Agents in grid extended to clouds

K. Wasielewska, M. Ganzha, M. Paprzycki, C. Bădică, M. Ivanovic, I. Lirkov, and S. Fidanova

Citation: AIP Conference Proceedings **1773**, 070002 (2016); doi: 10.1063/1.4964984
View online: http://dx.doi.org/10.1063/1.4964984
View Table of Contents: http://scitation.aip.org/content/aip/proceeding/aipcp/1773?ver=pdfcov
Published by the AIP Publishing

Agents in Grid Extended to Clouds

K. Wasielewska[1,a)], M. Ganzha[1,2], M. Paprzycki[1,3], C. Bădică[4], M. Ivanovic[5], I. Lirkov[6] and S. Fidanova[6]

[1]*Systems Research Institute Polish Academy of Sciences, Warsaw, Poland*
[2]*Warsaw University of Technology, Warsaw, Poland*
[3]*Warsaw Management Academy, Warsaw, Poland*
[4]*Department of Computer Science, University of Craiova, Craiova, Romania*
[5]*Department of Mathematics and Informatics, Faculty of Sciences, University of Novi Sad, Novi Sad, Serbia*
[6]*Institute of Information and Communication Technologies, Bulgarian Academy of Sciences, Sofia, Bulgaria*

[a)]Corresponding author: Katarzyna.Wasielewska@ibspan.waw.pl

Abstract. The presented work is an attempt to extend considerations from the Agents in Grid (AiG) project to the Clouds. The AiG project is aimed at the development of an agent-semantic infrastructure for efficient resource management in the grid. Decision support within the AIG system helps the user, without in-depth knowledge, to choose optimal algorithm and/or resource to solve a problem from a given domain, and later to choose the best contract defining terms of collaboration with the provider of a resource used to solve the problem. Cloud computing refers to an architecture, in which groups of remote servers are networked, to allow online access to computer services or resources. The general vision is the same as in the case of computational grids, *i.e.*, to reduce cost of computing, as well as to increase flexibility and reliability of the infrastructure. However, there are also important differences. It is relatively easy to notice that solutions considered in the context of the AiG system can be easily extended to computational clouds that evolved from computational grids. As it was shown in the case of grids, integrating software agents, semantics and cloud computing could enable highly efficient, intelligent systems, making clouds even more flexible, autonomic and usable.

INTRODUCTION

The starting point for the presented considerations is the Agents in Grid (AiG; [1, 2, 3]) project that aims at the design and development of an agent-semantic infrastructure for efficient resource management and resource utilization in the grid. We claim that, since the general vision for the computational grids and computational clouds is "almost the same," many issues addressed in the AiG project can be extended to the clouds. We start with the description of basic characteristics of the AiG system, next we will discuss the relation between grid and cloud infrastructures, and finally we will map specific considerations for moving from the grid onto the cloud.

Agents in grid

The AiG project is structured around the following tenants:

- Grid is treated as an open environment that can be easily joined and used,
- System architecture is based on software agents and semantic data processing,
- Grid users can be divided into:
  - Resource providers that offer resources and are paid for their use,
  - Resource consumers (clients, end users) that commission job execution and pay for resource utilization,
- All users and resources are represented by software agents,
- Agents representing resources work in teams lead by managers,
- the Client Information Center (CIC) component stores (and delivers) information about the grid structure and available teams / resources.

Note that, users that want to commision job execution may have different level of expertise regarding required resource configuration and problem to be solved. In this context, decision support module was proposed that helps the user without in-depth knowledge to choose an optimal algorithm and/or resource to solve a problem and to choose the best contract, defining terms of collaboration with the resource provider. Specifically, we consider the multicriterial assessment of how accurate is the user description of her/his needs, taking into account complex nature of grid resource description and domain knowledge. User support should propose possible extensions / modifications of these descriptions. Note that, user support mechanisms in the AiG have been discussed in [4].

Decision support

The need for decision support originates in observation that two types of grid resource consumers can be identified: (i) active users (few) that are ready to explore possibilities and expand their toolbox of methods for completing their computational jobs, (ii) standard users (majority) that are only interested in solving their problems. The latter are not interested and willing to spend time learning new "things" related to the available computing infrastructure. Usually, they remember one particular way to submit a task which may be due to their minimal technical knowledge. As a result, there is a communication gap between new method creators / infrastructure developers and grid clients, which leads to continuous usage of one method / algorithm / library to solve a given problem even though better alternatives may exist.

