Service based integration of Grid and Multi-Agent Systems models

Clement Jonquet, Pascal Dugenie and Stefano A. Cerri

February 2006

LIRMM, CNRS & University Montpellier II
161 Rue Ada, 34392 Montpellier Cedex 5, France
{cerri,dugenie,jonquet}@lirmm.fr

Abstract

The GRID and MAS (Multi-Agent Systems) communities believe in the potential of GRID and MAS to enhance each other because they have developed significant complementarities. Thus, both communities agree on the "what" to do: promote an integration of GRID and MAS models. However, even if the "why" to do it has been stated and assessed, the "how" to do it is still a research problem. The paper addresses this problem by means of a service oriented approach. We present the concept of service at the intersection of GRID and MAS i.e., how an integration of these domains may enhance the realisation of dynamically generated services based on conversations: Services are exchanged (i.e., provided and used) by *agents* through *GRID* mechanisms and infrastructure. We firstly introduce a novel formalisation to describe GRID and MAS key concepts. These concepts are directly influenced by OGSA (Open Grid Service Architecture) and the STROBE agent communication and representation model developed for dynamic service generation. This formalisation intends to represent the respective domain ontologies and the relations between those concepts. Secondly, we use this formalisation as a tool to map GRID and MAS concepts in order to suggest directions for the conception, design and implementation of an integrated model supporting both agents and Grid services.

Keywords: Multi-Agent Systems, agent, STROBE model, GRID, Grid service, OGSA, Distributed systems, Service oriented architecture, Web Service, Dynamic service generation.

1 Introduction

Historical background. GRID and MAS (Multi-Agent System) are two kinds of distributed systems. Yet, the motivations are different. In 2004, [FJK04] summarised these motivations. GRID focuses on a reliable and secure resource sharing infrastructure, whereas MAS focus on flexible and autonomous behaviour in uncertain and dynamic open environments. For the last twenty years, distributed systems were based on system oriented architectures (client-server, peer-to-peer). These kinds of architectures are sufficient to provide remote connections and basic operations like file transfer. However, system oriented architectures are not suitable for a real interoperation and integration of distributed components such as documents, resources, applications, knowledge... and services. Thus, Service Oriented Architectures (SOA) have emerged for which Web services are the main current framework. SOAs define a model for the execution of distributed software applications[1]. The first ideas of the software component based approaches were i) to standardize invocation mechanisms; ii) to make transparent the location of these components on the network. Then, from the success of XML languages, the concept of service detached from a common middleware emerged; a notion based only on standardized and interoperable protocols and technologies over the Internet. More recently, GRID took a major place in SOA by augmenting the basic notion of Web Service with two significant features: service state and service lifetime management. This is the concept of Grid service. MAS have also followed naturally the path towards SOA as the interests turn into dynamic service providing, business process management, semantic services, etc. The Service Oriented Computing

[1]This is historically due to software component based approaches, for example, the work of the Open Group with Distributed Computing Environment (DCE) and the Object Management Group with CORBA or Microsoft with (Distributed) Component Object Model (COM/DCOM) or Sun with Java Remote Method Invocation (RMI).

(SOC) community turns to MAS considering the important capacities of agents to provide and use one another services.

Service as a unifying concept. Although GRID and MAS are two separated domains, they meet with the concept of service. [FJK04] forsees services as the core ”unifying concept” that underlies GRID and MAS (also suggested by [RM00]). GRID is said to be the first distributed architecture (and infrastructure) really developed in a service oriented perspective: Grid services are compliant Web services, based on the dynamic allocation of virtualized resources to an instantiated service [FKNT02, CTT05]. On the other side, agents are said to be autonomous, intelligent and interactive entities which may offers services (in the sense of particular problem solving capabilities) [Fer99, Jen01]. The concept of service is clearly at the intersection of the GRID and MAS domains as figure 1 shows.

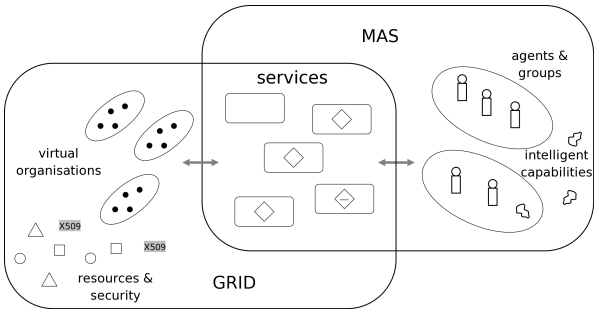

Figure 1: Intersection of GRID and MAS

Dynamic Service Generation. Quite recently, a research interest appeared to distinguish real services from simple product deliveries in SOA: Business Process Management, semantics, engagement etc. (see [SH05] for a recent overview of challenges in SOAs). Web services are often criticised because they are no more than Remote Procedure Calls (RPC) which do not have any user adaptation, no memory, no lifetime management, no conversation handling (simple request/answer interaction). They are passive, they lack semantics and they do not consider autonomy of components. The SOC community realises progressively that the notion of service has to surpass HTTP protocols, SOA current standards, RPCs and Extensible Markup Language (XML) to be enriched by results from other research domains such as information systems, concurrent systems, knowledge engineering, interaction and especially GRID and MAS.

To provide a service means to identify and offer a solution (among many possible ones) to the problem of another. The next generation of services will consist of dynamically generated services i.e., services constructed on the fly by the service provider according to the conversation it has with the service user. In *Dynamic Service Generation* (DSG), the user (human or artificial) is not assumed to know exactly what the provider (also human or artificial) can offer him/her. He/she finds out and constructs step by step what he/she wants as the result of the service provider’s reactions. The main idea of DSG is that a service may be based on a conversation. Actually, DSG highlights the idea of process and the creation of something new instead of merely delivering something already existing. In current life, when somebody looks for clothes, *buying ready-to-wear clothes* is analogue to asking for a product, whereas *having clothes made by a tailor* is analogue to requiring a service to be generated. Singh and Huhns [SH05] talk about *service engagement*, instead of simple method invocation. In [JC06] the authors extensively described the DSG notion by enumerating a set of characteristics that are related to DSG. The two main inspiring sources of requirements for DSG are MAS and GRID. In [JC05] the authors also present the STROBE model as an agent representation and communication model thought and constructed in order to developed dynamically generated services. The STROBE model, further developed in section 3.3, is one of the key means of the proposed GRID-MAS integrated model. The shift from a current limited perspective in service exchange scenarios to DSG is the problem addressed by this paper.

Toward a MAS-GRID integration. In order to summarize our reflection at the crossing of the three domains i.e., GRID, MAS and SOC, we identify two key ideas that underlie the paper:

- GRID and MAS have each developed a service oriented behaviour, therefore the concept of service may represent and glue a common integration;

- New needs in service exchange scenarios are clearly highlighted (dynamicity, conversation based, user centred behaviour, business process, semantics etc.) and may be fulfilled by integrating GRID and MAS complementarities.

Therefore, this paper proposes firstly, a novel formalisation to describe GRID and MAS key concepts. The GRID concepts are directly influenced by OGSA (Open Grid Service Architecture) such as for example: resource, host, virtualization unit, service container, Grid service, X509 certificate, virtual organization etc. The MAS concepts are directly influenced by different approaches in MAS, such as the STROBE model, the Agent Group Role (AGR) model, Believe Desire Intention (BDI) architectures, Foundation for Intelligent Physical Agents (FIPA) agents etc. They are for example: agent, group, capability, cognitive environment etc. This formalisation represents the domains concepts as well as their relation, for example: resource virtualization, Grid service instantiation, Grid user memberships, agent interaction, etc. With this formalisation, we aim to identify key opportunities for harmonisation and cross fertilization between GRID and MAS, as they have developed significant complementarities.

