Journal of Web Semantics, Volume 4, Issue 2, June 2006.

On Agents and Grids: Creating the Fabric for a New Generation of Distributed Intelligent Systems

Yolanda Gil
Information Sciences Institute
University of Southern California
4676 Admiralty Way
Marina del Rey, CA 90292
gil@isi.edu

March 24, 2006

Abstract

The semantic grid is the result of semantic web and grid researchers building bridges in recognition of the shared vision and research agenda of both fields. This paper builds on prior experiences with both agents and grids to illustrate the benefits of bringing agents into the mix. Because semantic grids represent and reason about knowledge declaratively, additional capabilities typical of agents are then possible including learning, planning, self-repair, memory organization, meta-reasoning, and task-level coordination. These capabilities would turn semantic grids into *cognitive grids*. Only a convergence of these technologies will provide the ingredients to create the fabric for a new generation of distributed intelligent systems.

1 Introduction

Semantic grid researchers (www.semanticgrid.org) have noted the common themes in the semantic web and grid research agendas, as represented in technical publications in both areas [Berners-Lee et al 01; Euzenat 01; Foster and Kesselman 04; Berman et al 03]. These commonalities have been articulated in [De Roure et al 01; Goble and De Roure 04; De Roure et al 04; De Roure et al 05]. Not surprisingly, in recent years both communities share an interest in the widespread service-based architectures. Semantic web researchers are developing extensions to web services to include semantic representations (http://www.w3.org/2002/ws/swsig). Grid services and the more recent WS framework (http://www.globus.org/wsrf) align web services with more traditional features of grids. With this emphasis in service-based paradigms, these fields are now sharing more than ever a research agenda with yet another community: autonomous agents and multi-agent systems. Agent systems have historically been developed in a modular fashion, with a clear advertisement to other agents of their capabilities, which is an important emphasis of service-based architectures. The research agenda in the agents community [Luck et al 05; Jennings et al 98] shares much with the semantic web and grid research agendas. Some commonalities have been pointed out in [De Roure 05; Luck et al 05; Foster et al 04], including trust, negotiation, resource allocation, composition, and autonomy.

This paper argues that neither grid nor agent systems alone can deliver on their promise without building on one another. The arguments put forward are based on first-hand experiences with agent systems [Chalupsky et al 02; Gil and Ratnakar 03], scientific workflow creation and execution in grids [Deelman et al 03a; Deelman et al 03b; Deelman et al 03c; Blythe et al 03a; Blythe et al 03b; Gil et al 04; Deelman et al 04; Kim et al 04; Maechling et al 05; Gil 06], and metadata catalogs for grids [Tuchinda et al 04; Gil et al 05; Gil et al 06]. In our work, we have used grid techniques to address shortcomings in agent systems, and AI techniques to bring not only semantics but also intelligent reasoning capabilities to grids.

The paper is organized in two main sections, one focused on why agent systems need grids, and the second on why grids need agent systems. The first section discusses capabilities that grid technologies provide that would address important limitations of agent systems, concentrating on providing robust and sustainable performance. These facilities include state, persistence, and lifecycle management. The second part of the paper discusses important capabilities developed and used in the agents community at large that would address current limitations of grid computing. These capabilities include learning, planning, interaction, and coordination. We argue that these capabilities would take grids to a new level of scale and sophistication, able to make complex informed decisions and flexibly adapt their performance to new information and unexpected situations.

2 Why Grid: Limitations of Current Multi-Agent Systems for Robust and Sustainable Performance

Robustness is a very important requirement for distributed problem solving of the kind performed by multi-agent systems. Robustness is challenging when heterogeneous collections of components need to be coordinated in a highly dynamic non-centralized execution environment. These are the environments that grids were designed to address. Reliability, quality of service, and robustness have been central themes in grid research as essential architectural principles. Research in multi-agent systems has focused more on distributed operations at the task level, concentrating on architectures, agent communication languages, and coordination [Finin et al 94; Cheyer and Martin 01; FIPA 02]. We argue that grid environments provide an ideal substrate for developing multi-agent architectures and distributed problem solving capabilities. This section describes key features provided by grid services [Foster et al 02; Tuecke et al 03] and their relevance to support robust implementations of agent-based systems.

Our experience with multi-agent systems, and in particular with the Electric Elves project [Chalupsky et al 02], can be used to motivate these issues. Electric Elves supported users with office tasks and integrated a number of agents that were heterogeneous in their function and in their implementation, and were deployed in a distributed environment. Some of the agents accessed web information sources, others matched requests with agent capabilities, and others communicated with users about task status. As we strived to use the system 24/7 inside and outside of our organization, we discovered that a non-trivial amount of effort was required to keep the groups of agents functioning appropriately. The capabilities of each agent and the kids of information they exchanged were clearly specified. Yet, even such an agent community with modest distributed and heterogeneous nature posed clear challenges. As the agents operate, it was necessary to monitor their status and the status of specific requests. This was done manually, but ideally it should handled automatically. The kinds of status questions that came up included: 1) is the other agent still working on my request? 2) does the agent still need my reply? 3) has the agent already responded but the response was not received? 4) what is the status and progress of my request? Other problems came up when machines were down or connections failed. Should the requesting agents have a memory of their requests and resend them, or should the servicing agent keep track of their requests and reinstate them upon restart? Who should notice that an agent is not running and who should restart it? Although it was possible to manage the multi-agent system manually, this is not a desirable option and clearly impossible if multi-agent systems are to scale up in number and heterogeneity.