The general job execution scenario consists of the following sequence of actions (without considering user support):

1. User specifies requirements for job execution (restrictions on resource configuration).
2. User's agent communicates with the CIC and obtains the list of agent teams that can execute the job.
3. User specifies requirements for the contract defining the terms of collaboration with resource provider.
4. FIPA Contract-Net Protocol is used to obtain job execution contract proposals and complete selection.
5. User's agent submits job to the selected team.

When we consider decision support, applying expert knowledge, the above sequence is re-defined as follows:

1. User specifies requirements for job execution (restrictions on resource configuration and problem description and, optionally, method and input data properties according to her/his knowledge).
   (a) User's agent searches in expert knowledge repository, selects pertinent expert opinions and chooses one that is the most representative. Three cases are possible:
      - user fully and "correctly" defined requirements (no modifications needed),
      - resource and problem specification could use some improvement,
      - resource and/or problem specification are "wrong" (alternatives should be proposed).

   (b) User accepts, or rejects, feedback based on the expert knowledge.
2. User's agent communicates with the CIC and obtains the list of agent teams that can execute the job.
3. User specifies requirements for the contract defining the terms of collaboration with resource provider.
4. FIPA Contract-Net Protocol is used to obtain job execution contract proposals and complete selection.
5. User's agent submits job to the selected team.

The multicriteria analysis for selection of a team (managing required resource), and a contract, is based on semantic analysis, where different methods can be used depending on the extent to which user support should be applied [4]. SPARQL queries provide the selection mechanism, in which results are not ranked and the selection criteria are treated as equally important. Class expression matching is a selection mechanism in which results are also not ranked and the selection criteria are treated as equally important, however, there is more expressiveness in query formulation. In a graph-based matching approach, ontology is treated as a directed graph, in which closeness between instances is measured (results are ranked, expert knowledge and user's expertise may be involved in the algorithm). Finally, the MCA approach, based on Analytical Hierarchy Process (AHP), with ontologically represented domain expert knowledge (for details, see [5, 6]) can also be used.

Ontologies

As it was already mentioned, all knowledge in the system is semantically represented and stored as ontologies (this approach is also recommended for clouds). Note that, contrary to syntax, semantics is focused on understanding, and constructing meaning representation (knowledge model). Specifically, ontology is a formal, explicit description of concepts in a domain and relationships between them [7]. An ontology, together with a set of individual instances of classes, constitutes a knowledge base. In the AiG the following ontologies were developed:

- Grid Ontology - concepts for description of grid structure and resources; extension of Core Grid Ontology [8],
- Conditions Ontology - concepts related to SLA representation and negotiation,
- Messages Ontology - concepts represent content of messages exchanged by agents,
- Expert Ontology - concepts related to problems to be solved, algorithms to solve them, objects that these algorithms operate on.

These ontologies were discussed in [9, 10, 11, 12, 13].

SLA

Another important aspect of the grid (and respectively the cloud) computing is the SLA (Service Level Agreement) defined as a contract between two parties specifying service level objectives, as expressions of requirements and assurance on the availability of resources and service qualities to provide Quality of Services (QoS) [14]. The SLA records a common understanding about services, priorities, responsibilities and guarantees. Each area of service scope should have the “level of service” defined, *e.g.*, the levels of availability, serviceability, performance, operation, or other attributes of the service, such as billing. Negotiations constitute one of the crucial phases of establishing the SLA between a client and a provider. In the AiG, SLA is represented semantically and semantic analysis is used to assess SLA offers received from the resource teams. The SLA negotiation is based on the FIPA Contract-Net Protocol [15], which is a specification dealing with pre-agreed message exchange using ACL messages (software agents communication standard).

GRID VS CLOUD

The aforementioned, considerations for the grid environments, can be extended to the clouds, since it can be stated that computational clouds evolved from computational grids [16]. In [16] Figure 1 the relations between distributed technologies are presented. Grids stem from supercomputing and clusters and form the basis for clouds and Web 2.0. Note that, grids and clouds are overlapping but one technology does not replace another, there are still deployments, in which application oriented approach (contrary to services oriented approach) can be used, *e.g.*, scientific computing. Comparision between grid and cloud is also addressed in [17, 16, 18]. The distinction between application and service oriented approaches will be discussed in the following section.