Secondly, the paper shows how to use this formalisation to map GRID and MAS concepts in order to propose an integrated model. One of the key ideas of this integration is a representation of agent capabilities as Grid services in a service container. Another one is the assimilation of the service instantiation mechanism – fundamental in GRID as it allows Grid services to be stateful and dynamic – with the dedicated cognitive environment instantiation mechanism – fundamental in STROBE as it allows an agent to dedicate to another one a conversation context. We do not want services to be realised by simple object programs, but by intelligent, autonomous and interactive agents (human or artificial) able to have conversations in order to dynamically generate services.

Paper overview. The rest of the paper is organised as follow: section 2 presents a state of the art concerning GRID and MAS. Through this state of the art, we have identified three GRID-MAS analogies that have strongly influenced our integrated model. Section 3 presents the elements of the formalisation: a service taxonomy, as well as GRID and MAS key concepts. Section 4 reports about related work in GRID-MAS convergence. Then a mapping of key concepts is done in order to integrate them in a common model. Finally, section 5 gives some perspectives and concludes the paper.

2 State of the art

2.1 GRID and MAS need each other: Brain meets brawn

Fortunately, GRID and MAS have several similarities and relate to the same domain: the development and deployment of a distributed SOA. However, GRID and MAS have developed different aspects of SOA. [FJK04] emphasise the overlap in problems that GRID and MAS address without sharing research progress in one or the other area: ”An integrated GRID/agent approach will only be achieved via a more fine-grain intertwining of the two technologies”: GRID and MAS need each other. The authors describe GRID as the ”brawn” i.e., infrastructure, tools, and applications for secure resource sharing within dynamic and geographically distributed virtual organizations. However:

- GRID is rigid, inflexible and interaction poor; GRID provides uniform mechanisms for accessing raw data, but is not able to deal with these data and their semantics to consider them as knowledge; Grid services are able to manage state, but do not have the intelligent ability to decide how and why to change state; Orchestration/choreography to compose Grid services needs a high level communication language GRID does not provide; VO are groups of Grid user entities (human and machine) that form according to common needs, objectives and services shared; but GRID does not provide the adequate mechanisms to create, manage, integrate or leave a VO.

MAS is described as the ”brain” i.e., concepts, methodologies and algorithms for autonomous problem solvers that can act flexibly in uncertain and dynamic environments in order to achieve their aims and objectives. However:

- MAS are often not robust, not large scaled, ant not secure; MAS need robustness, interoperation and standardisation; MAS provide sophisticated internal reasoning capabilities, but offer no support for secure interaction or service discovery; MAS cooperation or collaboration algorithms produce socially optimal outcomes, but assume the agents have complete knowledge of all outcomes that any potential grouping can

produce; MAS negotiation algorithms achieve optimal outcomes for the participating agents, but assume that all parties in the system are known at the outset of the negotiation and will not change during the system operation.

2.2 GRID state of the art

2.2.1 Sharing resources in virtual organization

The essence of the GRID is nicely reflected by its original metaphor: the delegation to the electricity network to offer us the service of providing us enough electric power as we need it when we need it even if we do not know where and how that power is generated. At the end of the month, we pay a bill that corresponds to our consumption. The GRID aims to enable "flexible, secure, coordinated resource sharing and coordinated problem solving in dynamic, multi-institutional virtual organization" [FKT01]. Actually, it was originally designed to be an environment with a large number of networked computer systems where computing (Grid computing) and storage (data Grid) resources could be shared as needed and on demand. GRID provides the protocols, services and software development kits needed to enable flexible, controlled resource sharing on a large scale. This sharing is, necessarily, highly controlled, with resource providers and users defining clearly and carefully just what is shared, who is allowed to share, and the conditions under which sharing occurs. A GRID system is naturally highly dynamic and should be able to adapt at runtime to changes in system state as resource availability may fluctuate. Such fluctuation may result from connection/disconnection of computing resources, human interaction/interruption on the computers, etc. GRID was firstly limited to computing and storage services but after extended to any kind of service insofar this is based on the dynamic allocation of virtualized resources to an instantiated service.

Grid users are members of *virtual organizations/communities*. A virtual organization is a dynamic collection of individuals, institutions and resources bundled together in order to share resources and services as they share common goals.

2.2.2 Heritage of Web services

Nowadays, the main way of implementing a SOA (framework) is by means of Web services (`www.w3.org/2002/ws`). Web services allow to access distributed functionalities on a network in a standardised way to enable interoperability. They are describable, discoverable and message based software components that perform some function. Many Web service technologies exist but Web services are mainly based on the three XML based standard languages: i) WSDL (Web Services Description Language) to describe software components i.e., functions that may be invoked; ii) SOAP (Simple Object Access Protocol) to describe methods for accessing these components i.e., message exchanges; iii) UDDI (Universal Description, Discovery and Integration) to publish a service and to identify/discover a service provider in a service registry.

The two main objectives of Web services are standardisation and interoperation (intra-enterprise and inter-enterprise). With Web services, different applications can communicate (with HTTP and through firewalls) with each other without knowing anything special about their implementation (operating system or programming language) but only dealing with a standardised interface where exchanges are expressed by XML documents. Certainly, the main advantage of Web services arises when we can compose them to create new services. Thus, from service invocations, which are single-shot two-party interactions, Web services started to evolve to business processes that are typically long-lived multiparty interactions. This is called Business Process Management (BPM). See for example on the workflow/orchestration side, BPEL4WS (Business Process Execution Language for Web Services) and on the conversation/choreography side, WSCL (Web Services Conversation Language). BPM is also detailed in section 2.4.

The lack of semantics of Web services is at the origin of research on ontologies, Semantic Web and Semantic Web Services i.e., semantically described Web services. The two main current technologies are proposed by the OWL-S (OWL-based Web Service Ontology) and WSMO (Web Service Modeling Ontology) working groups.

2.2.3 Grid services and GRID standardisation

GRID technologies have evolved from ad hoc solutions, and de facto standards based on the Globus Toolkit, to Open Grid Services Architecture (OGSA) [FKNT02] which adopts Web service standards and extend services to all kind of resources (not only computing and storage). Foster et al. call service [FKNT02]: *A (potentially transient) stateful service instance supporting reliable and secure invocation (when required), lifetime management, notification, policy management, credential management, and virtualization*. OGSA introduces two major

concepts in SOC by distinguish between service factory and service instance: the management of state (explicit service instantiation) and lifetime management (the transient aspect of service). Whereas Web services have instances that are stateless and non-transient, Grid service instances can be either stateful or stateless, and can be either transient or non-transient.

More recently, the Web Service Resource Framework (WSRF) [FFG+04] defines uniform mechanisms for defining, inspecting, and managing stateful resources in Web/Grid services. WSRF models Grid service as an association between two entities: a stateless Web service, that do not have state, and stateful resources that do have state. Stateful resources are elements with state, including physical entities (e.g. databases, file system, servers) and logical constructs (e.g. business agreements, contracts) that are persistent and evolve because of service interactions. A stateful service has internal state that persists over multiple interactions. For a recent precise overview of Grid service concepts and standardisation see for example [CTT05].