These challenges are not uncommon and it is reasonable to consider they hinder growth and scale of deployed agent communities. In a published study, [Riemenschneider et al 04] noted problems in agent infrastructures with

significant development effort in dependability and survivability at larger scale (hundreds of distributed agents). We believe that these are symptoms of the need for better distributed infrastructure to build multi-agent systems.

These issues can be summarized as follows:

1. Introspection: Once a request is sent to an agent, it would be useful to be able to query the agent regarding the receipt of the request, the status of the request, and perhaps any intermediate results that the agent may have generated. It would be useful to setup automatic notifications of changes in a request status.
2. Tracking multiple requests: For each agent, we have to implement a mechanism for tracking multiple requests. For each request, the agent has to maintain its status, current result, etc. and detect the same request sent repeatedly when the message acknowledging its receipt delayed.
3. Persistence: Each agent can implement a database to store requests so they can be recovered in case the agent process dies. Special purpose scripts may be needed to check the agent process and restart it if it is not running.
4. Shelf life of requests: Each agent needs to implement a mechanism to deal with expiration of requests. Otherwise, all requests would be active forever. Other services that may not have a clear date associated with them may be harder to handle in terms of terminating the request.
5. Evolution of the specifications: The agent specification (in many cases due to changes in the underlying web site) may change over time. This requires manually tracking changes in other agent specifications, so that request message formats could be updated.

The implementation of individual agents can be extended to support these questions in an ad-hoc manner. However, these problems clearly raise the need for better infrastructure to support robust continuous operation as tasks are executed in a dynamic distributed environment. Instead of developing ad-hoc solutions for these problems, it is useful to investigate whether we can build on existing approaches in distributed computing that are designed to address these very issues.

2.1 Some Useful Concepts in Grids

We conducted a study of how grid services would support a more robust agent infrastructure, focusing on how the requirements discussed above would be addressed in grids. We implemented a set of agents using grid service definitions and found that the built-in mechanisms in grid services provided

facilities to support much of the functionality we needed regarding the state of the agents and the requests. A detailed account of the implementation is provided in [Gil and Ratnakar 03]. This section describes some useful concepts in grids illustrated in the context of our multi-agent system. Our implementation was done with the Open Grid Services Infrastructure (OGSI) [Tuecke et al 03] and its GT3 specification. Here we use terminology from that work. Its new generation is the WS-Resource Framework (WSRF) (http://www.globus.org/wsrf) and has slightly different terminology but the same underlying core concepts. See [Foster et al 05] for a detailed mapping of terminologies.

Grid Services are extensions of web services that incorporate several key features that are crucial for robust distributed operations. One key feature of grid services is the notion of lifecycle management: a request has a lifetime that is managed by the system by assigning it a process. A request sent to a grid service may create a transient process called *grid service instance* with a unique identifier, a *handle*, that can be used to locate and query the instance. The service instance exists only for a limited amount of time, after which the instance will be destroyed. If required, the client/application can also extend the expiration time of the service instance. Service instances can be created across pools of hosts with appropriate load balancing.

Another key feature of grid services is that they have internal state and support introspection. Grid services are stateful, which means that they maintain information across multiple operations issued over time. Therefore, there is a standard mechanism to expose the data associated with each service instance for query, update and change notification. This also supports third party notification of request status. Each grid service instance can have *persistent properties* associated with it that are saved on permanent storage, and can be restored by the system on a restart of the grid service instance in case the process is accidentally terminated (e.g., if the host goes down).

Grid services also adopt useful conventions for version management. Grid services cannot use the same identifier if their interface specification is changed. This enables clients to have a handle on upgrades and changes to the specification through declarations of the shelf life of a service specification.

Note that some of the features of grid services described here are being considered for adoption by web service standards, such as service state declarations and access mechanisms.

2.2 Supporting Distributed Agent Communities with Grids

The challenges raised earlier can be addressed by grid services as follows:

1. **Introspection:** The grid service data provides the infrastructure needed

to track requests and to enable clients to setup notifications for updates to the status of the request.

2. **Tracking multiple requests:** The agent had an overhead of dealing with multiple requests. Here, we do not have to worry about multiple requests in the implementation of the service, since each request spawns off a new service instance with its own properties and service data.
3. **Persistence**: This is addressed by the "persistent properties" of grid services. Data is not lost in a machine crash because persistent storage is provided by the grid service.
4. **Shelf life of requests**: In grid services, the lifetime of the service instance can be explicitly set at the time of creation. Also, all service instances are destroyed after their expiration time, so there is an explicit way to handle obsolete requests. There is no standard mechanism to set the expiration of the request for an agent system.
5. **Evolution of the specifications**: Grid services have a unique identifier that remains the same as long as they have the same interface specifications. If those specifications change, the grid service will have a new identifier. The specifications may have an expiration date indicated as a guarantee of validity. This does not solve the problem, but at least the client/application has an explicit way to be informed about it.