Grid computing is focused on heterogeneous, geographically distributed, multi-domain computer resources. Grid solutions systems that deliver sets of protocols and services, as well as grid middleware management software include: Open Grid Service Architecture by Open Grid Forum [19], Globus Toolkit [20], Unicore [21], GridWay [22], OpenLava [23]. Examples of real world computational grids include: general purpose grids (Teragrid, Open Science Grid, PL-Grid), domain specific grids (Cern Data Grid, caBIG, Earth System Grid) and voluntary computing projects (SETI@home, BOINC, Folding@home, GIMPS).

Cloud computing refers to an architecture in which groups of remote servers are networked to allow online access to computer services or resources. Cloud computing is based on research areas such as: thin clients, grid computing (infrastructure support), utility computing, cluster computing. Examples of cloud services include Amazon Elastic Compute Cloud (EC2) and Data Cloud (S3), Google App Engine, Microsoft Azure Service Platform, Amazon Web Services (AWS).

The general vision is the same as in the case of grids, *i.e.*, to reduce the cost of computing, increase flexibility and reliability. It can be observed that grids are designed with a bottom-up approach, *i.e.*, heterogenous resources are combined together and offered to end users (preconfigured resources are ready to accept jobs). On the other hand, clouds follow a top-down design approach in order to serve specific use cases (functionalities). Specifically, in grid systems clients look for resources that match their needs, in cloud systems they look for services. Interestingly, clouds may be composed from grids with services and interfaces superimposed on them. In this way, clouds offer a limited set of exposed features (compared to lower layers of the architecture). Grids tend to expose maximal semantics to the end user, via mostly programmatic interfaces, which leads to technical complexity. Contrary, clouds tend to expose a minimal amount of internal characteristics and semantics in order to increase simplicity and usability. The overall goal of a grid is collaborative sharing of resources,the goal of cloud is using the services at the same time while hiding their details.

The delivery model for grids is based on resources such as CPU, network, memory, bandwidth, device, storage, *etc.* Components aggregate different elements (OS, CPU, software, *etc.*) and are offered to end users. Naturally, in this case there are multiple owners and decentralized user management. In clouds, the delivery models are more abstract (everything as a service): IaaS (Infrastructure-as-a-Service; offering computing infrastructures, *i.e.*, virtual machines and other resources), PaaS (Platform-as-a-Service; offering computing platforms including, *e.g.*, operating system, programming-language execution environment, database, and web server), SaaS (Software-as-a-Service; providers install and operate software in the cloud and end users access the software from cloud clients). Services are bound to components that aggregate resources providing funtionalities. As a result, higher level of transparency and abstraction with possible real-time response time can be achieved. In this case, usually, there is a single ownership and centralized user management.

In the AiG, and generally, in the grid, two user roles were identified: resource providers and resource consumers (clients). In the cloud computing model, we can distinguish three roles: infrastructure provider (leases resources on demand), service provider (rents resources to one or more infrastructure providers) and service client (final consumer). Both grid and cloud computing require SLA based service negotiation, distributed resource management and scheduling.

Common goals and issues

The field of grid and cloud computing is lead by common tenants such as reducing cost of computation, and at the same time increasing flexibility and reliability of the infrastructure. Authors of [16] underline that, despite some differences, problems for grid and cloud communities are mostly the same, *i.e.*, to define methods by which consumers discover, request and use resources. In the cloud computing paradigm, the layer of services was added to hide the infrastructure details. In the grid, an intelligent middleware such as the AiG system is proposed to make the interactions with the grid infrastructure easier. Additionally, both communities define methods for the SLA management, based on service negotiation, distributed resource management and scheduling. Interestingly enough, the grid is closest to the Infrastructure-as-a-Service delivery model, in which a cloud provider offers servers, storage, network and operating systems and other infrastructure components. Leading IaaS providers include Amazon Web Services (AWS), Windows Azure, Google Compute Engine.

Finally, the common characteristic of both environments is an architecture based on components grouping limited availability resources (physical or virtual) with types: computing, storage, network, *etc.* Both communities deal with end users who have problem to be computed / data to be stored / data to be analyzed /, *etc.* – in a remote environment. Since there are multiple available resources / services, the problem arises for the client, which one should be used. Selection criteria may include: who is the owner, what is the price, when they are available. User may know a lot or may know little about the resources / services that are "the best for her/him." How can the system assure that proper / best resources / services will be selected? Note that, besides mentioned issues, both communities deal with the problem of efficient management of the infrastructure.