2.2.4 GRID evolution

Recently, the same semantic level add-on objective that occurred in the Semantic Web community occurred also in GRID. For an introduction to the concept of Semantic Grid, see [RJS01, Gel04]; or [CT03] for GRID based distributed knowledge discovery. The GRID is also used as a "learning Grid" where GRID architecture and principles are used to change e-learning paradigms [ACR+05]. BPM and Grid service composition start to be a research question in the GRID community as [KWvL02] which addresses the challenge of modelling workflow of Grid services.

2.3 MAS state of the art

2.3.1 The concept of agent

An agent [Fer99, Jen01] is a clearly identifiable physical or virtual autonomous entity which: is situated in a particular environment; is capable of perceiving (with sensors) and acting (with effectors) in that environment; is designed to fulfil a specific role; communicates directly with other agents; possesses its own state (and controls it) and skills; offers services (in the sense of particular problem solving capabilities); has a behaviour that tends to satisfy its objectives. Agents are reactive (able to respond in a timely fashion to changes that occur in their environment) and proactive (able to opportunistically adopt goals and take the initiative), capable and efficient (able to solve problems and reach private objectives) and adaptive (able to learn and change as a consequence of experiences).

Agents and MAS were extensively studied in literature, for example [Fer99, HS98, Woo02]. Historically, the agent paradigm comes from the object one, but differs according to three major aspects: autonomy i.e., for example, the fact of being able to refuse to answer a message; intelligence i.e., the ability to change its own state alone, without message passing; interaction i.e., the faculty to communicate directly and asynchronously with their environment and other agents by means of direct or indirect message passing with a communication language independent from the content of the communication and from the internals of agents. Agents have a persistent state and are able to have conversations. They are also able to use and reconcile ontologies. The term "societies of agents", highlights the importance of the social aspect in MAS. A MAS is not a simple set of agents put together in a common environment, but a real organization with social rules and interactions allowing cooperation and collaboration to solve problems that centralized systems (as intelligent as they can) would not have solved alone [Fer99, Woo02]. All these properties are interesting for using and providing services.

2.3.2 Agents and Web services

Actually, Web services are most of the time an interface on object oriented programs and one of the crucial evolutions toward DSG concerns the substitution of an agent oriented kernel to the current object oriented kernel for Web services. Some work has already been proposed using agents to enhance Web services or integrating the two approaches. For a precise comparison between these two concepts see for example [Mor02]. [Huh02] points out some drawbacks of Web services that significantly distinguish them from agents: they know only about themselves, and they do not possess any meta-level awareness; they are not designed to utilize or understand ontologies; and they are not capable of autonomous action, intentional communication, or deliberatively cooperative behaviour. According to us, different kind of activities may be distinguished in agent-Web service integration:

Distinct view of agents and Web services. Agents are able both to describe their services as Web services and to search/use Web services by using mappings between MAS standards and SOA standards [LRCSN03, GC04]. This approach is often based on a gateway or wrapper which transforms a standard into another one. As the main approach in agent standardisation is the FIPA's one, these works always consider only FIPA agents and settle relationships between SOA and FIPA standards. A particular difficult issue of this approach is communication. The challenge consists in bridging the gap between asynchronous behaviour of agent communication and synchronous behaviour of Web services.

Uniform view of agents and Web services. Agents and Web services are the same entities. All services are Web services and they are all provided by agents (that means that the underpinning program application is an agent based system) [IYT04, Pet05].

MAS based SOA. MAS to support SOA architecture. This approach is not directly interested in agent service - Web service interaction but rather on the use of MAS to enhance SOAs. For example, [EMM03] discusses the use of agents for Web services selection according to the quality of matching criteria and ratings.

MAS based Business Process Management. Detailed hereafter.

2.4 GRID MAS analogies

2.4.1 Agent communication vs. BPM and service interaction

GRID and MAS use the same communication principle. In SOA, methods of the services are not directly invoked by users, but when messages are received, a service provider decides how to proceed by interpreting the user's message and invoking itself the good method(s). This is an important difference with software component based approaches. This is called direct message passing based communication, and it was originally suggested by Hewitt [Hew77]. MAS communication is also based on message passing[2].

Workflow or service orchestration (in Web and GRID) is analogue to interaction protocol in agent communication [Hug03]. Both terms describe a common communication principle based on interaction protocols that specifies a set of intermediate states in the communication process as well as transition between these states. The applicability of MAS to workflow enactment has been noted by [SH99]. More specifically [BVV03] makes a strict comparison between workflow (in BPEL4WS) and interaction protocol (as FIPA defined them).

Conversation of service or choreography is analogue to agent conversation. Conversation are long lived high level interactions which need a peer-to-peer, proactive, dynamic and loosely coupled mode of interaction. Using agent conversation to enhance service scenarios was currently only suggested within the Web service community [MML05, HNL02, DCG05]. Especially, [AGP03] suggests to use a dialogic agent communication approach. The authors do not explain how to represent Web services by agents, but propose a conversational model (speech act inspired) that is not based on a diagram of messages to be sent (i.e., interaction protocol) but rather on local operation calls: the interaction is modelled as a sequence of turns where one of the peers requires that the other peer performs an operation. The service provider has to maintain a set of context explicit interaction contexts, corresponding to each of its user. [AGP03] ideas are very closed to the ones adopted by the STROBE model.

2.4.2 Agent autonomy and intelligence vs. stateful and dynamic Grid service

"Intelligent agents" means that they are able to keep their own internal state and make it evolve in a way not necessarily dependent from the messages they receive; for example with machine learning algorithms. In the same sense, Grid services are stateful i.e., they own their running context, where the contextual state memory is stored. The analogy should also be made considering that an agent having a conversation dedicates a context (i.e., a part of its state) to this conversation like a Grid service factory which instantiates a new service instance with its own state. This idea is fundamental in our integrated model, presented in section 4.

"Autonomous agents" means that they are able to manage their own resources. This is analogue to Grid service lifetime management which allows them to be dynamic and manage themselves the resources they are allocated to.

[2] Always asynchronous for MAS and both synchronous and asynchronous for Grid service.

2.4.3 Organisational structure

As the numbers of entities in the system increases, as the tackled problems augments, and as the system is more and more distributed, different perspectives appear concerning how to define the system behaviour. Instead of being centred on how each of these entities should behave, one may alternatively adopt the perspective of organizations. An organization is a rule based partitioned structure in which actions can be lead (problem solving, interaction, etc.) by people playing roles and sharing one or more goals. Both GRID and MAS choose an organizational perspective in their description.

In OCMAS (Organizational Centred MAS) [WJK00, FGM03] the concepts of organizations, groups, roles, or communities play an important role. Especially, [FGM03] presents the main drawbacks of agent centred MAS and proposes a very concise and minimal OCMAS model called AGR for Agent/Group/Role. We will base our integration on this simple but very expressive model summarized in table 1.