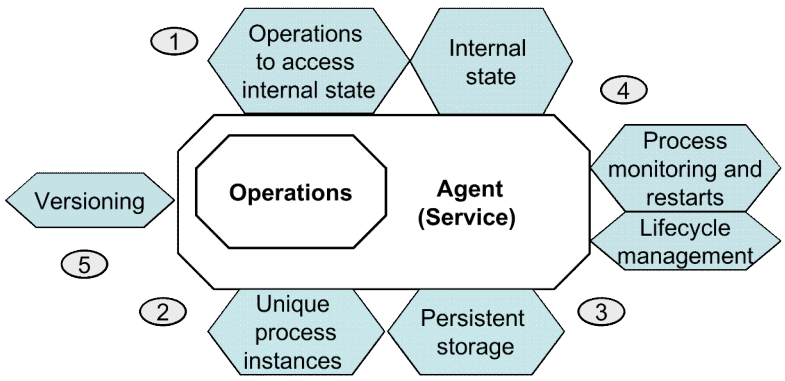

Figure 1. Making agents more robust through grid services: 1) Introspection is supported by making agents be stateful processes that have standard operations to publish and access information about their internal state; 2) Tracking requests through creation of unique process instances; 3) Persistence through permanent and replicated storage of requests and other process state; 4) Management of shelf life of requests through expiration dates and other lifecycle management techniques; 5) Responsible evolution and change management through versioning conventions about operation changes.

2.3 Summary: Why Grids

Figure 1 summarizes the kinds of built-in facilities from grid services discussed in this section that can be used to provide agents with more robust behaviors. Grid services offer an infrastructure for distributed computing that supports robust coordination in highly dynamic environments. Multi-agent and distributed intelligent systems could be made more robust by taking advantage of their mechanisms to address introspection, state, lifecycle management, and persistence. There are many other benefits of grids that are relevant to agent systems. Because grid computing has addressed distributed systems bottom up, it also provides a well-developed infrastructure for security, cross-organization access policies, load balancing, efficient data transfer, and execution management. These are all of interest to distributed intelligent systems.

3 Why Agent Architectures: Limitations of Current Grids in Knowledge and Complexity

One of the prominent applications of grid computing, though many others exist, is large-scale scientific computations. Several years of collaboration on workflow planning and execution in grids [Deelman et al 03a; Deelman et al 03b; Deelman et al 03c; Blythe et al 03a; Blythe et al 03b; Gil et al 04; Deelman et al 04] and workflow creation [Kim et al 04; Maechling et al 05] show the utility of AI techniques to assist users to create scientific workflows and to automatically complete and map workflow tasks to resources for execution. We have also investigated the benefits of using mediators for information agents to support semantic integration of metadata catalogs for grids [Tuchinda et al 04; Gil et al 05]. The results of this work only scratch the surface of the wide range of AI techniques utilized in agent systems. This section suggests the role that more advanced agent architectures and techniques could play in specific aspects of grid computing.

Agents and grid applications face similar challenges: modeling and understanding their external environment and making informed decisions to achieve their tasks appropriately. Agents typically have sensors to collect information about their external environment and effectors to change it in order to accomplish their tasks. These environments are often dynamic, may be shared with other agents, and may include adversarial or interfering agents. Grid applications gather information about the grid environment through information services, and affect the environment through a variety of actions such as job submission and data management operations. Grid environments are very dynamic in nature, since any resource available can become unavailable, and many resources must be shared among applications.

A variety of important contributions from agent research could be incorporated in grids to provide significant new capabilities. Agent architectures are designed to exhibit autonomy, decentralized coordination, and complex distributed behaviors in highly dynamic environments. All these criteria are important to grids. This section discusses some important capabilities developed for agent systems that could be used in grids. Each topic is briefly described in terms of the issues addressed in agent systems and their potential relevance to grids.

3.1 Planning and Failure Models in Dynamic Environments

An important capability of many agents is to plan ahead in order to accomplish complex tasks. Planning ahead can help by assessing the feasibility of an option being considered and by analyzing the interactions among related aspects of a task. This enables an agent to prepare the environment to enable later tasks as well as to coordinate better with other agents. In addition, the agent can understand the impact of a failure when one occurs and the best way to fix it in the context of the overall plans.

The Remote Agent illustrates these concepts [Muscettola et al 98]. It was developed to schedule and execute science experiments autonomously as it flew in a spacecraft to Jupiter and beyond in the Deep Space One mission. Using automatic planning techniques, high-level mission goals were turned into specific actions to set the hardware of the spacecraft, to send data back to Earth, and to partially process the data on-board. Observations had to be arbitrated and scheduled for the available time frames in a setting where human controllers on ground could not possibly control the agent's operations. Remote Agent used planning horizons to determine how far ahead and with how much detail it specified its operations. It looked ahead as far as 72 hours, but only planned in detail the operations for the coming observation window.

