Dynamic Grid tasks composition and distribution through agents

A. Negri, A. Poggi*,†, M. Tomaiuolo and P. Turci

Dipartimento di Ingegneria dell’Informazione, Università degli Studi di Parma, 43100 Parma, Italy

SUMMARY

This paper presents a multi-agent system called GAIN (Grid Agent Infrastructure), which can be used for the development of flexible agent-based Grid systems. The system supports users both in the development and execution of Grid applications. In particular, GAIN allows the definition of workflow applications by composing different tasks made available by the Grid nodes. Furthermore, it follows the different phases of the execution of the workflow providing transparent allocation and re-allocation of the tasks on the different nodes of the Grid. A first prototype of the system has been realized by using the JADE agent development software and it has been tested in lab trials involving a network of different JADE platforms. In these trials, tasks provided both by the agents of the system and by external legacy software systems have been composed. Copyright © 2005 John Wiley & Sons, Ltd.

KEY WORDS: multi-agent system; Grid; workflow; task composition and distribution; JADE

1. INTRODUCTION

Computational Grids are geographically distributed environments for high-performance computation [1]. A Grid couples a wide variety of geographically distributed resources (such as PCs, workstations and clusters, storage systems, data sources, databases and special purpose scientific instruments) and presents them as a unified integrated resource. This is possible due to a set of global services that any Grid should provide (information services, resource management, data transfer, security, and so on).

Unfortunately the task of building Grid applications remains extremely difficult because there are few tools available to support developers. To build reliable and re-usable Grid applications,
programmers should be equipped with a set of tools that hides the details of most Grid services and provides developers with a consistent, non-complex model in which applications can be composed from well tested, reliable sub-units. For some time, Grid workflows have been emerging as an important alternative to develop Grid applications. In fact, Grid workflow management systems are evolving towards a system that will offer both the definition of applications at a high level of abstraction (without dealing with implementation details) and their automatic deployment and execution on the basis of the available resources in the Grid. Moreover, the use of agents has been proved a good means for intelligent task distribution and for supporting users in both on-line and off-line activities.

In this paper we present GAIN (Grid Agent Infrastructure), a multi-agent system that supports users both in the development and execution of Grid applications defined as workflow-based composition of atomic and compound tasks. The next section describes the GAIN system. Section 3 presents the main aspects of its implementation and some preliminary evaluation results. Section 4 discusses the most interesting related work and considers the problem of integrating workflow, Grid and agent technologies. The paper ends by outlining some considerations regarding the current prototype of the GAIN system and the future research directions to improve such a system.

2. THE GAIN SYSTEM

GAIN is a multi-agent system that supports users both in the development and execution of Grid applications. In particular, this system allows (i) the definition of workflow applications by composing the different tasks provided by the Grid nodes, (ii) the transparent distribution of workflow tasks on the Grid nodes, (iii) their execution and, if necessary, (iv) their re-allocation on different nodes.

The system is based on six different kinds of agents: personal agents, component managers, task description managers, workflow managers, e-mail agents, starter agents and directory facilitators.

Personal agents are the agents that allow the interaction between the user and the different parts of the system and, in particular, between the users themselves. User-agent interaction can be performed in two different ways: through a graphical user interface when the user is active in the system or through e-mails when it is off-line.

Component managers provide the execution of tasks either through their internal resources or wrapping some legacy software systems. In the latter, there can be a direct connection between the agent and the legacy software system (e.g. using a software API or TCP port connection) or their interactions are managed through a Web service.

Task description managers are responsible for informing an agent about the types of tasks that the component managers can perform (e.g. a personal agent can ask a task description manager about the tasks, suitable for a specific application domain, provided by the component managers of the GAIN system).

Workflow managers are the principal actors of the GAIN system. In fact, they are responsible for (i) distributing the tasks requested by the users, (ii) monitoring task execution and if necessary reallocating tasks to other component managers (e.g. in the case of failure or when a task cannot be executed with the fixed constraints), (iii) collecting the results, and (iv) informing the users about the success or failure of the executions. In the case of success, they forward the results to the users through their personal agents.
E-mail managers are responsible for the exchange of e-mails between off-line users and their personal agents.

Starter agents are responsible for activating a personal agent when either a user logs on or another agent requests it.

Directory facilitators are responsible for informing an agent about the address of the other agents active in the system (e.g. a personal agent can ask a directory facilitator about the addresses of the component managers able to perform a particular task).

A GAIN system is composed of a federation of GAIN platforms. Each GAIN platform has at least a directory facilitator, its agents can be distributed on different computation nodes and the legacy software systems managed by the component managers of the platform can also be located in different computational nodes. The federation of the platforms is built by organizing a hierarchy of directory facilitators enabling the interaction between any couple of agents in the federation.