AGENTS IN CLOUD

The combination of software agents and semantic technologies, as an intelligent grid infrastructure provider, was selected for the AiG system. Note that, both grid and agent communities address open distributed systems (concepts and mechanisms) but from different perspective. The concept of combining software agents and grid was discussed in [24]: *current grid systems are somewhat rigid and inflexible in terms of their interoperation and their interactions, while agent systems are typically not engineered as serious distributed systems that need to scale, that are robust, and that are secure*. While the grid community is focused on the interoperable infrastructure, tools, applications for reliable and secure resource management (the "brawn"), the agent community is focused on flexible autonomous problem solvers for dynamic and uncertain environment (the "brain").

In [25] the author continues the discussion from [24] stating that *the convergence of interests between multi-agent systems that need reliable distributed infrastructures and Cloud computing systems that need intelligent software* *with dynamic, flexible, and autonomous behavior can result in new systems and applications*. He distinguishes two cases for utilizing agents: (i) cloud as an execution environment for complex modeling and simulation MAS systems, (ii) components for implementing intelligence in cloud computing systems, *e.g.*, in resource management and service provisioning. Despite the potential of an agent-based cloud computing to produce innovative techniques and applications, only a few research activities deal with this topic. In [26], authors present a concept in which software agents as cloud computing services represent clients in virtual environments. The Agency Service model facilitates providers with knowledge and technological infrastructure to provide agents as a service that users can contract to participate in virtual societies. In [27], authors present an agent-based approach to manage cloud computing infrastructure, in which agent are used in service discovery and cloud resources negotiation. More on cloud and agents integration can be found in [28] (service-oriented QOS-assured cloud computing architecture), [29] (integration of cloud on grid architecture with mobile agent platform to offer virtual clusters that are dynamically configurable with mobile agent based services), [30] (enhancing cloud infrastructure by reconfiguring allocated resources at the runtime).

Aforementioned solutions present interesting approaches and justify the rightness of applying software agents in cloud environments. We claim that the AiG system, and issues addressed within it, can provide basis for further elaborations on software agents and semantic data processing in cloud systems. We consider Agents in Cloud (AiC) to act as an agent-semantic intelligent middleware for efficient cloud service management. In the following sections we attempt to map different aspects of the AiG to clouds.

Decision support

In the AiG, we have considered a software selection problem, as the basis for the user decision support module. The goal is to help user in selection of the best method / tool / resource / contract to solve her problem, *i.e.*, to commission execution of respective job to the grid system. In the case of a cloud, this type of problem is considered a service selection problem. Note that, multiple cloud services offered by different providers may have similar functional properties. However, service descriptions are usually available on vendor's website in a non-standardized format, which makes autonomous machine-processing impossible. For the end user, the task of of analysis and selection of offers is difficult and time consuming. Moreover, selection of optimal service usually requires multi-criteria analysis (service should not only implement a certain functionality, but also quality levels should be considered). The difficulty in service and provider selection leads to the problem of vendor lock-in and problem of migration between service providers (in such a case, another comparison of offers should be performed).

In [31], trustworthiness-aware service selection with missing values prediction is discussed (based on criteria such as scalability, reliability, availability, *etc.*). In [32], authors propose a service selection mechanism based on evaluation of quality-of-service (QoS) parameters by clients based on their experience. The ratings are collected and later used in collaborative filtering to recommend services to the client. These two steps constitute the Service Recommendation protocol. The solution is based on the SOA infrastructure, where services are registered in a service broker, by which clients search and discover services.

The aforementioned service selection approaches offer user support, but it is limited to the quality and trustworthiness requirements. They do not consider common understanding of service description and analysis of functional properties with respects to the needs of the client. In [33], authors propose a rule-based decision support system that will help a client to decide whether or not to use a given cloud service provider. Additionally, they provide mechanisms to compare different providers' offers, however based on questionnaires that need to be filled by the providers.