Remote Agent also used models of possible failures in the hardware it was controlling, their relationships to various indicators and diagnostics actions, and used these models to detect and respond to failures. Failures were detected based on the failure conditions hypothesized and/or determined by the system. Failures were resolved by dynamically configuring other resources available in the spacecraft.

In grid environments, failures are commonplace due to the dynamic availability of resources and a variety of unexpected events. Models of failure modes and diagnostic conditions together with approaches that resolve failures would improve current capabilities to perform complex tasks. This would enable advanced resource reservations, arbitration among requests, and analysis of feasibility. Planning ahead of execution time, and relating individual tasks to overall common goals or optimization criteria would lead to better overall performance.

3.2 Planning and Control to Manage Uncertainty

In many environments, there is a disconnect between the situation that the agent planned for and the actual situation due to uncertainty. Uncertainty may be a result of the impossibility of controlling sensors and effectors with exact precision, the accumulation of errors over many steps, and the continuous dynamics in the environment that result in errors in the agent's models of the external world. Tiered architectures to combine high-level planning and low-level dynamic control have been powerful paradigms for agents to cope with uncertainty.

Path planning tasks illustrate the benefits of the combination of high-level planning and low-level control (eg, [Urmson et al 03]). Consider robot exploration tasks that require moving around to locations where certain observations can be made or samples collected. High-level science goals can be established and planned for. In a deliberate planning stage before execution. Given the main locations or landmarks, route optimization techniques can be used to generate an overall target plan. Intermediate waypoints can be generated to create a path description that will steer the lower-level controls. A tight control loop for execution steers the motors and exploits sensor information dynamically to reach the planned waypoints and landmarks.

Grid applications contain high-level goals to achieve complex overall tasks that must be executed in a highly uncertain execution environment. Higher-level planning and tasking can be combined with lower-level control systems that can dynamically steer more execution-related decisions.

3.3 Learning to Improve Performance

Agents would be hardly considered intelligent if they repeated the same mistakes and never improved their performance in tasks they perform routinely. Learning may be simply collecting performance metrics and deriving statistical predictive models, or may encompass more complex learning where efficient procedural knowledge is compiled after reasoning from first principles. Learning can also improve performance by recognizing common failure conditions and designing mechanisms to anticipate and avoid them.

A wide range of learning techniques is used in agent systems. These include reinforcement learning from reward feedback, learning Markov decision processes to improve overall policies, and symbolic compilation of behavior-triggering rules (e.g., [Kaelbling et al 95; Dietterich 00; Hengst 00; Newell 90; Anderson et al 04; Knoblock 94]).

In grid environments, performance and quality of service are paramount concerns. In such a highly dynamic and uncertain execution environment, learning and adaptation mechanisms would improve resource assignment and load balancing.

3.4 Learning to Handle Novel Situations

Dynamic environments and continuous operations lead to unusual and unanticipated situations, and autonomous agents must resort to learning techniques that expand their initial knowledge and skills.

Learning techniques to handle novel situations include exploration and experimentation to gather additional knowledge about the new situation, learning from first principles, learning from observation of other agents, learning cooperative behaviors in situations that require capabilities that the agent does not have, and selecting when and how to resort to learning from user instruction when all else fails [Lieberman 01; van Lent and Laird 99; Gil 93; Cypher 93; Dent et al., 1992].

Grid applications function in very complex environments. Unanticipated behaviors and situations require adaptive capabilities in order to function appropriately in novel environments. Learning to detect and resolve unpredictable situations will be an important capability for autonomous operations.

3.5 Problem Solving and Memory Structures

Complex behaviors that require a range of skills and scope often require organizing knowledge and problem solving abilities and bring to bear only the relevant factors for any given situation. By reasoning at various levels of abstraction, and by organizing memory structures with appropriate relevance indices, agents are able to scale up in the size and coverage of their behaviors.

Perhaps the most complex agents have been developed with the Soar architecture [Newell, 1990], which has been used to develop agents in simulated worlds used for training and tutoring. Complex and interdependent decision factors are organized in separate problem spaces, each with its own goal set, search strategies, and search control heuristics. Problem spaces are related through subgoals, where an unresolved decision in a problem space may trigger a separate problem space that brings to bear additional factors to consider in making that decision. Large amounts of knowledge are needed to describe goals, tasks, and problem solving methods. Additional knowledge is learned automatically by the system as it performs tasks. All this knowledge is structured and indexed in memory for efficient retrieval and performance. A long-term memory organizes problem-solving rules into the problem spaces that they are applicable for. A short-term memory is used to store knowledge relevant to the problem-space currently active. Powerful memory matching schemes bring to the working memory used for a particular problem space all the rules from long-term memory that are applicable to the decision at hand.

Interacting decisions may result in backtracking. This complex deliberation mechanism is complemented by learning techniques that improve performance by generalizing and compiling new rules. These learned rules are immediately applied when a similar decision is encountered in the future, avoiding the repetition of the deliberation process.