Figure 1 shows a graphical representation of a GAIN platform that identifies the interactions between the different kinds of agents and between agents and users. Note that agents can be distributed in different computational nodes.
2.1. Workflow generation

In this phase, the user defines the workflow she/he would like to execute by composing the tasks that can be performed by the different nodes of the GAIN Grid.

The personal agent helps its user by retrieving the task descriptions of those related to the user’s application domain and that can be executed on the GAIN Grid. This operation is performed through the following steps (see also Figure 2).

1. The user asks its personal agent about the tasks executable on a certain domain in the Grid.
2. The personal agent asks the directory facilitator about a task description manager.
3. The directory facilitator returns the address of the nearest task description manager.
4. The personal agent asks the task description manager about the tasks applicable in the required domain.
5. The task description manager asks the directory facilitator about the component agents performing tasks in the required domain.
6. The directory facilitator returns the list of the addresses of the component agents satisfying the query.
7. The task description manager asks the listed agents about the tasks they can perform in the required domain.
8. Each component agent returns a list containing the description of the tasks satisfying the query.
9. The task description manager collects the descriptions (eliminating duplicates) and then sends the whole list to the personal agent that forwards it to its user.

After the reception of the list of task descriptions, the user can start the definition of the workflow. Workflows are described by using the XPDL workflow language [2], an XML-based language defined by the Workflow Management Coalition [3]. The workflows are generated using JaWE [4], an XPDL editor that provides a graphical interface for building new XPDL files. We have used XPDL instead of the representation formats currently used by the Grid community (e.g. Taverna [5] and GridAnt [6]) because we think it is important to try to refer to international standards when possible and because XPDL is one of the most used workflow definition languages with the advantage of a continuous improvement of the language, of its documentation and of its support tools and software libraries.

A workflow is composed of a set of linked activities which collectively realize the objective pursued by the user. The ‘activity’ element in the XPDL language is the basic building block of a workflow process definition. In our domain each activity corresponds to a Grid task and can be linked to other elements of the workflow through a set of ‘transitions’. XPDL supports only two kind of transitions, specifically (i) transitions to a set of incoming activities through a synchronous (AND) or asynchronous (XOR) join, or (ii) transitions to a set of outcoming activities through a parallel (AND) or alternative (XOR) split. It is possible to insert some transition restrictions such as conditions or synchronization points. XPDL also supports hierarchical decomposition of activities as the execution of a sub-flow: in this specific case an activity is a link to another XPDL file and workflow. In GAIN this functionality is used to realize some required loops and to delegate the execution of a specific part of a workflow to another workflow manager.

As far as constraints are concerned, XPDL allows by default only the use of (i) waiting time, as the amount of time which is needed to prepare the performance of a task, and (ii) working time, as the amount of time the performer of the activity needs to perform the task. The ‘limit’ property is used
to set a time constraint for the execution of a task: if the limit set is exceeded, the workflow manager stops the task and computes some other operations, according to the rules established by the user.

The need of using, in a XPDL workflow description, some other constraints, e.g. resource-specific performance limits, is supported by implementing an ExtendedAttribute. An Extended Attribute is characterized by a couple of attributes ‘name’/‘value’ plus some case-specific code. As a simple example, if we need to delegate a task on a machine with a minimum amount of RAM, we could define a new ExtendedAttribute for the activity that represents that task, as follows: `<ExtendedAttribute name="RAMLowerBound" value="64">` (we are asking to perform the task on a machine with more than 64 MB of RAM free).

Therefore, the definition of a GAIN workflow corresponds to (i) the identification of the Grid tasks involved in the workflow, (ii) the identification of the links connecting the different tasks, (iii) the mapping of input and output parameters of each task to the parameters of the other linked tasks, and (iv) the definition of the execution time constraints for the different parts of the workflow.

2.2. Workflow execution

After the definition of the workflow, the user can submit the workflow to the GAIN system for its execution. Execution is managed by a workflow manager and a set of component managers that realize together a distributed workflow engine. The submission and the workflow execution involve different
phases of negotiation based on the contract net protocol [7]. Such a protocol has been proved to be one of the most flexible and efficient negotiation mechanisms used in multi-agent systems. In particular, it allows us to realize easily and in a short time a task distribution algorithm that offers a distributed implementation, stability and fast convergence. Moreover, the simplicity of the implementation also allows the rapid modification of the criteria for bid assignment. The submission and the workflow execution can be described by the following steps (see also Figure 3).

(1) The user asks its personal agent to manage the execution of the workflow and inform it if it can relax the time constraints without asking her/him.
(2) The personal agent announces the task of executing the workflow to the workflow managers.
(3) Each workflow manager able to execute the workflow submits a bid, describing mainly when it can be executed.
(4) The personal agent assigns the workflow to the workflow manager that has submitted the best bid and informs it if it can relax time constraints without an authorization.
(5) The chosen workflow manager gets the workflow and submits all the task descriptions contained in the workflow to the component managers of the GAIN system. In particular, each task description is submitted only to the component managers that can execute it.

