” cooperating with each other. This approach is an improvement compared to traditional OMS, where everything is coordinated by a central controller. The proposed system is much less complex, as the same algorithms can be replicated and cooperate to achieve their objectives, but could be expected to provide results similar to the ones obtained by centralized algorithms.

Other MAS-based restoration concepts are presented in the literature, such as in [79,80]. Pan and Tsai [81] also propose a BDI-based solution to this problem.

15.5.4 Protection Fault Analysis and Management

Davidson et al. [82] propose a fault diagnosis and management called PEDA, for Protection Engineering Diagnosis Agents. PEDA has been used since 2004 by a British utility. Its objective is to automate the analysis and management of faults recorded by SCADAs and digital fault recorders (DFR). As thousands of faults can be recorded during an event such as a storm, engineers need to be supported by a system capable of extracting the most important information from the mass of collected data.

In this case, MAS technology was used as a medium for system integration. The hardware and software used to achieve fault analysis and management can indeed change rather frequently over time, and the system has to be able to accommodate such evolutions. The proposed MAS integrates several tools, including a rule-based expert system to classify and interpret data from SCADAs, and a model-based reasoning system to validate protection operations using DFR data.

The architecture of PEDA is based on FIPA's agent management reference model presented earlier, and, therefore, implements both DF and AMS services. As shown in Fig. 15.28, additional agents are used for retrieving, interpreting, analyzing, etc. data, each with its own role in the system. Some of these agents encapsulate systems that were already being used in the legacy system.

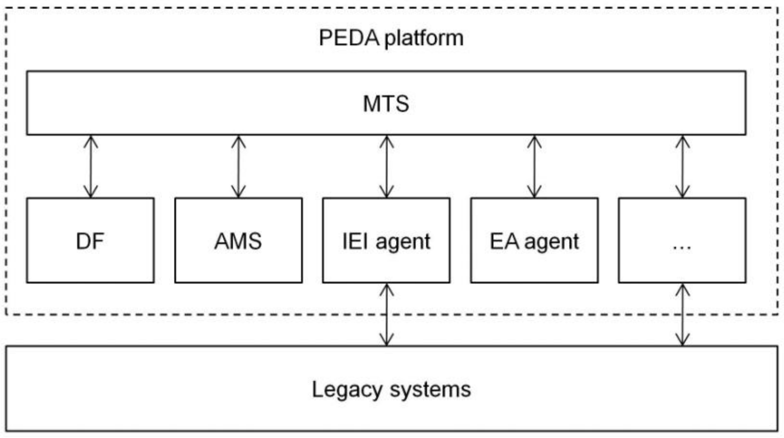

Fig. 15.28 In this MAS, both DF and AMS agents are used, in addition to agents encapsulating legacy systems, and to other agents such as the EA agent that interacts with operators. Only a part of all agents is represented in this diagram [82].

ACL and FIPA subscription protocols are used for inter-agent interaction and communication, together with an ontology for disturbance diagnosis. For example, all agents subscribe to IEI agents (telemetry processors), which in turn inform them when they have identified an incident. The AMS and DF enable the MAS to acknowledge when an agent joins the system: each agent autonomously searches for other agents that can provide them the information they need.

This MAS is one of the first applications tested in the power industry. The problems the authors had to face when transferring their laboratory prototype to the real system provide useful lessons (e.g., on platform choice (here JADE), standards, user interfaces, etc.) for other projects that may try to achieve similar tests. It also builds on an approach different from the ones used in traditional MAS work, as it does not primarily focus on distributed intelligence, but rather on the flexibility of the approach, transforming a legacy fault analysis and management system into a flexible and extensible tool, in which software and hardware parts can be easily changed.

15.6 Future Developments

Although they are very promising, MAS are still an emerging technology in the field of power systems. Topics to explore in the future include:

- *Making agents smarter*. Most agents used in MAS for power systems tend to be closer to reactive agents than to cognitive agents. Making them smarter would enable a greater autonomy and more complex decision-making processes with additional parameters taken into account. The learning capability of agents could for example have many applications, from forecasting to scheduling, and could partially pre-solve some problems. Similarly, the planning capacity of agent is rarely used.

- *Fully distributing and automating the decision-making process*. Decision-making processes are usually partially centralized, i.e., not fully distributed. If one or more central controller fails, or if a communication channel is cut [83], then the whole system operation may be compromised. Redundancy can partially help avoid such situations, but increases costs. Achieving a fully distributed decision-making process would solve this problem: if each agent takes his own decisions, sometimes after discussing with others, it would not be greatly impacted by the failure of another agent. However, distributing decision-making often comes at the cost of having more intensive communication.

