Trusting Agents for Grid Computing *

Justin R.D. Dyson, Nathan E. Griffiths, Hélène N. Lim Choi Keung,
Stephen A. Jarvis, Graham R. Nudd
University of Warwick
Coventry, UK
justin@dcs.warwick.ac.uk

Abstract - *The Grid vision is to allow compute resources to be shared and utilised globally, with these distributed resources belonging to the same Virtual Organisation (VO). These resources execute jobs submitted by users, who are not in the resources' local domain and hence have no control over these resources. Conversely these users are not controlled by the resource owners. Certificates provide a common, useful security mechanism to overcome these barriers and set out access rights, but they do not guarantee that the resources, or users, can be trusted. This is due to the fact that resources and users may be unreliable; this situation may not be reflected in the users' perception of the reliability of the resource owner as a whole or vice versa. This paper describes a trust framework model for Grid computing, which enables users to execute their jobs on reliable and efficient resources, thereby satisfying clients' quality-of-service (QoS) requirements.*

Keywords: Grid, Trust, Agents, Security, Resource Management

1 Introduction

Grid computing [3] can be seen as a multi-agent system facilitating the sharing of compute resources, allowing users to discover and use remote resources. Users are able to submit jobs to remote resources and typically have no explicit control over the resources themselves. Thus, both users and resources can be viewed as autonomous agents, having control of their own behaviour. This autonomy gives rise to inherent uncertainty, since an individual cannot predict how another will respond to changing situations. In this paper, the notion of trust provides a mechanism for such agents to manage the risk from interacting with others who may have different objectives, or may fail to fulfil their commitments.

Applying trust to Grid computing is a relatively new area of research. For example, based on the work of Abdul-Rahman et al. [1], Azzedin et al. [2] has proposed a trust framework to monitor security overheads and improve scheduling. Hui et al. [6] use the notion of "mission-aware" trust models, which take into account the cost of performing allocated tasks. However, trust has not been used to add negotiation and monitoring in a Grid environment, as is proposed in this paper.

We investigate the effectiveness of using trust to manage interactions in a Grid computing context. Furthermore, we describe a trust framework for Grid computing and explore a range of trusting dispositions. Finally, we describe a simulated Grid environment showing how users can choose to execute their jobs on reliable resources, and how resources can be utilised in an efficient way.

2 Trusting Agents

The environment is a multi-agent system comprising two fundamental types of agent: resource agents and user agents. The former corresponds to the resources that are available and the second corresponds to the users who are interested in utilising these resources. The physical entities onto which these logical resources can be mapped are any processing power, data storage, software, or any component which can be accessed and utilised over a network. Examples of typical Grid resources include telescopes, large databases, supercomputers and pools of desktop computers. Moreover, the classes of users are varied and examples include students, scientists, academics and corporate employees. In addition to user and resource agents, the multi-agent system includes negotiating agents to broker interactions between users and resources. A definition of these agents and their separate, yet interdependent roles, is given below:

- **User Agents** The goal of the user agents is to use the resources that are available to them, whilst meeting the requirements of the user and adhering to acceptable trust limits. User agents respond to requests for a particular resource usage. For example, a user may make a request to access a database that is situated in a remote location. The user agent will then need to find an appropriate resource.

- **Resource Agents** The agents which offer resource access have the goal of ensuring that the resources under their management are utilised to their maximum capacity, whilst keeping within their own trust limits. The resource agents not only process current requests, but they are also able to reserve resources for future usage. For example, a

user may wish to access a telescope immediately and may also want to reserve the usage of a database for a later time.

- **Negotiating Agents** Both user agents and resource agents have their own trusting dispositions and may trust other agents differently from their peers. Before user and resource agents can cooperate, the requirements of both have to be analysed and met. This is achieved by a mediating negotiation agent, which takes the requirements of a user agent and finds a suitable match for collaboration. Each user agent has its own negotiation agent who performs all the mediations for them. For example, a user agent may wish to find a suitable supercomputer for its user. However, there could be many available matching resources with different constraints and costs. The negotiating agent would then match the user requirements with the availability and characteristics of appropriate resources

2.1 Trust

As agents encounter each other, they may choose to interact. During this interaction, there is an associated degree of risk, which is inherent. The interaction could involve parties who have little or no prior experience of performing transactions with one another; therefore the potential outcome cannot be foreseen. The use of trust in such scenarios, where uncertainty is prevalent, can help assess the associated risk involved for the interaction to take place [7].

In this work, trust T in an agent α, based on Marsh's formalism [7] and the work of Griffiths [5], is represented using a value in the interval between 0 and 1: $T_\alpha \in [0,1]$. As this value approaches 0, the agent becomes increasingly distrusting and conversely, as it approaches 1 the agent has complete or blind trust. The value represents the view of an individual agent and cannot be directly compared with that of other agents due to its subjectivity.

Trust is initially set to a value according to the agent's disposition. This disposition determines both the initial value that trust takes and how trust is altered after an interaction with another agent. When the initial trust is represented by a low value, then the agent can be considered to be pessimistic, whilst conversely higher values represent optimism.