As grid applications scale up, there will be many complex and interrelated decisions and constraints regarding the resources and services available. Complex and well-informed decisions can only be made if the decision space can be appropriately represented and factored. The knowledge in the system must be structured so the relevant portions that need to be brought to bear are indexed and retrieved.

3.6 Metareasoning

Meta-reasoning enables an agent to make decisions about what to reason about, setting its own goals and deciding how to allocate its resources.

Retracting active behaviors in response to new information has been researched in beliefs-desires-intentions (BDI) architectures [Morley & Myers, 2004; Rao & Georgeff, 1998; Bratman et al., 1998]. A dynamic control system incorporates new beliefs by observing the current state, creates desires based on its pending tasks and goals, and based on them decides what intentions to trigger by selecting from possible venues of action. The deliberation often results in meta-level reasoning to assess whether to retract or continue pursuing currently active behaviors. Meta-reasoning is used in other contexts to decide whether to explore and acquire new capabilities by seeking novel experiences or to exploit current abilities to maximize performance. Meta-reasoning is also appropriate to decide how to partition a complex tasks into subtasks carried out by distributed agents, and on the subsequent coordination and communication as well as failure repairs.

Grid environments will include many components that could optimize various aspects of performance, pursue opportunistic courses of action, and reconsider goals selected or postponed. The ability to make meta-level decisions will likely impact overall performance and even the resilience of the system to accomplish complex tasks in the light of drastic changes in the environment.

3.7 Intervention and Interaction

Many agent systems require human interaction, either because they are assisting humans and need to report progress or results, or because they perform collaborative tasks that involve human-agent cooperation. Such interaction skills require the ability to explain the reasons for a response or for a question, the ability to incorporate information provided by a human, and modeling the

human user in terms of their skill level as well as their assumptions based on prior dialog conducted with the agent.

Conversational agents and dialog systems offer architectures to support the communication between a user and a complex system [Allen et al 01; Sidner et al 03; Grosz and Kraus 99]. Tasks that require user-system collaboration can be successfully accomplished through shared task structures that include explanations provided by the system about its behavior, as well as models of user's beliefs and the skills they contribute to the task at hand.

Grid environments are envisioned to serve a wide variety of uses and users. Some grid environments are accessed daily by computer researchers and professionals, system developers, high-school students, and expert domain scientists. Usability remains a challenge for grid environments. User steering of grid applications, needed in some domains, require users to understand the status of intermediate results of applications so additional or alternative tasks can be inserted. Helping users understand the workings of such complex systems as they are relevant to their particular tasks and goals would be facilitated by user and dialog modeling techniques as well as explanation-centered system design.

3.8 Coordination, Shared Goals, and Adversarial Models

Complex tasks are often only possible by drawing from groups of agents with complementary skills or resources. Coordinating the delegation and accomplishment of subtasks in an multi-agent system has let to important research results in defining and maintaining joint shared goals, teamwork behavior, and hierarchical subtask organizations. Other agents often get in the way of accomplishing tasks, and in some cases may have adversarial or competing goals.

The Robocup agents exemplify this research area, as teams of robots or software players need to coordinate their behavior by playing different roles in the team in the presence of an opposing team (http://www.robocup.org). What to communicate and when, how to detect whether a team goal is threatened and can be supported by changing an agent or the overall team behavior, dynamically changing goals and even roles within the team, and how to thwart the plans of adversarial agents are among the issues that arise in this multi-agent domain. Coordination needs not be centralized. When all agents have models of shared goals, they can support mechanisms to plan and enforce jointly agreed intentions. Plan recognition techniques can create models of adversarial plans which can be incorporated into the planning process.

Grid applications could be designed to be more flexible with architectures that support declarative representations of overall tasks and goals, and manage delegation and coordination among individual components that may then act autonomously to make their contributions to the overall task.

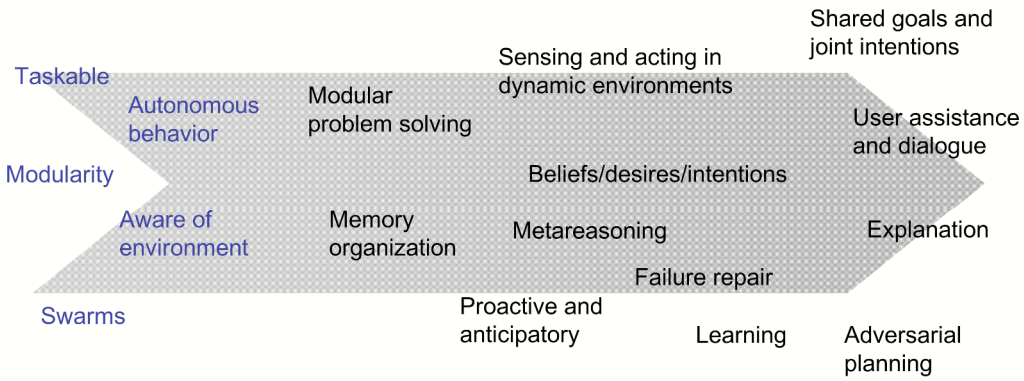

Figure 3. Increased sophistication in agent architectures that could benefit grid computing: from basic capabilities such as modularity, autonomy, swarm-level coordination, and immersion in their environment, to more advanced capabilities such as problem solving and memory organization, metareasoning, learning, shared goals, adversarial reasoning, and human-system collaboration.