Table 1: Organisational structure analogies between GRID and MAS

MAS	GRID
Agent	Grid user
An agent is an active, communicating entity playing roles within groups. An agent may be a member of several groups, and may hold multiple roles (in different groups).	A Grid user is an active, communicating entity providing and using services within a VO. A Grid user may be member of multiple VOs, and may provide several services (in different VOs).
Group	VO
A group is a set of (one or several) agents sharing some common characteristics and/or goals. A group is used as a context for a pattern of activities and for partitioning organizations. Two agents may communicate only if they are members of the same group. An agent transforms some of its capabilities into roles (abstract representation of functional position) when it integrates a group.	A VO is a set of (one or several) Grid users sharing some common objectives. A VO and the associate service container is used as a context for services execution and for partitioning all the community of Grid users. Two Grid users may exchange (provide/use) services only if they are members of the same VO. A Grid user publish some of its capabilities into services when it integrates a VO.
Role	CAS(Community Authorisation Service)
The role is the abstract representation of a functional position of an agent in a group. A role is defined in a group structure. An agent may play several roles in several groups. Roles are local to groups, and a role must be requested by an agent. A role may be played by several agents.	The service is the abstract representation of a functional position of a Grid user in a VO. A service is accessible via the CAS service. A Grid user may provide several services in several VOs. Services are local to VO (situated in the associate container), and a Grid user must be allowed to publish services in a VO. A service may be provided by several Grid users.

3 A common formalisation of GRID and MAS

3.1 The concept of service

We define a service by identifying a number of key criteria that characterise it. Figure 2 summarizes these criteria. Some criteria are taken as hypothesis: we only consider services deployed on SOA (3), which are standard compliant and publishable (4), and which may be used by any kind of entities (9) (as they are standardly interfaced). We further encourage and expect semantic services (8) but we do not take it as a hypothesis. Some criteria (2, 5, 6, 7) have graphical representations some not (1) as figure 3 shows. The case of criterion (10) is a little bit special as orchestration of services may be viewed as another service (such a BPEL4WS workflow of Web services itself defined as a Web service) and choreography may be viewed as a set of services interacting together and with different users (themselves service providers).

Figure 3 presents services we aim to formalise and their associated symbols. Stateless services are quite restrictive: they are synchronous (i.e, messages can not be buffered and do block the sender or receiver), point-to-point (i.e., used by only one user) and interact via simple one shot interaction (i.e., request/answer). A stateless service does not establish a conversation. Instead, it returns a result from an invocation, akin of function. Stateful

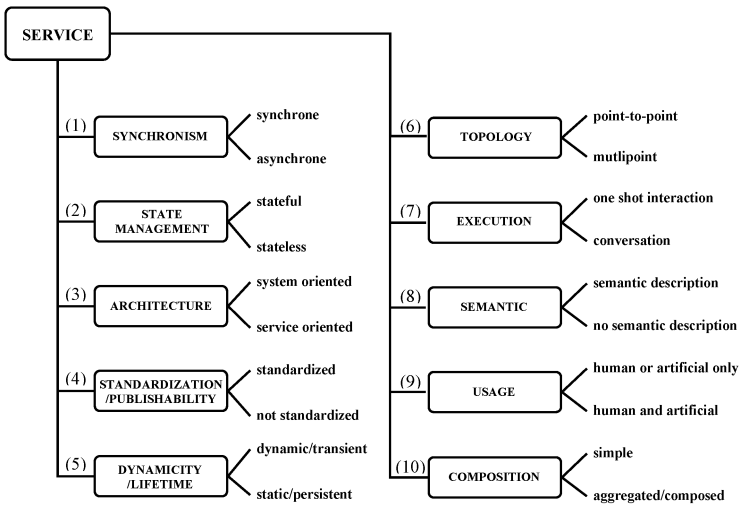

Figure 2: Taxonomy of services

services deserve more consideration: They can be persistent or transient[3]. Transient services are instantiated by a service factory where as persistent services are created by out-of-band mechanisms such as the initialisation of a new service container. Stateful services may be multipoint i.e., used by several users and may interact by simple one shot interaction or long lived conversation[4]. Stateful services may be synchronous or a asynchronous.

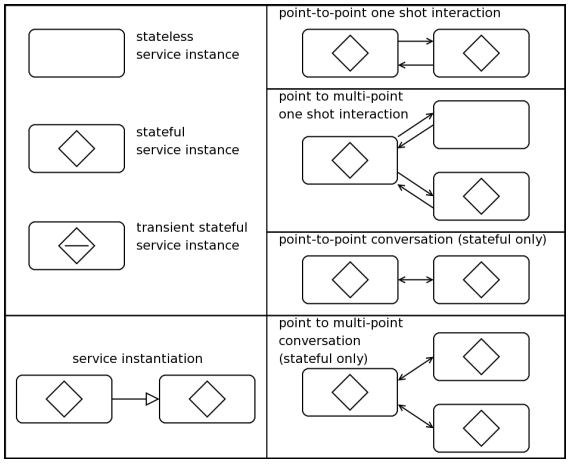

Figure 3: Service key concepts representation

3.2 GRID key concepts

GRID key concepts (figure 4) have been introduced either by the OGSA or WSRF specifications. As our concern is to focus on concepts, we adopt the most convenient terminology between both sets of specifications.

GRID is a resource sharing system. Grid resources are brought by *hosts*. A host is either a direct association between a *computing resource* and a *storage resource* or a *host coupling*. The sharing of these resources is realised by a set of *virtualization units* and a reification[5] of these resource in *service container*. Services need a hosting

[3]Transient (or dynamic) services are dynamically instantiated for a given period of time. This period may change dynamically.

[4]Service users are represented in figure 3 by other services for a sake of simplicity. In our integrated model there are all considered to be agents.

[5]Resource virtualization and reification is done at the core GRID level (middleware). The rest of GRID core level mechanisms (e.g. container, CAS, etc.) are themselves described with a single unit: the Grid service.

environment to exist and evolve with their own private context (i.e., set of resource). This is the role of the service container which is the reification of a portion of the virtualized resource available in a secure and reliable manner. A service container contains several types of services. A service may instantiate another one in the same or another service container. Each service is identified by a *handle*. Since a container is a particular kind of service it is created either through the use of a service factory or by direct core GRID mechanism. A service container is allocated to (and created for) one and only one group of Grid *users*, called a *Virtual Organization* (VO). Each Grid user may be a *member* of several VO. The relation between a VO and a service container is embodied by a *Community Authorisation Service* (CAS) which formalises the VO dedicated policies of service by members. The CAS may be viewed as a MxS matrix, where M corresponds to the number of members of the VO, S to the number of currently active service, and the matrix nodes are deontic rules. These rules enable to accurately specify the right levels for a member on a service (e.g., permissions, interdictions, restrictions etc.). In order to participate in GRID, hosts and Grid users must hold a *X509 certificate* signed by a special authority. Entities with X509 certificate are sometime called Grid *node*. These set of nodes and authorities form Grid *trusts*.