- *Creating a modeling and simulation tool to facilitate the development and testing of MAS for power systems.* Current solutions require co-simulation between a MAS and a simulation tool such as Matlab/Simulink or PowerWorld Simulator [84]. A tool mixing both would highly simplify simulation, development and testing MAS for these applications.

- *Testing MAS on very large scale grids*. In most research papers, MAS are tested on small scale systems and show good results. However, very few work has focused on the scalability of MAS for such applications: how would the system perform when the grid size increases? will communication and computation requirements explode? how will the system behave in case of perturbations or attacks? Such studies will be essential to determine the optimal decentralization degree to use, as well as the corresponding infrastructure costs for developing smart grid communication infrastructures.

- *Testing MAS for control on real scale industrial systems*. So far, MAS have been mostly a subject of research in the field of power systems, but their industrial and commercial applications are rare, with exceptions such as the application presented in section 15.5.4. Feedback and lessons learned from early experiments will be essential to foster the adoption of MAS technology in the industry. Three main difficulties have to be faced: utilities are sometime reluctant to investigate the use of MAS, which are still widely unknown due to a lack of practical experience; communication costs are high because MAS often require new equipment, especially at the distribution level; and power system standards are not written with MAS in mind. These difficulties also apply to most new smart grid technologies.

- *Integrating user behavior models*. As consumers are and will remain central players in smart grids, arises the need to model consumer behavior. In the future, so-called “prosumers” may take decisions using their home energy management interface and will directly impact how and when electricity is consumed, for example through demand response mechanisms that may affect consumers' comfort level and billing, but also utilities. Qualitative studies provide important and useful information, but quantitative results are required to make regulatory and investment decisions, and to develop potentially successful new products. In parallel to modeling such consumer agents, real experiments will also have to take place to make these models more accurate and validate them.

- *Securing MAS communication*. A prerequisite to any commercial application is that systems must be secure and resist to attacks of all sorts. Although this problem is much wider than only for MAS, its importance was stressed by the recent concerns raised by worms such as Stuxnet, which aimed at disrupting industrial processes controlled by PLCs (programmable logic controllers). MAS thus have to include the most efficient and secure technologies before being deployed in the industry.

15.7 References

1. Wooldridge M and Weiss G (1999) Multi-Agent Systems. The MIT Press.
2. Ferber J (1999) Multi-Agent Systems: An Introduction to Artificial Intelligence. Addison-Wesley.
3. Rao AS and Georgeff MP (1991) Modeling Rational Agents within a BDI-Architecture. In Allen J, Fikes R and Sandewall E (eds.). Proceedings of the 2nd International Conference on Principles of Knowledge Representation and Reasoning. Morgan Kaufmann publishers Inc., San Mateo 473-484.
4. McArthur S, Davidson E, Catterson V, Dimeas A, Hatziargyriou N, Ponci F and Funabashi T. (2007) Multi-Agent Systems for Power Engineering Applications - Part I: Concepts, Approaches, and Technical Challenges. Power Systems, IEEE Transactions on 22:1743-1752
5. McArthur S, Davidson E, Catterson V, Dimeas A, Hatziargyriou N, Ponci F and Funabashi T. (2007) Multi-Agent Systems for Power Engineering Applications - Part II: Technologies, Standards, and Tools for Building Multi-agent Systems. Power Systems, IEEE Transactions on 22:1753-1759

7. Zambonelli F, Jennings N and Wooldridge M (2003) Developing multiagent systems: The Gaia methodology. ACM Transactions on Software Engineering and Methodology (TOSEM). 12:370
8. Omicini A (2001) SODA: Societies and Infrastructures in the Analysis and Design of Agent-Based Systems. In: Ciancarini P and Wooldridge M (eds.) Agent-Oriented Software Engineering. Lecture Notes in Computer Science. Springer Berlin / Heidelberg. 1957:311- 326
9. Wood M and DeLoach S (2001) An overview of the multiagent systems engineering methodology. Agent-Oriented Software Engineering. 1-53