(6) Each component manager able to execute the task within the time constraints indicated in the task description submits a bid describing mainly when it can be executed.

(7) The workflow manager assigns the task to the component manager that has submitted the best bid.

(8) After the assignment of all the tasks, the workflow manager completes the distribution of the information necessary for the execution of the tasks submitted to the chosen component managers. The information concerns mostly the links between the tasks they must execute, the other tasks of the workflow and, of course, the component managers that will execute them.

(9) At this point, the workflow manager is able to start the execution (activating the different component managers involved in the workflow) and monitor its different phases.

(10) On completion of the execution, the workflow manager collects the results and sends them to the personal agent that presents them to its user if she/he is still logged onto the system, otherwise, it sends her/him an e-mail about the success of the execution.

In the current version of the GAIN system the submission of the workflows to the corresponding workflow managers is simply done by sending them the XPDL file representing the whole workflow. After the reception of the file, the workflow manager extracts all the activities and the related data fields, separates them according to the rules imposed by the user and sends them to the component managers, formerly selected, for the completion of the task.

If an activity in the workflow is a sub-flow delegation, the workflow manager sends the whole XPDL file and all the other files involved in the workflow description to another workflow manager. This kind of operation is done recursively for every sub-flow. We chose to create a new agent for each new sub-flow in order to preserve the linearity and consistency of the design of the system, even if there is a computational and communication overload due to the creation and consequent management of a new agent in the JADE platform.

### 2.3. Failure management

What has been described in the previous subsection is what will happen if no problems arise. However, sometimes problems can arise in the task assignment as well as in the execution phase.

In the assignment phase, the most common problem is that no component manager can satisfy the time constraints for a particular task and the workflow manager is not authorized to relax the time constraints. In this case, the behaviour of the system can be described by the following steps.

1. The workflow manager informs the personal agent about the problem.
2. The personal agent informs the user about the problem asking for authorization to relax the time constraints of this specific task. In this case, the personal agent either interacts directly with the user, if the user is still connected through the graphical user interface offered by the system, or through the exchange of e-mails delegating their sending and reception to the e-mail manager.
3. The user sends its decision to the personal agent that forwards it to the workflow manager.
4. If the decision is positive, the workflow manager continues the assignment of the tasks, otherwise it cancels the previous assignments and drops the management of that workflow.
In the execution phase, different problems can happen, but two problems are the most important: (i) a component manager involved in the workflow either ceases to be available or cannot execute the promised task; or (ii) a component manager is executing a task, but the time execution has already exceeded the time constraints.

In the first case, the workflow manager may resolve the problem submitting a new bid for the task and, if necessary, asking for the relaxation in the time constraints.

The solution of the second case is more problematic, because it is not possible to recognize if the execution will be completed with a reasonable delay. In this case, besides asking to relax the time constraints (if necessary), the solution provided for this problem is:

1. submit a new bid for the task;
2. execute that task in parallel with the delayed task;
3. abort the execution of the remaining task as soon as the execution of the other task has been completed.

3. SYSTEM IMPLEMENTATION AND EXPERIMENTATION

GAIN is still under development, but the first prototype has been realized by using the Java Agent Development (JADE) framework [8,9]. JADE is a software framework to aid the realization of agent applications in compliance with the FIPA specifications for interoperable intelligent multi-agent systems [10]. JADE is an Open Source project and the complete system can be downloaded from JADE Home Page [11].

Given the distributed nature of JADE-based agent systems, a GAIN system can be distributed on a set of agent platforms, situated in different parts of the world and connected via the Internet through the openNet Services Network Infrastructure Services [12]. Each agent platform can be distributed on different computation nodes and is connected to a Web server to allow direct interactions with the users, and to a mail server, enabling e-mail interactions with the users. In each agent platform there is a unique starter agent, broker agent and e-mail agent, but there might be more than one workflow manager and component manager. This usually happens when the agent platform is distributed on different nodes in order to cope with performance issues due mainly to the number of tasks and users to be managed. Finally, there can be one or more directory facilitators. In the case of more than one directory facilitator, these agents build a hierarchical map of the agents of the system; therefore, when an agent is created, the only information it needs to know is simply the address of the main (root) directory facilitator.