3 Agent Trust Framework

3.1 Adaptive Functions

As an agent encounters new experiences E with a particular agent α, its trust of that agent T_α is adjusted accordingly. An experience is represented as a tuple $E(S,EO)$, where the experience is the result of an outcome EO from a particular situation S. The situation can be any goal that the agent wishes to accomplish and the impact of the situation will be determined by the outcome. For example, if a resource fails 1 s before a request is completed, the impact for a 1 min duration request will be less than one which takes 1 week of execution. If the experience is considered to be negative, then a degrading function f_d is applied to trust. Conversely, for positive experiences, an increasing function f_i is applied. These functions use past experiences and the current trust value to calculate the trust change. Past experience is taken into account by using a sliding window of experiences, the size of which varies according to an individual agent. The value by which the function changes trust is also be affected by the disposition of the agent. For example, optimistic agents will use an f_i which will increase trust by a higher value than a pessimist.

Over time, trust values relating to past experiences will become inaccurate and outdated. A decay function f_γ is applied to converge the trust value to the initial value as set by the agent. This is applied regardless of whether the ascribed trust value represents distrust or trust; thus the positive effect of successful interactions on trust will reduce over time, as will the negative effect of unsuccessful interactions.

Each of these functions for manipulating trust can be defined by each individual agent. Equation 1 illustrates an example for the increasing function which is used in the scenario described in Section 4. Here, f_i is defined as the product of the current trust disposition and the entity's overall feeling of trust. The current view is come about by analysing and weighting all the conditions C set out before the cooperation and seeing whether the outcome O for each of the conditions was considered acceptable AO; then the mean average is taken. The overall trust $T_{overall}$ is a weighted value, which uses the set of past experiences $T_{previous}$ and the entity's disposition $T_{disposition}$.

$$f_i = T_{current} \times T_{overall} \tag{1}$$

$$T_{current} = \frac{\sum_{i=1}^{n} WT(C_i(AO_i, O_i))}{n} \tag{2}$$

$$T_{overall} = WT(\{T_{previous}\}, T_{disposition}) \tag{3}$$

3.2 Negotiating Usage

As the sharing of resources involves a degree of risk, a mediating agent called a negotiator is used during the allocation process. The purpose of the negotiator is to match potential resources with the requests that are submitted to the user agent. For example, a request could be put forward for the use of data storage in a known location, but the user may not know the exact storage

capabilities. Furthermore, it is possible that the user will not know where to locate such storage in the first instance. For the process to be efficient and consistent each time, a standardised negotiation mechanism is essential. It is crucial that user and resource constraints are stated clearly and unambiguously to avoid problematic situations.

A contract net [8] is based on the principle of contract tendering and is a possible solution for problem solving. Contracts are tendered using networks of communicating problem solvers; in this case the problem solvers are the resource providers. The steps involved when using a contract net are summarised in Figure 1. The process is started by the submission of a user request for resource usage by a user agent to a negotiating agent.

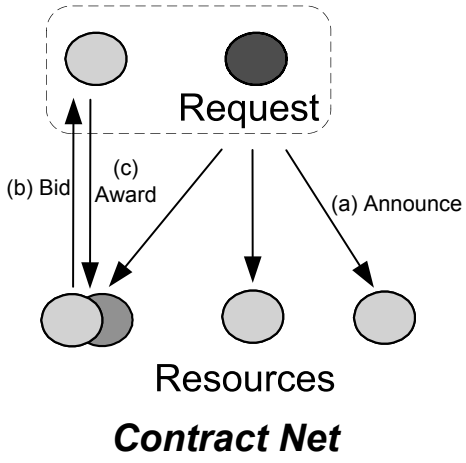

Figure 1: Contract Net

The negotiator inspects the request and sends out a request for tenders from the resource agents in its vicinity. The request would contain details about the resource needed and any constraints involved. For example, the request could be for general machine usage for a specific architecture. When a collection of tenders has been received or too much time has passed, the negotiator can select and award the contract to the most appealing resource agent. The contract terms are originally set out by the users as they expect a certain quality-of-service (QoS) when utilising resources and services, especially as they can be used at a financial cost. The terms are written out as conditions, as described in Section 3.1, where C is a condition which has an acceptable outcome, AO. The acceptable outcome AO can be a range of values to allow the negotiator to have a degree of flexibility when creating terms for resource usage or conversely be strictly defined; for example a resource may have to guarantee that a certain amount of physical memory is available.

The successful bid will also have a set of conditions applied, which may match those of the user agent to satisfy the potential contract, but may also set out extra conditions. These will clearly define all elements of the contract and protect its own QoS so as not to impact on other clients. For example, it may be the case that memory is also an important factor for other clients who are using the resource at the same time, so the user would have to take this into consideration when selecting the required resource.

Finally, trusting the party involved will determine the final outcome. Resource agents will not offer their services to users with whom they have had undesirable transactions in the past. A resource may offer the perfect solution for the user; however, it could have failed in the majority of occurrences in the past. Moreover, the user may turn down a contract if the selected resource is distrusted or may lessen the constraints set in the contract to use a more trusted resource. The process is summarised in Figure 2.