An interesting approach for solving problem of efficient finding of a cloud service over the Internet was presented in [34]. Here, an attempt to build an agent-based discovery system that utilizes ontology when retrieving information about cloud services is presented. So far, a standard way to access a service has been through the websites of the cloud provider. Nowadays, there is a growing need to have a discovery mechanism that would search different kinds of clouds. The proposed solution uses existing search engines to gather information about websites of cloud services (eventually to generate alternative queries based on the cloud ontology), and then refines the results based on similarity reasoning in the cloud ontology. Noteworthy, the proposed solution utilizes semantics for service selection, even though there is no explicit user support.

It can be observed that the AiG user support mechanisms, based on semantic analysis, can be easily adapted to the analysis of cloud providers and offered services characteristics. The only precondition that should be satisfied is the semantic description of mentioned elements. With ontologies describing cloud(s) characteristics and depending on user level of expertise, the optimal service can be proposed by the system. The methods that were used in the case of the grid can be easily adopted to another ontology, *e.g.*, the cloud ontology. The expert knowledge representation is independent from the infrastructure on which it should be applied. In the AiG, the combination of the problem,

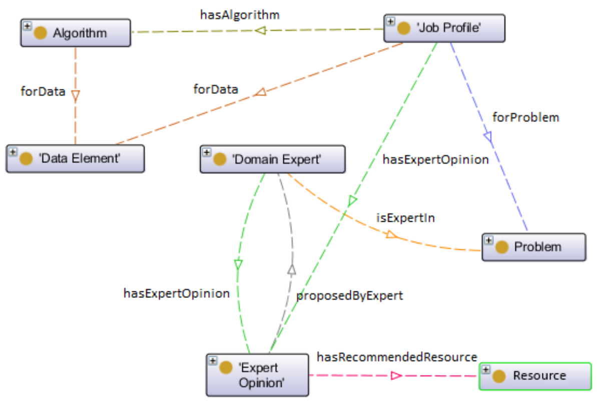

FIGURE 1: Main concepts in Expert Ontology

with input data properties, method and recommended resource, represents an expert opinion. In the case of the grid, a recommended resource was an instance of the *GridEntity* class, in the case of the cloud it can be the *CloudEntity* representing a service. Both classes are subclasses of *Resource* class. Since the ontologies are modularized such changes do not affect the overall design.

Ontologies

As mentioned in section on grid ontologies, for the AiG several ontologies were designed and implemented to model different aspects of knowledge in the system. Table 1 presents how ontologies from AiG can be mapped onto ontologies utilized in the cloud environment. The design of a generic and extensible AiG Expert Ontology was discussed in [10]. The selected approach to construct this ontology enables easy adaptation to the cloud environment.

TABLE 1: Mapping of respective grid and cloud ontologies

grid	Cloud	Content
AiG Grid Ontology	Cloud Ontology	Concepts for description of cloud infrastructure; Extension of AiG Grid Ontology with services;
AiG Conditions Ontology	Cloud Conditions Ontology	Concepts related to SLA representation and negotiation; Extension of grid SLA with cloud-specific concepts;
AiG Messages Ontology	Cloud Messages Ontology	Concepts represent content of messages exchanged by agents that can be used in the agent-based system for managing clouds;
Expert Ontology	Expert Ontology	Concepts related to problems and their properties, methods to solve them, recommended resource can be a grid entity or a cloud service depending on the context system;

Note also that the semantic descriptions of clouds will enable easier management of federations of clouds. Cloud federation requires one provider to make available (sell / rent) computing resources to another cloud provider. Those resources can temporarily, or permanently, extend the buying provider’s cloud computing environment, depending on the specific federation agreement between providers. Semantic analysis provides mechanisms to enable “common language” for negotiating such collaborations and exposing service descriptions as offers to the both service clients and other service providers.

SLA

Naturally, the SLA management is part of both grid and cloud environments, in which services can be treated as equivalent to resources. In the case of the grid, the common problem with the SLA representation was lack of standard, common and shared terminology and SLA document structure. We addressed this with AiG Conditions Ontology, which captured concepts related to the semantic representation of SLA terms.