10. Caire G, Coulier W, Garijo F, Gomez-Sanz J, Pavon J, Kearney P and Massonet P (2004) The MESSAGE methodology. Methodologies and Software Engineering for Agent Systems. Springer. 177-194
11. Bresciani, P. and Perini, A. and Giorgini, P. and Giunchiglia, F. and Mylopoulos, J (2004) Tropos: An agent-oriented software development methodology. Autonomous Agents and Multi-Agent Systems. Springer. 8(3)203-236
12. Bernon C, Gleizes MP Peyruqueou S and Picard G (2003) ADELFE: A methodology for adaptive multi-agent systems engineering. Engineering Societies in the Agents World III. Springer. 70-8
13. Iglesias C, Garijo M, González J and Velasco J (1996) A methodological proposal for multiagent systems development extending CommonKADS. Proceedings of the 10th Banff Knowledge Acquisition for Knowledge-Based Systems Workshop
14. Brazier F, Dunin-Keplicz B, Jennings N and Treur J (1997) Desire: Modeling multi-agent systems in a compositional formal framework. International Journal of Cooperative Information Systems. 6:67
15. Henderson-Sellers B and Giorgini P (2005) Agent-oriented methodologies. IGI Global.
16. Dam KH and Winikoff M (2004) Comparing agent-oriented methodologies. Agent-Oriented Information Systems. Springer. 78-93
17. Gomez C, Isern D and Moreno A (2007) Software Engineering Methodologies to Develop Multi-Agent Systems : State-of-the-art, Research report DEIM-RR-07-003. Universitat Rovira i Virgili
18. Crespi V, Galstyan A and Lerman K (2008) Top-down vs bottom-up methodologies in multi-agent system design. Autonomous Robots. Springer. 24:303-313
19. Foundation for Intelligent Physical Agents (FIPA) (2011). Available from: http://www.fipa.org Accessed 1 July 2011
20. International Electrotechnical Commission (IEC) (2011). Available from: http://www.iec.ch Accessed 1 July 2011
21. International Electrotechnical Commission (2011) IEC TC 57 Power Systems management and associated information exchange. Available from: http://tc57.iec.ch Accessed 1 July 2011
22. Liang Y and Campbell RH. Understanding and simulating the IEC 61850 standard. Available from: https://www.ideals.illinois.edu/handle/2142/11457 Accessed 1 July 2011
23. DNP users group (2005) A DNP3 Protocol Primer. Available from: http://www.dnp.org/About/DNP3%20Primer%20Rev%20A.pdf Accessed 1 July 2011
24. CIM users group (2009) Available from: http://cimug.ucaiug.org Accessed 1 July 2011
25. Britton JP and Devos AN (2005) CIM-based standards and CIM evolution. Power Systems, IEEE Transactions on 20(2):758-764
26. McNaughton G and McNaughton W (2006) MultiSpeak Version 3.0 User's Guide. Available from: http://www.multispeak.org/utilities/UserGuides/Documents/MultiSpeak_V3_UserGuide Final_013006.pdf Accessed 1 July 2011
27. Saleem A, Honeth N and Nordström L (2010) A case study of multi-agent interoperability in IEC 61850 environments. Innovative Smart Grid Technologies Conference Europe (ISGT Europe), 2010 IEEE PES, 1-8
28. Apostolov A (2006) Multi-agent systems and IEC 61850. IEEE Power Engineering Society General Meeting
29. FIPA (2002) FIPA ACL Message Structure Specification. SC00061G