3.9 Summary: Why Agents

Agent architectures offer valuable techniques to provide the autonomy and flexibility required in highly dynamic and heterogeneous environments. These are defining characteristics of grid environments. Figure 3 summarizes some of the capabilities discussed here, and highlights the increasing sophistication that agent architectures can support. Grid applications would benefit from incorporating such techniques.

4 Conclusions

Agent systems embody current AI research on distributed intelligent systems. Yet, agent systems have drawn in very limited ways from distributed computing, which hinders their scalability and long-lived operations. Grids, in turn, aim to develop similarly intelligent capabilities but from the bottom up,

and have concentrated more on managing low-level resources than higher-level tasks. Cross-cutting research across both areas would enable significant progress in grids as well as agents research.

Semantic grids bring declarative representations of knowledge to grids, and once that knowledge is available there are various cognitive capabilities typical of agent systems that become possible including learning, self-repair, planning, meta-reasoning, coordination, and communication abilities. The result will be *cognitive grids*, a new generation of distributed intelligent systems that will be significantly more capable, autonomous, and adaptive.

Acknowledgements

Special thanks to Varun Ratnakar for the implementation of agents as grid services. I would like to thank Ewa Deelman and Carl Kesselman for many fruitful discussions on the topics put forward by this paper, and other members of ISI's Center for Grid Technologies for their technical support with OGSA and their comments on this work, especially Ben Clifford, Carl Czajkowski, and Hongsuda Tangmunarunkit. Thanks also to Jim Blythe, Jihie Kim, Craig Knoblock, and to other researchers at ISI's Intelligent Systems Division and the Electric Elves project. This research was supported in part by the National Science Foundation's Shared Cyberinfrastructure program under grant SCI-0455361, and in part by an internal grant from USC's Information Sciences Institute.

References

[Allen et al 01] Allen, J, Donna Byron, Myroslava Dzikovska, George Ferguson, Lucian Galescu, and Amanda Stent, "Towards Conversational Human-Computer Interaction," AI Magazine, 2001.

[Anderson et al 04] Anderson, J. R., Bothell, D., Byrne, M. D., Douglass, S., Lebiere, C., and Qin, Y. "An Integrated Theory of the Mind", Psychological Review 111, (4). 1036-1060, 2004.

[Berners-Lee et al 01] Berners-Lee, T., Hendler, J., and Lassila, O. "The Semantic Seb", Scientific American 284(5):35-43, 2001.

[Berman et al 03] Berman, F., Hey, A. J.G., and Fox, G. "Grid Computing: Making The Global Infrastructure a Reality" John Wiley & Sons, 2003.

[Blythe et al 03a] Jim Blythe, Ewa Deelman, Yolanda Gil, Carl Kesselman. "Transparent Grid Computing: A Knowledge-Based Approach", Proceedings of the 15th Annual Conference on Innovative Applications of Artificial Intelligence (IAAI), August 12-14, 2003, Acapulco, Mexico.

[Blythe et al 03b] Jim Blythe, Ewa Deelman, Yolanda Gil, Carl Kesselman, Amit Agarwal, Gaurang Mehta, Karan Vahi. "The Role of Planning in Grid Computing", Proceedings of the 13th International Conference on Automated Planning and Scheduling (ICAPS), June 9-13, 2003, Trento, Italy.

[Bratman et al 88] Bratman, M. E., Israel, J.D., and Pollack, M.E. "Plans and Resource-Bounded Practical Reasoning", Computational Intelligence, 4(4):349-355, 1988.

[Chalupsky et al. 02] Hans Chalupsky, Yolanda Gil, Craig Knoblock, Kristina Lerman, Jean Oh, David Pynadath, Tom Russ, Milind Tambe. Agent Technology to support Human organizations. In AI Magazine, Vol 23, No 2, Summer 2002.

[Cheyer and Martin 01] Cheyer, Adam and Martin, David. "The Open Agent Architecture". Journal of Autonomous Agents and Multi-Agent Systems, vol. 4 , no. 1, pp. 143-148, March 2001.

[Cypher 93] Cypher, A. (ed) "Watch What I Do: Programming by Demonstration", MIT Press, Cambridge, MA, 1993.

[De Roure et al 01] De Roure, D., Jennings, N. and Shadbolt, N. "Research Agenda for the Semantic Grid: A Future e-Science Infrastructure". Technical report UKeS-2002-02, UK e-Science Technical Report Series, National e-Science Centre, Edinburgh, UK. December 2001.

[De Roure et al 04] De Roure, D.C., Gil, Y, and Hendler, J.A. (eds). IEEE Intelligent Systems, Special Issue on E-Science, Volume 19, Issue 1 (January/February), 2004. ISSN: 1094-7167.

[De Roure et al 05] De Roure, D., Jennings, N. and Shadbolt, N. "The Semantic Grid: Past, Present, and Future" Proceedings of the IEEE, Volume 93, Issue 3, March 2005.