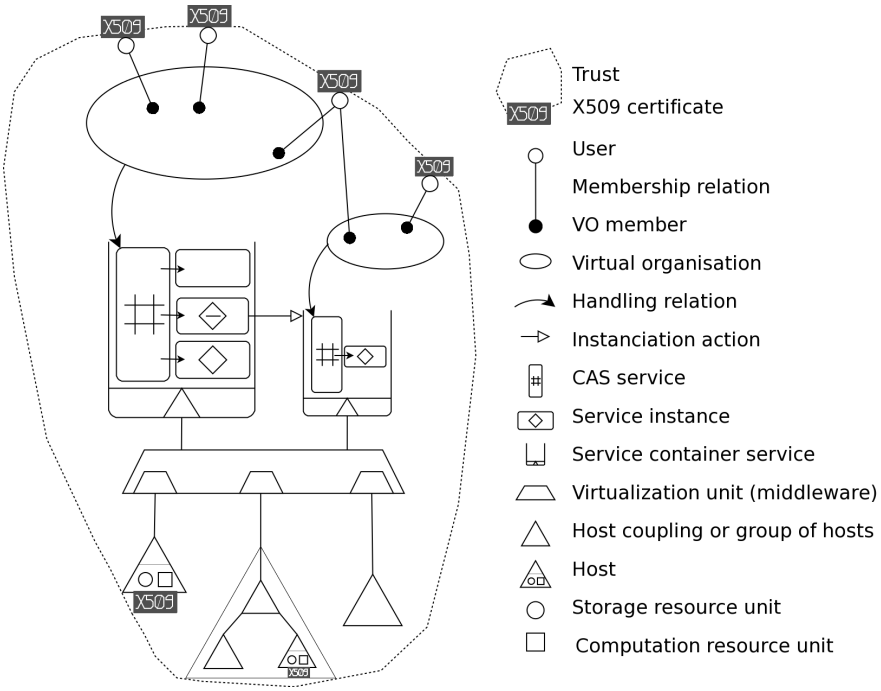

Figure 4: GRID key concepts representation

3.3 MAS key concepts

MAS key concepts (figure 5) have been introduced by different MAS approaches especially [JC05, FGM03]. Here again, as our concern is to focus on concepts, we adopt the most convenient and simple terminology between all these approaches.

An *agent* possesses both intelligent and functional abilities. There are represented respectively by the agent *brain* and *body*. The brain is composed of set of rules and algorithms (e.g. machine learning) that give to the agent learning and reasoning skills. It also contains the agent knowledge, objectives, and mental states (e.g., BDI). The body is composed of a set of *capabilities* which correspond to the agent capacity or ability to do something, to perform some task. It is a first class black box abstraction that can be stored, named, called with arguments and that return a result. These stateful or stateless capabilities may be executed in a particular context called a *cognitive environment*. An agent may have several cognitive environments which correspond to the different languages it develops by *interaction* with other agents. Agents are themselves structured in *groups* and have roles according to their capabilities.

This agent representation unifies the Artificial Agent (AA) and Human Agent (HA) representations. Agents are the current best metaphor for humans in computing. HA are of course autonomous, intelligent and interactive;

we can consider that they have set of capabilities as well as dedicated context for conversations. Of course this is a simple (and restrictive) HA consideration.

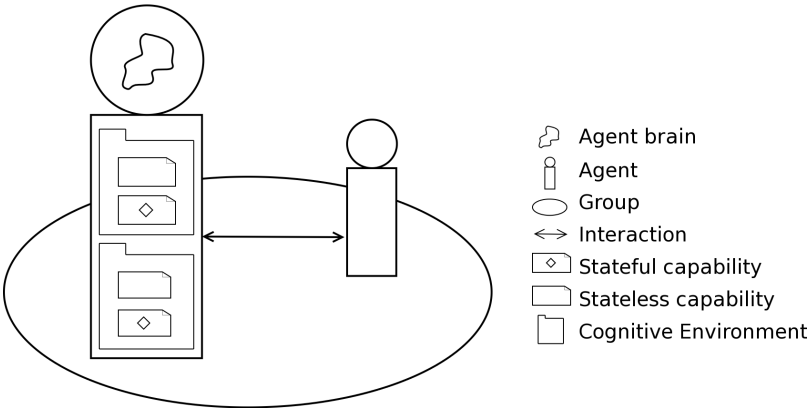

Figure 5: MAS key concepts representation

We may detail a little bit more the concept of *cognitive environment* (CE), which is the most important one and quite new concept related to the STROBE agent and communication model [Cer99, JC05]. Conversations and their states are represented in the STROBE model by cognitive environments[6]. STROBE agents are able to interpret communication messages in a given CE, including an interpreter, dedicated to the current conversation. We showed in [JC05] how communication enables to dynamically change values in a CE and especially how these interpreters can dynamically adapt their way of interpreting messages (meta-level learning by communication). By representing a CE as a context composed of a pair [environment, interpreter] we use to say that they correspond to different languages[7] an agent develop by interaction with other agents. The same concept of putting at the centre of the agent architecture the communication contexts in which it interprets messages appears also in [AGP03] which assumes that each agent in service exchanges may separately maintains its own internal context of the conversation state.

Actually, a STROBE agent has two types of CE: i) A *global* one, unique and private, which belongs to the agent and represents its own believes and generic capabilities; ii) Some *local* ones, dedicated to a specific interlocutor or group of interlocutors. Each time an agent receives a message, its selects the unique corresponding CE dedicated to the message sender in order to interpret the message. When a STROBE agent receive a message for the first time it instantiates a new local CE for this agent following three policies: i) copying the global CE; ii) copying a local CE; iii) sharing a local CE. The fact of having, in the STROBE model, dedicated contexts and thus dedicated capabilities is for us the key element to realise DSG.

4 Toward a GRID-MAS integrated model

4.1 Status of current integration activities

The research activity in GRID and MAS convergence is increasingly active[8]. Using MAS principles to improve core GRID performances (e.g. directory services, scheduling, brokering services, task allocation, dynamic resource allocation and load balancing) is a very active topic in the MAS community, for example:

- MAS based GRID for resource management [MT99, GHCN99, WOvSB02, CJS+02, CSJN05, SLGW02, MBP05];

[6]The term environment is here used with its programming language meaning, that is to say, a structure that binds variables and values. It does not mean the world surrounding an agent.

[7]Highly influenced by the functional and applicative programming languages community (e.g. Scheme, LISP), we assume that a language is basically a pair consisting of: i) a language expression evaluation mechanism and ii) a memory to store these mechanisms and abstractions constructed with the language.

[8]See for example, "Agent-Based Cluster and Grid Computing" workshops, "Smart Grid Technologies" workshops, the MultiAgent and Grid System journal.

- MAS based GRID for VO management [NPC+04, PTJ+05].

However all this related works are not proposing an integration of MAS and GRID. Rather, they focus on how MAS and AI techniques may enhance GRID core functionalities. Our vision of a GRID-MAS integration goes beyond a simple use of one technology to enhance the other one. Our vision, to benefit of the most relevant aspects of both GRID and MAS, is to adopt a common approach for the integration. This common approach is centred on the concept of service. One of the crucial explorations concerns the substitution of an agent-oriented kernel to the current object-oriented kernel of services available on the GRID. Some concepts can be directly mapped between GRID and MAS; some other ones can not as the next section shows.

4.2 Mapping of GRID and MAS concepts

The mapping of GRID and MAS key concepts concerns five aspects.

1. The term *agent* is used to denote uniformly Artificial Agent, Human Agent and Grid user. Especially, by viewing Grid users as agents, we may considered them as a potential artificial entities.

2. The term *VO* unifies the concept of VO in GRID and the concept of group in MAS. Thus we talk now about "VO of agents".

3. The two concepts of *service* and *capability* stay true, however we add a new one-to-one relation between them called an *allocated interface relation* (represented by a dotted line figure 6). A service is viewed as the interface of a capability published in a service container and with allocated resources. An agent has a set of capabilities it may transform into services available in the different VOs it is a member of. To "transform" a capability into service is called the *servicisation process*[9]. When a capability is servicised it means:

   - to interface this capability with the SOA standards i.e., mainly WSDL/SOAP;
   - to add (eventually by using a add-service service) this service in the VO service container by assigning it a handle and by allocating it private resources;
   - to request the VO CAS service to add an entry for this service (the agent has to decide the user's right levels);
   - to publish the service description in the VO register if it exists;
   - to notify members of the VO that a new service is available;
   - etc, according to VO local rules.