The SLA negotiation in cloud computing are still under investigation [35]. There are works conducted on the ISO standards (ISO/IEC FDIS 19086) for cloud computing SLA framework and technology, but their status is still *under development*. Author of [36] compares SLAs of most popular cloud service providers on the market, pointing out common aspects and differences that appear in IaaS SLAs for compute and data services. He lists aspects that are currently considered and issues that should be considered in the future, *e.g.*, standardization. Author notices that lack of structured and semantically consistent SLA representations leads to difficulties in comparison of different provider's services. In [37, 38], an analysis and assessment of publicly available agreements for leading cloud service providers is given. Notable observations are: SLAs do not follow common industry-wide terminology, service level objectives should be easily comparable between SLAs. Authors consider the SLA as a legal contract and do not focus on its machine-processable electronic representation, however, given observations are common for any document representation form.

From the presented analysis it can be concluded that the problem of lack of standardization and commonly acceptable semantics that was identified for the grids, is also actual in the case of cloud computing. The difficulty of SLAs comparison and assessment that was identified for the grids is also observable for the clouds.

An integral part of SLA management is an agreement negotiation (performed by entities acting on behalf client and providers). In the case of the grid and the cloud they can be considered at two levels:

1. - Grid: The multilateral negotiation between clients and grid resource brokers (resource team leaders in the AiG) for establishing contracts to provide resources to clients.
   - Cloud: The multilateral negotiation between clients and third-party cloud service providers for establishing contracts to provide services to the clients.
2. - Grid: Concurrent one-to-many negotiation between grid resource brokers and resource (infrastructure) providers to establish contracts with multiple providers to obtain a collection of resources exposing required functionalities/features for sub-leasing to the clients. In the AiG, this step is done in the team joining scenario, where agent representing resource wants to join a team of resources in the system. Teams that offer functionalities are formed asynchronously to job executions.
   - Cloud: Concurrent one-to-many negotiation between cloud service providers and resource providers to establish contracts with multiple providers to obtain a collection of resources to compose a service for sub-leasing to clients.

In the AiG, the implementation of concurrent negotiation mechanisms was based on distributed agent infrastructure and exchange of messages with semantic (ontologies) content. There has been a lot of work devoted to multi-agent systems usage for SLA negotiations [39, 40, 41]. This approach can be also utilized for negotiations in the cloud systems.

CONCLUDING REMARKS

Many similarities have been identified between grid and cloud computing, which lets us observe that research done to enable efficient utilization of the grid infrastructures can be extended and adopted to the cloud infrastructures. Following the AiG system, we tried to show that, as it was stated in the case of grids, integrating software agents, semantics and cloud computing can enable novel intelligent systems, making clouds even more flexible and autonomic. One of the concerns for the grids was the amount of technical details that was exposed to client, which made the grid utilization complex and required expert knowledge. On the other hand, clouds expose minimum amount of technical details to enable user-friendly use of services. Application of agent and semantic technologies may allow to efficiently control clouds with access to more details and provide grids with a "more user-friendly" interface. Moreover, we recognize semantic technologies as an enabler to bring together heterogeneous world of cloud providers by allowing easier integration and comparison of offers.

ACKNOWLEDGEMENTS

Work presented here is a part of the Poland-Bulgaria collaborative grant "Parallel and distributed computing practices." It is also a part of Poland-Romania collaboration "Software agents and agent systems." Finally, it is also a part of the Serbia-Romania-Poland collaboration "Agent systems and applications."

REFERENCES

[1] M. Dominiak, W. Kuranowski, M. Gawinecki, M. Ganzha, and M. Paprzycki, "Utilizing agent teams in Grid resource management—preliminary considerations," in *Proc. of the IEEE John Vincent Atanasoff Conference*, (IEEE CS Press, Los Alamitos, CA, 2006), pp.46–51.

[2] W. Kuranowski, M. Ganzha, M. Gawinecki, M. Paprzycki, I. Lirkov, and S. Margenov (2008) *International Journal of Computational Intelligence Research* **4**, 9–16.

[3] K. Wasielewska, M. Drozdowicz, M. Ganzha, M. Paprzycki, N. Attaui, D. Petcu, C. Badica, R. Olejnik, and I. Lirkov, "Negotiations in an Agent-based Grid Resource Brokering Systems," Chapter in *Saxe-Coburg Publications*, (Stirlingshire, UK, 2011).

[4] K. Wasielewska, M. Ganzha, M. Paprzycki, C. Badica, M. Ivanovic, and I. Lirkov, "Semantic technologies in a decision support system," in *AMiTaNS'15*, AIP Conference Proceedings, Vol.1684, edited by M.D.Todorov (American Institute of Physics, Melville, NY, 2015), paper 060002.