[De Roure 05] De Roure, D. "Agents and the Grid – a personal view of the opportunity before us", Agentlink Newsletter, Issue 17, April 2005.

[Deelman et al 03a] Ewa Deelman, Jim Blythe, Yolanda Gil, Carl Kesselman, Gaurang Mehta, Karan Vahi, Kent Blackburn, Albert Lazzarini, Adam Arbree, Richard Cavanaugh, and Scott Koranda. " Mapping Abstract Workflows onto Grid Environments", Journal of Grid Computing, Vol. 1, No. 1, 2003.

[Deelman et al 03b] Ewa Deelman, Jim Blythe, Yolanda Gil, and Carl Kesselman. "Workflow Management in GruPhyN", In Grid Resource Management, J. Nabryski, J. Schopf, and J. Weglarz (Eds), Kluwer 2003.

[Deelman et al 03c] Ewa Deelman, Jim Blythe, Yolanda Gil, Carl Kesselman, Scott Koranda, Albert Lazzarini, Gaurang Mehta, Maria Alessandra Papa, and Karan Vahi. "Pegasus and the Pulsar Search: From Metadata to Execution on the Grid". PPAM Applications Grid Workshop (AGW), Czestochowa, Poland, 2003.

[Deelman et al 04] Ewa Deelman, Jim Blythe, Yolanda Gil, Carl Kesselman, Gaurang Mehta, Sonal Patil, Mei-Hui Su, Karan Vahi, and Miron Livny. "Pegasus: Mapping Scientific Workflows onto the Grid". Across Grids Conference, Nicosia, Cyprus, 2004.

[Dent et al. 92] Dent, C. L., Boticario, Jesus, McDermott, John P., Mitchell, Tom M., and Zabowski, David. "A Personal Learning Apprentice", Proceedings of Proceedings of the Tenth National Conference on Artificial Intelligence (AAAI-92), San Jose, CA, July 12-16, 1992.

[Dietterich 00] Dietterich, T. "Hierarchical Reinforcement Learning with the MAXQ Value Function Decomposition", JAIR 13: pp. 227-303, 2000.

[Euzenat 01] Euzenat, J. (Ed.). "Research Challenges and Perspectives of the Semantic Web." Report from the joint European commission and National Science Foundation, Strategic workshop on the semantic web. 3-5 October 2001, Sophia Antipolis, France. Report available from http://www.ercim.org/EU-NSF/semweb.html.

[Finin et al 94] Tim Finin, Yannis Labrou, James Mayfield. "KQML as an agent communication language". Proceedings of the 3rd International Conference on Information and Knowledge Management (CIKM'94), 1994.

[FIPA 02] "FIPA ACL Message Structure Specification", http://www.fipa.org/specs/fipa00061, 2002.

[Foster et al 02] Foster, I., Kesselman, C., Nick, J. and Tuecke, S. "The Physiology of the Grid: An Open Grid Services Architecture for Distributed Systems Integration". Globus Project, 2002, www.globus.org/research/papers/ogsa.pdf

[Foster et al 04] Foster, I., Jennings, N. R., and Kesselman, C. "Brain Meets Brawn: Why Grid and Agents Need Each Other". Proceedings of 3rd Int. Conf. on Autonomous Agents and Multi-Agent Systems (AAMAS), New York, USA, 2004.

[Foster and Kesselman 04] Foster, I., and Kesselman, C. "The Grid: Blueprint for a New Computing Infrastructure", 2nd Edition, Morgan Kaufmann, 2004.

[Foster et al 05] Foster, I. Czajkowski, K. Ferguson, D.E. Frey, J. Graham, S. Maguire, T. Snelling, D. Tuecke, S. "Modeling and managing State in distributed systems: the role of OGSI and WSRF". Proceedings of the IEEE, 93 (3), March 2005.

[Gil 93] Gil, Y. "Efficient Domain-Independent Experimentation", Proceedings of the Tenth International Conference on Machine Learning (ICML-93), Amherst, MA, June.

[Gil 06] Gil, Y. "Workflow Composition: Semantic Representations for Flexible Automation", in "Workflows for e-Science", Deelman, E., Gannon, D. Shields, M., and Taylor, I. (Eds), Springer Verlag, 2006.

[Gil and Ratnakar 03] Gil, Y. and Ratnakar, V. "Multi-Agent Systems and Grid Services: Towards Robust Continuous Distributed Problem Solving". Internal Project Report, October 2003.

[Gil et al 04] Yolanda Gil, Ewa Deelman, Jim Blythe, Carl Kesselman, and Hongsuda Tangmurarunkit. "Artificial Intelligence and Grids: Workflow Planning and Beyond", IEEE Intelligent Systems, January 2004.

[Gil et al 05] Yolanda Gil, Varun Ratnakar, and Ewa Deelman. "Augmenting Metadata Catalogs with Semantic Representations", Short paper at the Fourth International Semantic Web Conference (ISWC-05), Galway, Ireland, November 7-10, 2005.