   This servicisation process is not discrete but continuous. For example, if the capability of the agent changes then the service changes at the same time. Consider also that service right levels in the VO CAS or service allocated resources may evolve through time. With this view, an agent can provide different services in different VOs. Notice also that a service is agent specific, that means that only one agent can execute the service in a container. However, this does not prevent that another agent of the VO provides the same type of service.

   What is important in this servicisation process is that it is the same whatever is the kind of agent implied. Both AA and HA transform their capability in the VO service container modulo a graphical interface. For example, an AA may publish its capability to compute square roots (i.e., a function that receives a number as a parameter and returns a float as result), and a HA may publish its pattern recognition capability (i.e., a function that receives an image as a parameter and returns a concept as result (described in an ontology)).

4. The GRID key idea of service instantiation is mapped with the STROBE cognitive environment instantiation mechanism. It means that instantiating a new service in GRID means to instantiate a new CE in MAS; this is the same process but viewed differently (figure 6). The capability corresponding to the new service instance is located in the new local CE[10]. The service benefits of a standard interface and allocated resources from GRID, and of a dedicated context of execution and local conversation representation from MAS.

[9] We should say that as GRID virtualizes resources and reifies them in a service container, an agent virtualizes capabilities and reifies them in a service container.

[10] In order to exactly map the STROBE mechanisms with OGSA and WSRF mechanisms we should say that a new CE may be viewed as a new WS-Resource i.e., a CE is a dedicated association between capabilities and stateful resources.

5. Agent-agent interaction includes all other kinds of interaction (Grid user-Grid service, Grid service-Grid service, agent-agent etc.) (cf. figure 7. They may be:

   Direct agent-agent interaction. Communication message exchanges directly from agent to agent. These are interactions in a general sense, i.e., any interactions, standardised or ad-hoc, protocol guided or not, semantically described or not, long lived or one shot etc. These interactions may occur within a VO, but also outside.

   Service thru agent-agent interaction. Communication message exchanges from agent to agent through a service. These are interactions that an agent may have with a service without directly request the agent but the service interface this agent proposes in the VO service container. These service thru interactions occur only within a VO.

In order to avoid the problem of mapping between SOAP and ACLs[11] [LRCSN03, GC04] we do not consider that there is a transformation of message (or any interaction) between the service allocated interface and the agent capability; they are the same thing represented differently. Agents are supposed to be able to interpret directly SOAP messages corresponding to the capabilities they servicised in WSDL.

What is important in this integrated model is to consider how a service may be adapted to serve user agent(s) in order to realise DSG. We identify three ways:

- The agent service provider may propose another service to change or adapt the first one;
- The agent service provider may use dynamic intelligent reflection rules to change the service it is currently providing;
- A direct agent-agent interaction may occur between the agent service user and the agent service provider.

4.3 Towards an OGSA-STROBE infrastructure

4.3.1 Sum-up of the integration

In our integrated model we consider agents exchanging services through VOs they are member of: both service user and service provider are considered to be agents. They may decide to make available one of their capabilities in a certain VO but not in another one. The VO service container is then used as a service publish/retrieve platform (the semantic may also be situated there). A service is executed or engaged, by the agent which proposes it but it uses resources allocated by the service container. Figure 6 shows the elements of this integration.

4.3.2 Discussions and benefits for GRID, MAS and SOC

- According to the STROBE agent policy, when a new local CE is created at least one capability (corresponding to the service used) may be copied in the CE but also all capabilities corresponding to services the agent may provide in this VO. Even if these services have not been used yet, they may help the STROBE agent to combine, or compose, its capabilities one another. Dedicated service composition is a real means to go toward DSG.
- There is no real standard in the MAS community to describe agent capability. The integration will help MAS developers to present and interface agent capability and therefore augment MAS interoperation.
- This integrated model is absolutely not restrictive for MAS. It does not prevent direct agent-agent interaction and thus, for example, it doest not prevent agents to perform tasks one another in a purely ad-hoc way. This is important if we want the integration to be followed by other MAS approaches and models: these models can keep their internal formalisms for their internal operation.
- VO management, in this integration, benefits from both GRID and MAS organizational structure formalism e.g., AGR, CAS service, X509 certificate etc.

[11] The biggest risk in attempting ACL-SOAP integration is not to reduce complex and semantically richer agent communication to simple request/response of Web service.

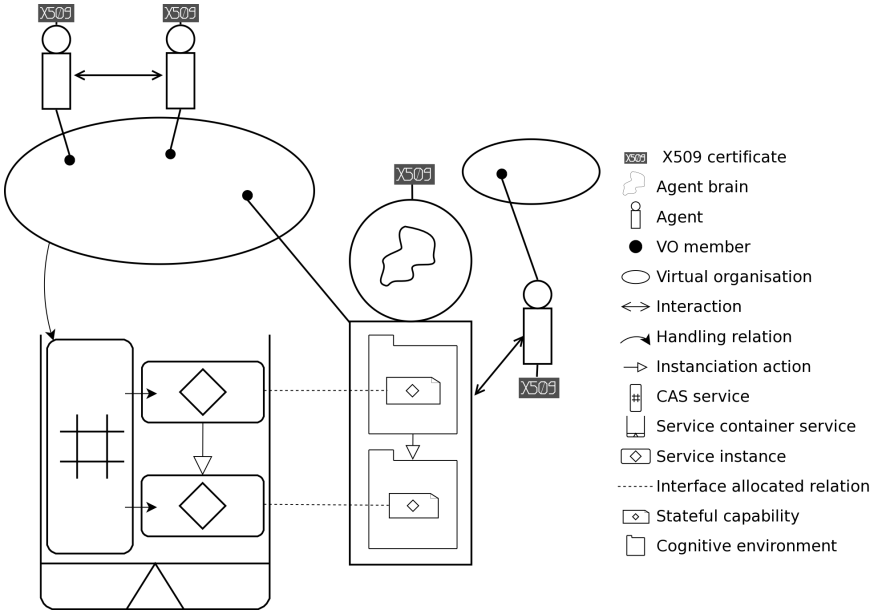

Figure 6: Integration of GRID and MAS

- Service exchanges in this integrated model benefit from the important agent communication abilities e.g., dealing with semantics, able to have conversation, etc. [Hug03]. The challenges of modelling conversation not by fixed structure (interaction protocol) but with dynamic dialogue becomes quite next to the challenge of dynamically composing and choreographing services in business processes such as it is suggested by DSG.

- Moreover, our vision of integration subsumes an important part of the MAS based GRID approaches cited in section 4.1. Indeed, thank to the GRID specification which defines some GRID core functionalities as (meta-)Grid service (e.g. service container, CAS) we may see these core GRID services executed also by agents. This realises an important part of MAS based GRID approaches which use MAS techniques to enhanced core GRID functionalities.

5 Conclusion and perspectives

Identifying key factors to demonstrate the convergence of MAS and GRID models is not an easy task. We pointed that the current state of GRID and MAS research activities have reached a sufficient maturity to enable justify exploring the path towards an integration of the two domains. At the crossing point of this integration stands the concept of service. The bottom-up vision of service in GRID with the top-down vision of service in MAS brings up a richer concept of service which integrates both GRID and MAS properties. We put this enhanced concept of service into the perspective of Dynamic Service Generation (DSG).