[5] K. Wasielewska, M. Ganzha, M. Paprzycki, P. Szmeja, M. Drozdowicz, I. Lirkov, and C. Badica, "Applying Saaty's multicriterial decision making approach in grid resource management," in *CEUR Workshop Proceedings*, Vol.920, edited by Z. Budimac *et al.*, (2012), p.134.

[6] K. Wasielewska and M. Ganzha, "Using analytic hierarchy process approach in ontological multicriterial decision making – Preliminary considerations," in *AMiTaNS'12*, AIP Conference Proceedings, Vol.1487, edited by M.D. Todorov (American Institute of Physics, Melville, NY, 2012), pp.95–103.

[7] S. Staab and R. Studer (Eds), *Handbook on Ontologies*, 2nd edn, (Springer, 2009).

[8] W. Xing, M.D. Dikaiakos, R. Sakellariou, S. Orlando, and D. Laforenza, "Design and development of a core grid ontology," in *Proc. of the CoreGRID Workshop: Integrated research in Grid Computing* (2005), pp.21–31.

[9] M. Drozdowicz, M. Ganzha, K. Wasielewska, M. Paprzycki, and P. Szmeja, in *Agreement Technologies*, Law, Governance and Technology Series, Vol. 8, edited by S. Ossowski (Springer, Netherlands, 2013), pp.149–168.

[10] P. Szmeja, K. Wasielewska, M. Ganzha, M. Drozdowicz, M. Paprzycki, S. Fidanova, and I. Lirkov, "Reengineering and extending the Agents in Grid Ontology," in *Large Scale Scientific Computing*, LNCS (Springer, Berlin Heidelberg, Germany, 2013), pp. 565–573.

[11] M. Drozdowicz, K. Wasielewska, M. Ganzha, M. Paprzycki, N. Attaui, I. Lirkov, R. Olejnik, D. Petcu, and C. Badica, "Ontology for contract negotiations in agent-based grid resource management system," in *Trends in Parallel, Distributed, Grid and Cloud Computing for Engineering*, (Saxe-Coburg Publications, Stirlingshire, UK, 2011).

[12] M. Drozdowicz, M. Ganzha, K. Wasielewska, M. Paprzycki, I. Lirkov, R. Olejnik, and N. Attaoui, "Utilization of modified coregrid ontology in an agent-based grid resource management system," in *Proceedings of the CATA 2010 Conference 25th International Conference on Computers and Their Applications* (2010), pp.240–246.

[13] W. Kuranowski, M. Paprzycki, M. Ganzha, M. Gawinecki, I. Lirkov, and S. Margenov, "Agents as resource brokers in grids – forming agent teams," in *Proceedings of the LSSC Meeting*, LNCS, Vol. 4818 (Springer, Germany, 2007), pp.472–480.

[14] R. Buyya and K. Bubendorfer, *Market-Oriented Grid and Utility Computing* (Wiley Publishing, 2009).

[15] "Fipa Iterated Contract Net Interaction Protocol Specification," `www.fipa.org/specs/fipa00030/PC00030D.pdf`.

[16] I. Foster, Y. Zhao, I. Raicu, and S. Lu, "Cloud computing and grid computing 360-degree compared," in *Grid Computing Environments Workshop, 2008*, (IEEE, 2008), pp.1–10.

[17] S. M. Hashemi and A. K. Bardsiri (2012) *ARPN Journal of Systems and Software* **2**, 188–194.

[18] S. Jha, A. Merzky, and G. Fox (2009) *Concurr. Comput.: Pract. Exper.* **21**, 1087–1108.

[19] "Open grid forum," `https://www.ogf.org/ogf/doku.php`.