The first prototype of the system has been completed. This prototype does not cover the functionalities of the GAIN system related to the workflow generation that, in the current implementation, is realized off-line. The workflow information is exchanged with the GAIN systems through files: (i) the personal agent writes the description of the available tasks into a set of files, (ii) the user builds the workflow through the JaWE XPDL editor [13], and (iii) the personal agent gets the file containing the workflow description. Up to now experimentation has had the main goal of testing the functionalities of the system. This experimentation has been done inside a ‘Lab Grid’, where we have been able to easily cause failures of software applications and hardware nodes without causing damage to other users. The ‘Lab Grid’ has been performed on a network of 11 Pentium IV computers connected by an 100 Mbps ethernet network. Each computer hosted a JADE container.
containing, in particular, some component managers that provided tasks both as internal services and as wrapping external services offered by legacy software (e.g. database management systems and rule-based engines). The result of the experimentation has validated all the functionalities of the current implementation including the most interesting and challenging, that is, the re-allocation of tasks in the case of failures (we simulated failures by killing Java virtual machines and legacy applications or by stopping computational nodes). Of course, we need to validate the system on a large network composed of a significant number of computers on which different interconnected JADE agent platforms run. This is feasible because the JADE agent software has been tested on large networks and provides support to build a federation of different agent platforms [14].

4. WORKFLOW, GRID AND AGENT TECHNOLOGIES

In recent years a lot of research work has been undertaken ranging from the use of workflows in distributed systems and, in particular, in the Grid (see [6,15–19]) to the use of agent technology in the Grid (see [4,20–24]) and the use of agent technology for the management of workflows (see [22,25–27]). In particular, we have taken our cue from the work of Buhler and Vidal [22,28], who propose the use of agents for the dynamic composition of Web services through workflows, and from GridFlow [21], in which agents are mainly used for the management of Grid resources.

The reasons for integrating agent and Grid technologies have been illustrated in different works (see, for example, [4]). In particular, this marriage allows the robustness of Grid infrastructure to be coupled with the flexibility of agent systems. However, the GAIN system has a more ambitious goal, that is, to prove agents as technologies for Grid applications based on the composition of Web services. In fact, the agent technology used in GAIN, i.e. JADE, provides good reliability and the possibility of distributing a system on a geographic network of computational nodes; moreover, different works have studied and experimented with the integration of JADE agents and Web services (see, for example, [29,30]) and the next release of the JADE software (3.3) will be providing a library for such an integration. Of course, a limit of the agent-based approach for some kinds of applications may be performance. In this case, an integration of agents with traditional Grid infrastructures (e.g. Globus) may be the right solution. We are thinking about this possibility and our idea is to use component agents as interfaces towards Globus Grids. A GAIN task will therefore correspond to a distributed Globus Grid task.

The most important contribution of our work is the use of agents to support the activities involved in the development and execution of a Grid application, i.e. the workflow generation, the distribution of workflow tasks, the control of their execution and finally the re-allocation of tasks in the case of failure of some Grid components. A lot of work has been done on the problem of task allocation in Grid systems (see, for example, Loadleveler [31], Condor [32] and Condor-G [33]). The main difference between these approaches and the one offered by the GAIN system is that, while the traditional Grid approaches submit, monitor and, if necessary, move a task from a computational resource to another resource, GAIN provides a more flexible and extensible way of managing tasks. In fact, tasks are usually managed in a similar way to traditional Grid approaches, but in special situations or applications new resource management techniques can be used (e.g. the execution of a task can be replicated on different computational resources, a failed task can be replaced with one or a set of different tasks that achieves the same goal, etc.).
5. CONCLUSION

In this paper we have presented GAIN, a multi-agent system that supports users both in the development and execution of Grid workflow applications.

The first prototype of the GAIN system has been completed. GAIN implementation is based on a multi-agent society built on top of JADE [8,9,11], a software framework to aid the creation of agent applications in compliance with the FIPA specifications for interoperable intelligent multi-agent systems [10]. Moreover, the system takes advantage of the JaWE XPDL editor [2] as a tool for workflow generation. Initial system experimentation has been done inside a ‘Lab Grid’, where we have validated all the functionalities of the system including task distribution and re-allocation.

Besides continuing system experimentation and the extension of the tests in a Grid for real applications, we are working on the improvement of the integration of the workflow generation and execution phases, on the use of more advanced load-balancing techniques (now, tasks are rescheduled only if either a hardware or a software component fails or task execution time constraints are not respected), and on the realization of a specialized workflow editor with the aim of coping with the main problems of the current solution. In fact, the current implementation of the system uses a general purpose XPDL editor, that is the JaWE editor, which offers neither an easy integration (e.g. information exchange with the GAIN agents) nor an easy customization (e.g. graphical presentation of the available tasks, syntax checking for task constraints). Moreover, we have planned, after the realization of the complete GAIN system, to study and develop the software that will allow the integration of our agent-based system with the Globus Grid-based systems. In fact, this will allows us to use the services offered by the GAIN system together with those offered by Globus systems. In our view, this integration will let us use GAIN as an ‘intelligent composer’ of compound tasks, each of them possibly corresponding to a distributed Globus Grid task.

REFERENCES