Besides describing the why GRID and MAS need each other, we explain how they can be synergic. Through an analysis of concrete models (mainly OGSA and STROBE) we extracted some key concepts and presented a set of GRID and MAS analogies. We proposed a new kind of (graphic) formalisation that we adopted to describe part of DSG mechanisms. Then, we proposed an integrated model inspired by the analogies that respects all the constraints and foundations of both GRID and MAS.

The GRID-MAS integrated model concepts can be summarised by analyzing the interactions in this model (figure 7). The most significant contribution in this paper that links together all the elements of figure 7 is the relation introduced between a Grid service instance and its state with an agent capability and its context. OGSA fits well for the fisrts part of the interaction; the STROBE model fits well for the second one.

The integration proposed in this paper is feasible considering today's state of GRID and MAS technologies. However, future developments of GRID and MAS may consider this integration in order to progress. For examples, hereafter, two aspects of the STROBE model and two aspects of the OGSA model that should evolve:

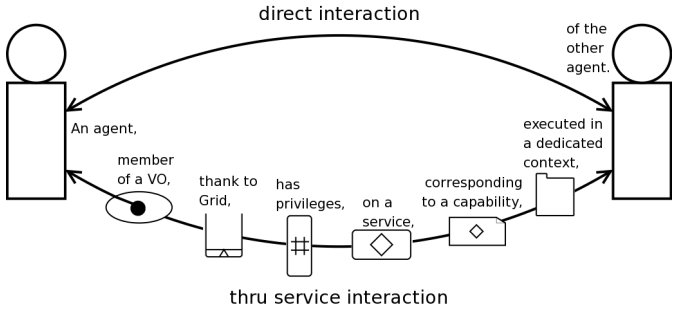

Figure 7: GRID-MAS integration loop of interaction

Better support for more than one interlocutor dedicated for a CE. Most of the scenarios using the STROBE model take as a rule that STROBE agents must have a local CE dedicated to a group of only one interlocutor. However, this view is non adequate in a GRID-MAS integration because it would imply that each service instance in a service container can be used by one and only one agent. Therefore the STROBE model should evolve to fit better the cases where a local CE is used for a group of interlocutors.

Ability for agent to duplicate. The STROBE model presupposes an agent have only one local CE for a given interlocutor i.e., interlocutor messages are always interpreted in the same CE (one personally dedicated or not). This is better for DSG to have one and only one conversation context for an interlocutor. Therefore, STROBE agents must become able to duplicate themselves (fork) in order to keep true this presupposition.

Support for non synchronous protocols and better support of the semantics. GRID and SOA standards must evolve to fit better agent communicating properties. Especially, the question of synchronicity (HTTP and thus SOAP are still synchronous communication protocols) and the question of semantics (which starts with the work of the Semantic Web community).

Finally, GRID and MAS communities, mostly industrial for the former and mostly academic for the latter addressed the question of service in distributed system with a completely different angle and thus developed different complementary aspects. Integrating these aspects following the guidelines given in this paper seems for us the best way to capitalise past, present and future work in order to simplify the scenarios and exploit concretely the power of distributed services, exchanged among communities of humans and machines.

6 Acknowledgment

Work partially supported by the European Community under the Information Society Technologies (IST) programme of the 6^{th} Framework Programme for RTD - project ELeGI, contract IST-002205. This document does not represent the opinion of the European Community, and the European Community is not responsible for any use that might be made of data appearing therein.

References

[ACR+05] Colin Allison, Stefano A. Cerri, Pierluigi Ritrovato, Angelo Gaeta, and Matteo Gaeta. Services, Semantics and Standards: Elements of a Learning Grid Infrastructure. *Applied Artificial Intelligence Journal, Special issue on Learning Grid Services*, 19(9-10):861–879, November 2005.

[AGP03] Liliana Ardissono, Anna Goy, and Giovanna Petrone. Enabling Conversations with Web Services. In *2^{nd} International Joint Conference on Autonomous Agents and Multi-Agent Systems, AAMAS'03*, pages 819–826, Melbourne, Australia, July 2003. ACM Press.

[BVV03] Paul A. Buhler, José M. Vidal, and Harko Verhagen. Adaptive Workflow = Web Services + Agents. In *International Conference on Web Services, ICWS'03*, pages 131–137, Las Vegas, USA, July 2003. CSREA Press.

[Cer99] Stefano A. Cerri. Shifting the Focus from Control to Communication: the STReam OBjects Environments Model of Communicating Agents. In J.A. Padget, editor, *Collaboration between Human and Artificial Societies, Coordination and Agent-Based Distributed Computing*, volume 1624 of *Lecture Note in Artificial Intelligence*, pages 74–101. Springer-Verlag, Berlin, 1999.

[CJS+02] Junwei Cao, Stephen A. Jarvis, S. Saini, D. J. Kerbyson, and Graham. R. Nudd. ARMS: an Agent-based Resource Management System for Grid Computing. *Scientific Programming, Special Issue on Grid Computing*, 10(2):135–148, 2002.

[CSJN05] Junwei Cao, Daniel P. Spooner, Stephen A. Jarvis, and Graham R. Nudd. Grid load balancing using intelligent agents. *Future Generation Computer Systems*, 21(1):135–149, January 2005.

[CT03] Mario Cannataro and Domenico Talia. The Knowledge Grid. *Communications of the ACM*, 46(1):89–93, 2003.

[CTT05] Carmela Comito, Domenico Talia, and Paolo Trunfio. Grid Services: Principles, Implementations and Use. *International Journal of Web and Grid Services*, 1(1):48–68, 2005.

[DCG05] John Domingue, Liliana Cabral, and Stefania Galizia. Choreography in IRS-III - Coping with Heterogeneous Interaction Patterns in Web Services. In *4^{th} International Semantic Web Conference, ISWC'05*, Galway, Ireland, November 2005.

[EMM03] Munindar P. Singh E. Michael Maximilien. Agent-based architecture for autonomic Web service selection. In *1^{st} International Workshop on Web Services and Agent Based Engineering, WSABE'03*, Sydney, Australia, July 2003.

[Fer99] Jacques Ferber. *Multi-Agent Systems: An Introduction to Distributed Artificial Intelligence*. Addison Wesley Longman, Harlow, UK, 1999.

[FFG+04] Ian Foster, Jeffrey Frey, Steve Graham, Steve Tuecke, Karl Czajkowski, Don Ferguson, Frank Leymann, Martin Nally, Igor Sedukhin, David Snelling, Tony Storey, William Vambenepe, and Sanjiva Weerawarana. Modeling Stateful Resources with Web Services. Whitepaper Ver. 1.1, Web Services Resource Framework, May 2004.

[FGM03] Jacques Ferber, Olivier Gutknecht, and Fabien Michel. From Agents to Organizations: An Organizational View of Multi-agent Systems. In P. Giorgini, J. P. Müller, and J. Odell, editors, *4^{th} International Workshop on Agent-Oriented Software Engineering, AOSE'03*, volume 2935 of *Lecture Notes in Computer Science*, pages 214–230. Springer-Verlag, Berlin, 2003.

[FJK04] Ian Foster, Nicholas R. Jennings, and Carl Kesselman. Brain Meets Brawn: Why Grid and Agents Need Each Other. In *3^{rd} International Joint Conference on Autonomous Agents and Multiagent Systems, AAMAS'04*, volume 1, pages 8–15, New York, NY, USA, July 2004.