[Gil et al 06] Yolanda Gil, Varun Ratnakar, and Ewa Deelman. "Virtual Metadata Catalogs: Augmenting Metadata Catalogs with Semantic Representations", International Provenance and Annotation Workshop (IPAW'06), Chicago, IL, May 3-5, 2006.

[Grosz and Kraus 99] Grosz, Barbara and Sarit Kraus. "The Evolution of SharedPlans." In Foundations of Rational Agencies, A. Rao and M. Wooldridge, eds., Kluwer Academic Press, pp. 227-262.

[Goble and De Roure 04] Goble, C. and De Roure, D. "The Semantic Grid: Myth Busting and Bridge Building," Proceedings of the 16th European Conference on Artificial Intelligence (ECAI-2004), Valencia, Spain, 2004.

[Hengst 00] Hengst, B. "Generating Hierarchical Structure in Reinforcement Learning from State Variables", Lecture Notes in Artificial Intelligence, 2000.

[Jennings et al 98] Jennings, N., Sycara, K., and M. Wooldridge. "A Roadmap of Agent Research and Development" Autonomous Agents and Multi-Agent Systems, Vol. 1, No. 1, July, 1998.

[Kaelbling et al 96] Kaelbling, L. P., Littman, M., L., and Moore, A. W. "Reinforcement Learning: A Survey", Journal of Artificial Intelligence Research, Vol 4, pp. 237-285, 1996.

[Kim et al 04] Jihie Kim, Marc Spraragen, and Yolanda Gil. "An Intelligent Assistant for Interactive Workflow Composition", In proceedings of the 2004 International Conference on Intelligent User Interfaces (IUI), Madeira Islands, Portugal, January 2004.

[Knoblock 94] Craig A. Knoblock. "Automatically generating abstractions for planning". Artificial Intelligence, 68(2), 1994.

[Lieberman 01] Lieberman, H. (ed), "Your Wish Is My Command: Programming by Example", Morgan Kaufmann, San Mateo, CA 2001.

[Luck et al 05] "Agent Technology: Computing as Interaction -- A Roadmap for Agent-Based Computing", Compiled, written and edited by Michael Luck, Peter McBurney, Onn Shehory, Steve Willmott and the AgentLink

Community, September 2005. Report available from http://www.agentlink.org/roadmap.

[Maechling et al 05] P. Maechling, H. Chalupsky, M. Dougherty, E. Deelman, Y. Gil, S. Gullapalli, V. Gupta, C. Kesselman, J. Kim, G. Mehta, B. Mendenhall, T. Russ, G. Singh, M. Spraragen, G. Staples, and K. Vahi. "Simplifying Construction of Complex Workflows for Non-Expert Users of the Southern California Earthquake Center Community Modelling Environment". In ACM SIGMOD Record, special issue on Scientific Workflows, 2005.

[Morley and Myers 04] D. Morley and K. Myers. "The SPARK Agent Framework", in Proceedings of the Third International Joint Conf. on Autonomous Agents and Multi Agent Systems (AAMAS-04), New York, NY, pp. 712-719, July 2004.

[Muscettola et al 98] Nicola Muscettola, P. Pandurang Nayak, Barney Pell, and Brian C. Williams. "Remote agent: To boldly go where no AI system has gone before". Artificial Intelligence, 103(1/2), August 1998.

[Newell 90] Newell, A. "Unified Theories of Cognition". Harvard University Press. Cambridge, MA, 1990.

[Parr and Russell 97] Parr, R., and Russell, S. "Reinforcement Learning with Hierarchies of Machines", Advances in Neural Information Processing Systems, 10, 1997.

[Rao and Georgeff 98] Rao, A., and Georgeff, Michael P. "Decision Procedures for BDI Logics", J. Log. Comput. 8(3): 293-342, 1998.

[Riemenschneider et al 04] Riemenschneider, R., Saïdi, H., and Dutertre, B. "Using Model Checking to Assess the Dependability of Agent-Based Systems". IEEE Intelligent Systems, Vol. 19, No. 5, 2004.

[Sidner et al 03] Sidner, C.L.; Lee, C.H.; Lesh, N.B., "Engagement Rules for Human-Robot Collaborative Interactions", IEEE International Conference on Systems, Man & Cybernetics (CSMC), ISSN: 1062-922X , Vol. 4, pp. 3957-3962, October 2003.

[Tuchinda et al 04] Rattapoom Tuchinda, Snehal Thakkar, Yolanda Gil, and Ewa Deelman. "Artemis: Integrating Scientific Data on the Grid", Proceedings of the 16th Annual Conference on Innovative Applications of Artificial Intelligence (IAAI), San Jose, CA, July 25-29, 2004.

[Tuecke et al. 03] Tuecke, S. et al. "The Open Grid Services Infrastructure (OGSI)", V 1.0, April 2003.

[Urmson 03] C. Urmson, R. Simmons and I. Nesnas. "A Generic Framework for Robotic Navigation". In Proceedings of the 2003 IEEE Aerospace Conference, Big Sky Montana, March 8-15 2003.

[van Lent and Laird 99] van Lent, M., and Laird, J. "Learning Hierarchical Performance Knowledge by Observation", ICML, 1999.