[20] I. T. Foster, "Globus toolkit version 4: Software for service-oriented systems," in *NPC*, LNCS, Vol. 3779, edited by H. Jin *et al.* (Springer, 2005), pp.2–13.
[21] D. W. Erwin and D. F. Snelling, "Unicore: A grid computing environment," in *Euro-Par 2001 Parallel Processing*, LNCS, Vol. 2150, edited by R. Sakellariou *et al.*, (Springer-Verlag, 2001), pp.825–834.
[22] "Gridway metascheduler," `http://www.gridway.org/doku.php`.
[23] "Openlava open source workload management," `http://www.openlava.org/`.
[24] I. Foster, N. R. Jennings, and C. Kesselman (2004) *Autonomous Agents and Multiagent Systems, International Joint Conference on* **1**, 8–15.
[25] D. Talia (2012) *IEEE Internet Computing* **16**, 78–81.
[26] I. Lopez-Rodriguez and M. Hernandez-Tejera, "Software agents as cloud computing services," chapter in *Advances on practical applications of agents and multiagent systems*, 9th International Conference on Practical Applications of Agents and Multiagent Systems, (Springer, Berlin Heidelberg, 2011), pp. 271–276.
[27] K. M. Sim, "Towards complex negotiation for cloud economy," in *Advances in Grid and Pervasive Computing,* Proc. 5th International Conference, GPC 2010, Hualien, Taiwan, May 10-13, 2010, edited by P. Bellavista *et al.*, (Springer, Berlin Heidelberg, 2010) pp. 395–406.
[28] B.-Q. Cao, B. Li, and Q.-M. Xia, "Cloud computing," in *A Service-Oriented Qos-Assured and Multi-Agent Cloud Computing Architecture*, Proc. First Intern. Conf., CLOUDCOM'2009, Beijing, China, December 1-4, 2009, edited by M. G. Jaatun *et al.*, (Springer, Berlin Heidelberg, 2009), pp. 644–649.
[29] R. Aversa, B. D. Martino, M. Rak, and S. Venticinque, "Cloud agency: A mobile agent based cloud system," in *Complex, Intelligent and Software Intensive Systems (CISIS)*, Proc. of International Conference, (2010), pp. 132–137.
[30] A. Tolba and A. Ghoneim (2015) *JSW* **10**, 1037–1044.
[31] S. Ding, C.-Y. Xia, K.-L. Zhou, S.-L. Yang, and J. S. Shang (2014) *PLoS ONE* **9**, 1–11.
[32] K. Tserpes, F. Aisopos, D. Kyriazis, and T. Varvarigou (2012) *Future Gener. Comput. Syst.* **28**, 1285–1294.
[33] T. Sander and S. Pearson (2010) *GSTF Journal on Computing (JoC)* **1**(1).
[34] T. Han and K. M. Sim "Intelligent control and computer engineering," in *An Agent-Based Cloud Service Discovery System that Consults a Cloud Ontology*, edited by S.-I. Ao *et al.*, (Springer, Dordrecht, Netherlands, 2011), pp. 203–216.
[35] Cloud Computing Service Level Agreements, Exploitation of Research Results, Techn. Rep. European Commission Directorate General Communications Networks, Content and Technology, Unit E2 – Software and Services, Cloud, 2013.
[36] S. A. Baset (2012) *SIGOPS Oper. Syst. Rev.* **46**, 57–66.
[37] "Cloud Service Level Agreement Standarisation Guidelines," `https://ec.europa.eu/digital-single-market/en/news/cloud-service-level-agreement-standardisation-guidelines`.
[38] "Cloud Standards Customer Council: Public Cloud Service Agreements: What to Expect and What to Negotiate," `http://www.cloud-council.org/deliverables/CSCC-Public-Cloud-Service-Agreements-What-to-Expect-and-What-to-Negotiate.pdf`.
[39] D. Ouelhadj, J. Garibaldi, J. MacLaren, R. Sakellariou, and K. Krishnakumar, "A multi-agent infrastructure and a service level agreement negotiation protocol for robust scheduling in grid computing," in *Advances in Grid Computing – EGC 2005*, European Grid Conference, Amsterdam, The Netherlands, February 14–16, 2005, Revised Selected Papers, edited by P. M. A. Sloot *et al.*, (Springer, Berlin Heidelberg, 2005), pp. 651–660.
[40] Q. He, J. Yan, R. Kowalczyk, H. Jin, and Y. Yang, "An agent-based framework for service level agreement management," in *11th International Conference on Computer Supported Cooperative Work in Design* (2007), pp. 412–417.
[41] N. Giri, S. Mundle, A. Ray, and S. Bodhe, "Multi agent system based service architectures for service level agreement in cellular networks," in *Proceedings of the 2nd Bangalore Annual Compute Conference*, COMPUTE (ACM, 2009), pp. 5:1–5:6.