[FKNT02] Ian Foster, Carl Kesselman, Jeff Nick, and Steve Tuecke. The Physiology of the Grid: An Open Grid Services Architecture for Distributed Systems Integration. In *Open Grid Service Infrastructure WG, Global Grid Forum*. The Globus Alliance, June 2002.

[FKT01] Ian Foster, Carl Kesselman, and Steve Tuecke. The Anatomy of the Grid: Enabling Scalable Virtual Organizations. *Supercomputer Applications*, 15(3), 2001.

[GC04] Dominic Greenwood and Monique Calisti. Engineering Web Service - Agent Integration. In *IEEE Systems, Cybernetics and Man Conference*, The Hague, Netherlands, October 2004.

[Gel04] Marije Geldof. The Semantic Grid: will Semantic Web and Grid go hand in hand? Technical report, European Commission DG Information Society Unit 'Grid technologies', June 2004.

[GHCN99] R. Ghanea-Hercock, J. C. Collis, and D. T. Ndumu. Co-operating mobile agents for distributed parallel processing. In *3^{rd} International Conference on Autonomous Agents*, pages 398–399, New York, NY, USA, 1999. ACM Press.

[Hew77] Carl Hewitt. Viewing Control Structures as Patterns of Passing Messages. *Artificial Intelligence*, 8(3):323–364, June 1977.

[HNL02] James E. Hanson, Prabir Nandi, and David W. Levine. Conversation-enabled Web Services for Agents and e-business. In *3rd International Conference on Internet Computing, IC'02*, pages 791–796, Las Vegas, Nevada, USA, June 2002.

[HS98] Michael N. Huhns and Munindar P. Singh, editors. *Readings in Agents*. Morgan Kaufmann, San Francisco, California, 1998.

[Hug03] Marc-Philippe Huget, editor. *Communication in Multiagent Systems, Agent Communication Languages and Conversation Policies*, volume 2650 of *Lecture Notes in Computer Science*. Springer-Verlag, Berlin, 2003.

[Huh02] Michael N. Huhns. Agents as Web Services. *IEEE Internet Computing*, 6(4):93–95, 2002.

[IYT04] Fuyuki Ishikawa, Nobukazu Yoshioka, and Yasuyuki Tahara. Toward Synthesis of Web Services and Mobile Agents. In *2nd International Workshop on Web Services and Agent Based Engineering, WSABE'04*, pages 48–55, New York, NY, USA, July 2004.

[JC05] Clement Jonquet and Stefano A. Cerri. The STROBE model: Dynamic Service Generation on the Grid. *Applied Artificial Intelligence Journal, Special issue on Learning Grid Services*, 19(9-10):967–1013, November 2005.

[JC06] Clement Jonquet and Stefano A. Cerri. Characterization of the Dynamic Service Generation Concept. Research Report 06007, LIRMM, CNRS & University Montpellier II, France, February 2006. www.lirmm.fr/~ jonquet/Publications/.

[Jen01] Nicolas R. Jennings. An agent-based approach for building complex software systems. *Communications of the ACM*, 44(4):35–41, 2001.

[KWvL02] Sriram Krishnan, Patrick Wagstrom, and Gregor von Laszewski. GSFL : A Workflow Framework for Grid Services. Draft paper, Globus Alliance, July 2002.

[LRCSN03] Margaret Lyell, Lowell Rosen, Michelle Casagni-Simkins, and David Norris. On Software Agents and Web Services: Usage and Design Concepts and Issues. In *1st International Workshop on Web Services and Agent Based Engineering, WSABE'03*, Melbourne, Australia, July 2003.

[MBP05] S. S. Manvi, M.N. Birje, and B. Prasad. An Agent-based Resource Allocation Model for computational grids. *Multiagent and Grid Systems*, 1(1):17–27, 2005.

[MML05] Zakaria Maamar, Soraya K. Mostéfaoui, and Mohammed Lahkim. Web services composition using software agents and conversations. In D. Benslimane, editor, *Les services Web*, volume 10 of *RSTI-ISI*. Lavoisier, 2005.

[Mor02] Luc Moreau. Agents for the Grid: A Comparison with Web Services (Part 1: the transport layer). In H. E. Bal, K-P. Lohr, and A. Reinefeld, editors, *Second IEEE/ACM International Symposium on Cluster Computing and the Grid, CCGRID'02*, pages 220–228, Berlin, Germany, May 2002. IEEE Computer Society.

[MT99] Frank Manola and Craig Thompson. Characterizing the Agent Grid. Technical report 990623, Object Services and Consulting, Inc., June 1999.

[NPC$^+$04] T. J. Norman, A. Preece, S. Chalmers, N. R. Jennings, M. Luck, V. D. Dang, T. D. Nguyen, V. Deora, J. Shao, A. Gray, and N. Fiddian. Agent-based formation of virtual organisations. *Knowledge Based Systems*, 17(2-4):103–111, 2004.

[Pet05] Jan Peters. Integration of Mobile Agents and Web Services. In *1st European Young Researchers Workshop on Service Oriented Computing (YR-SOC'05)*, pages 53–58, Leicester, UK, April 2005. Software Technology Research Laboratory, De Montfort University.

[PTJ$^+$05] J. Patel, W. T. L. Teacy, N. R. Jennings, M. Luck, S. Chalmers, N. Oren, T. J. Norman, A. Preece, P. M. D. Gray, G. Shercliff, P. J. Stockreisser, J. Shao, W. A. Gray, N. J. Fiddian, and S. Thompson. Agent-Based Virtual Organisations for the Grid. In *AAMAS'05 First International Workshop on Smart Grid Technologies, SGT'05*, Utrecht, The Netherlands, July 2005.

[RJS01] David De Roure, Nicholas Jennings, and Nigel Shadbolt. Research Agenda for the Semantic Grid: A Future e-Science Infrastructure. Technical report, University of Southampton, UK, June 2001. Report commissioned for EPSRC/DTI Core e-Science Programme.

[RM00] Omer F. Rana and Luc Moreau. Issues in Building Agent based Computational Grids. In *3^{rd} Workshop of the UK Special Interest Group on Multi-Agent Systems, UKMAS'00*, Oxford, UK, December 2000.

[SH99] Munindar P. Singh and Michael N Huhns. Multiagent Systems for Workflow. *Intelligent Systems in Accounting, Finance and Management*, 8:105–117, 1999.

[SH05] Munindar P. Singh and Michael N. Huhns. *Service-Oriented Computing Semantics, Processes, Agents*. John Wiley & Sons, Ltd, 2005.

[SLGW02] W. Shen, Y. Li, H. Ghenniwa, and C. Wang. Adaptive Negotiation for Agent-Based Grid Computing. In *AAMAS'02 Workshop on Agentcities: Challenges in Open Agent Environments*, pages 32–36, Bologna, Italy, 2002.

[WJK00] Michael Wooldridge, Nicholas R. Jennings, and David Kinny. The Gaia Methodology for Agent-Oriented Analysis and Design. *Autonomous Agents and Multi-Agent Systems*, 3(3):285–312, 2000.

[Woo02] Michael Wooldridge. *An Introduction to Multiagent Systems*. John Wiley & Sons, Chichester, UK, February 2002.

[WOvSB02] N.J.E. Wijngaards, B.J. Overeinder, M. van Steen, and F.M.T. Brazier. Supporting internet-scale multi-agent systems. *Data & Knowledge Engineering*, 41(2-3):229–245, June 2002.