30. Catterson VM, Baker PC, Davidson EM and McArthur SDJ (2010) An upper ontology for power engineering applications. Available from http://ewh.ieee.org/mu/pes-mas/ Accessed 1 July 2011
31. FIPA (2004) FIPA Agent Management Specification. SC00023K
32. Russell S and Norvig P (2010) Artificial Intelligence, A Modern Approach. 3rd edition, Prentice Hall
33. Brooks R (1986) A robust layered control system for a mobile robot. Robotics and Automation, IEEE Journal of [legacy, pre-1988] 2(1):14–23. doi:10.1109/JRA.1986.1087032
34. Laird J, Newell A, Rosenbloom P (1987) SOAR: An architecture for general intelligence. Artificial Intelligence 33(1):1-64
35. Albus JS (1993) A Reference Model Architecture for Intelligent Systems Design. In: Antsaklis PJ and Passino KM (eds.) An Introduction to Intelligent and Autonomous Control. Kluwer Academic Publishers
36. Rapid-I. Available from http://rapid-i.com Accessed 4 April 2012
37. KNIME. Available from http://www.knime.org Accessed 4 April 2012
38. WEKA. Available from http://www.cs.waikato.ac.nz/~ml/weka Accessed 4 April 2012
39. Sonnenburg S, Rätsch G, Henschel S, Widmer C, Behr J, Zien A, De Bona F, Binder A, Gehl C and Franc V (2010) The SHOGUN Machine Learning Toolbox, Journal of Machine Learning Research, 11:1799−1802
40. Mellit A (2008) Artificial Intelligence technique for modeling and forecasting of solar radiation data: a review. International Journal of Artificial intelligence and soft computing. Inderscience. 1(1):52-76
41. Bhaskar M, Jain A, Srinath V (2010) Wind speed forecasting: present status. International Conference on Power system technology (POWERCON) pp 1-642. Metaxiotis K, Kagiannas A, Askounis D and Psarras J (2003) Artificial intelligence in short term electric load forecasting: a state-of-the-art survey for the researcher. Energy Conversion and Management. Elsevier. 44(9):1525-1534
43. Kodogiannis VS and Anagnostakis EM (2002) Soft computing based techniques for short-term load forecasting. Fuzzy Sets and Systems. Elsevier. 128(3):413-426
44. Ernst D, Glavic M and Wehenkel L (2004) Power Systems Stability Control: Reinforcement Learning Framework. IEEE Transactions on Power Systems 19(1)
45. Horling B and Lesser V (2005) A Survey of Multi-Agent Organizational Paradigms. The Knowledge Engineering Review 19(4):281-316
46. Yen J, Yan YH, Wang BJ Sin PKH and Wu FF (1998) Multi-agent coalition formation in power transmission planning. System Sciences, Proceedings of the Thirty-First Hawaii IEEE International Conference on. 4: 433-443
47. FIPA (2001) FIPA Request Interaction Protocol Specification. XC00026F
48. FIPA (2002) FIPA Contract Net Interaction Protocol Specification. SC00029H
49. FIPA (2001) FIPA English Auction Interaction Protocol Specification. XC00031F
50. FIPA (2001) FIPA Dutch Auction Interaction Protocol Specification. XC00032F
51. Ausubel LM, Milgrom P (2006) The lovely but lonely Vickrey auction. In Cramton P, Shoham Y, Steinberg R (eds), Combinatorial Auctions (2006) pp 17–40
52. Kok JK, Scheepers MJJ and Kamphuis IG (2010) Intelligence in electricity networks for embedding renewables and distributed generation. Intelligent Infrastructures. Springer. 179-209
53. Motto AL, Galiana FD, Conejo AJ and Huneault M (2002) On Walrasian equilibrium for pool-based electricity markets. IEEE Transactions on Power Systems, 17(3):774-781

54. Pitt J, Kamara L, Sergot M and Artikis A (2006) Voting in multi-agent systems. The Computer Journal 49(2):156
55. FIPA (2001) FIPA Brokering Interaction Protocol Specification. SC00033H
56. Pease M, Shostak R and Lamport L (1980) Reaching agreement in the presence of faults. Journal of the ACM. 27(2):228-234
57. Olfati-Saber R, Fax JA and Murray RM (2007) Consensus and Cooperation in Networked Multi-Agent Systems. Proceedings of the IEEE 95(1):215-233
58. Kamboj S, Pearre N, Kempton W, Decker K, Trnka K and Kern C (2010) Exploring the formation of Electric Vehicle Coalitions for Vehicle-To-Grid Power Regulation. AAMAS workshop on Agent Technologies for Energy Systems (ATES 2010)
59. Contreras J, Klusch M and Yen J (1998) Multi-agent coalition formation in power transmission planning: a bilateral Shapley value approach. Proc. 4th Int'l Conf. Artificial Intelligence Planning Systems. 19-26
60. De Weerdt M, Ter Mors A, Witteveen C (2005) Multi-agent planning: An introduction to planning and coordination. Available from: http://www.st.ewi.tudelft.nl/~mathijs/publications/easss05.pdf Accessed 1 July 2011
61. Ziparo VA (2005) Multi-Agent Planning. Available from: http://www.dis.uniroma1.it/~dottoratoii/db/relazioni/relaz_ziparo_1.pdf Accessed 1 July 2011
62. Marinova ZL (2002) Planning in multiagent systems. Msc thesis submitted to the Faculty of Mathematics and informatics of Sofia University "St. Kliment Ohridski". Available from: http://www.ontotext.com/sites/default/files/publications/theis-zm.pdf Accessed 1 July 201163. Comparison of agent-based modeling software (2011) In: Wikipedia, The Free Encyclopedia. Available from: http://en.wikipedia.org/w/index.php?title=Comparison_of_agent-based_modeling_software&oldid=434070966 Accessed 1 July 2011
64. FIPA (2003) Publicly Available Agent Platform Implementations. Available from: http://www.fipa.org/resources/livesystems.html Accessed 1 July 2011
65. Bellifemine F, Caire G and Greenwood D (2007) Developing multi-agent systems with JADE. Wiley
66. BDI4JADE: A BDI layer on top of JADE. Available from: http://www.inf.puc-rio.br/~ionunes/bdi4jade/ Accessed 1 July 2011
67. Camacho D, Aler R, Castro C and Molina JM (2002) Performance evaluation of ZEUS, JADE and SkeletonAgent frameworks. Systems, man and cybernetics, 2002 IEEE international conference on. 4:6
68. Lagorse J, Paire D and Miraoui A (2010) A multi-agent system for energy management of distributed power sources. Renewable Energy 35:174-182
69. Hommelberg MPF, Van der Velde BJ, Warmer CJ, Kamphuis IG and Kok JK (2008) A novel architecture for real-time operation of multi-agent based coordination of demand and supply. IEEE Power and Energy Society General Meeting-Conversion and Delivery of Electrical Energy in the 21st Century
70. Kamphuis R, Kok K, Warmer C and HommelbergM (2008) Architectures for novel energy infrastructures: Multi-agent based coordination patterns. Infrastructure Systems and Services: Building Networks for a Brighter Future (INFRA), 2008 First International Conference on
71. Wedde HF, Lehnhoff S, Handschin E and Krause O (2006) Real-time multi-agent support for decentralized management of electric power. Real-Time Systems, 2006. 18th Euromicro Conference on

72. Praca I, Ramos C, Vale Z, Cordeiro M (2003) MASCEM: A multiagent system that simulates competitive electricity markets. IEEE Intell Syst 18(6):54–60
73. Dimeas AL and Hatziargyriou ND (2005) Operation of a multiagent system for microgrid control. Power Systems, IEEE Transactions on 20(3):1447-1455
74. Dimeas AL, Hatziargyriou ND (2007) Agent based control of virtual power plants. International conference on intelligent systems applications to power systems (ISAP 2007) pp 536-541
75. Pipattanasomporn M, Feroze H and Rahman S (2009) Multi-agent systems in a distributed smart grid: Design and implementation. IEEE/PES Power Systems Conference and Exposition, 2009 (PSCE'09)
76. Funabashi T, Tanabe T, Nagata T, Yokoyama R (2008) An autonomous agent for reliable operation of power market and systems including microgrids. Third international conference on electric utility deregulation and restructuring and power technologies (DRPT 2008) pp 173–177
77. Nagata T and Sasaki H (2002) A multi-agent approach to power system restoration. Power Systems, IEEE Transactions on 17:457-462
78. Nagata T, Tao Y, Kimura K, Sasaki H and Fujita H (2004) A multi-agent approach to distribution system restoration. The 2004 47th Midwest Symposium on Circuits and Systems (MWSCAS '04) 2:333-336
79. Solanki JM, Khushalani S and Schulz NN (2007) A multi-agent solution to distribution systems restoration. Power Systems, IEEE Transactions on 22(3):1026-1034
80. Xu Y and Liu W (2011) Novel Multiagent Based Load Restoration Algorithm for Microgrids. Smart Grid, IEEE Transactions on 2(1):152-161
81. Pan YT and Tsai MS (2009) Development a BDI-based intelligent agent architecture for distribution systems restoration planning. Intelligent System Applications to Power Systems (ISAP'09) 15th International Conference on
82. Davidson EM, McArthur SDJ, McDonald JR, Cumming T and Watt I (2006) Applying multi-agent system technology in practice: automated management and analysis of SCADA and digital fault recorder data. Power Systems, IEEE Transactions on 21(2):559-567
83. Buldyrev S, Parshani R, Paul G, Stanley H and Havlin S (2009) Catastrophic cascade of failures in interdependent networks. Nature 464:1025-1028
84. Roche R, Natarajan S, Bhattacharyya A, Suryanarayanan S (2012) A Framework for Co-simulation of AI Tools with a power systems analysis software. 23rd International Workshop onDatabase and Expert Systems Applications (DEXA) pp 350-354