1. User agent receives a request from its and contacts a negotiator
2. Negotiator creates a contract using request conditions C_x
3. Request for tenders is sent to neighbouring resource agents
4. Resource agents fine tune and add their own conditions C_y, and make bids
5. Tenders are received, matched and selected
6. Finally, the contract is awarded to the resource agent, which can be trusted and where all conditions C_n are satisfactory

Figure 2: Negotiation Process

4 Scenario: Grid Computing

In the following section, the Grid [3, 4] is used as an example scenario, where the process of acquiring and utilising resources can be viewed as a multi-agent system. The framework proposed in Section 3 has been developed and its reliability tested; the approach and experimental results follow. The scenario chosen is that of a Campus Grid [9] which enables multiple projects or departments to share computing resources in a cooperative way. Campus Grids may also consist of local and dispersed workstations and servers, in departments or across the university. In this example, resources could include workstations, clusters and supercomputers, which are networked and publicly accessible.

4.1 Experimentation

To test the validity of our approach, a series of experiments were performed. A range of agents with differing dispositions were used: optimistic agents inferred

high trust values from their experiences, and pessimistic agents who inferred low values. Interactions were generated by user agents randomly requesting access to resources. A negotiation agent was used on behalf of the user agent to find a suitable resource. Once the negotiation process was completed, the resource could be utilised. After each interaction agents' trust values were updated. The experiments explored the different situations that may occur with different sets of user agents: a high proportion of optimistic agents, a high proportion of pessimistic agents and a balanced collection of user agents. This was achieved using a large set of requests (3000) and with trust being used by the following groups:

- none of the agents, therefore the level of satisfaction with each experience was not taken into account;
- the user multi-agent system solely;
- all agents with a low rate of deception by the user agents;
- all agents with a higher rate of deception.

4.2 Experimental Results

Figure 3 is an illustrative trace of the evolution of trust for a group of six different types of user agent. Each line represents the overall trust achieved as the user agent gained experience. Agents 1 through 3 are pessimists, agent 4 is cautious, agent 5 is highly optimistic in positive situations, yet cautious in negative ones. Finally, agent 6 is cautious in positive situations and slightly cautious in negative ones. The trace shows that the overall disposition for all the agents was negative. This can be explained as the resources used had a high failure rate. The graph demonstrates that agent 4, who was cautious, had a more realistic view of the actual system. The highly pessimistic agents, especially 1 and 2, found it difficult to trust other agents in any way, as they both encountered continuous adverse experiences. Even with such a high failure rate, the user agents saw a 2% increase in their number of positive experiences when using trust.

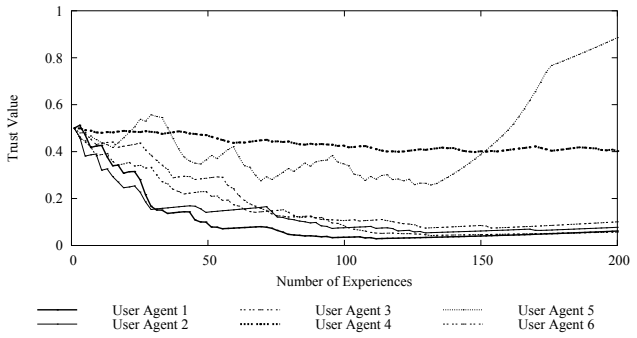

Figure 3: Trust Change using Unreliable Resources

A similar trend can be seen in Figure 4, though in this instance the resources were on average 30% more reliable. This reliability saw the number of jobs completing rise approximately from 56% in the results shown in Figure 3 to over 65% in Figure 4. The improvement in reliability can be seen from the graph, as agents 4 and 5 managed to gain positive trust for a larger continuous number of experiences. In fact, the rate of change towards distrust is reduced for all the agents. As in the previous scenario, the number of positive experiences per agent increased, this time by an average of 5%. This continuous increase in the number of positive experiences can be attributed to the fact that by using trust, the user agents began to distrust the unreliable resource agents they were cooperating with and consequently looked for alternatives. The resources that were then utilised might not have been as satisfactory or attractive as their predecessors, but were not distrusted at this point. Therefore, as some of the resources were more reliable to some extent, the number of positive experiences increased.

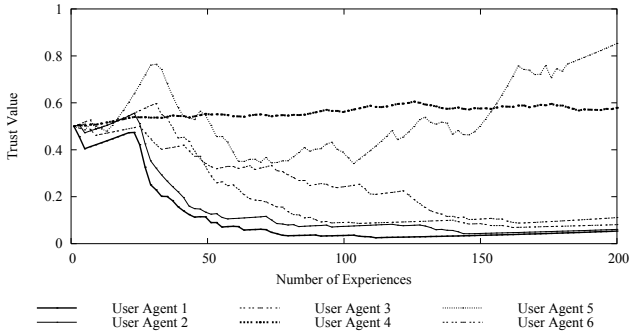

Figure 4: Trust Change using Resources with Improved reliability

Figures 5 and 6 show traces for when the resource agents were more reliable than the user agents; in Figure 6 the users were 10% more reliable. The increase in reliability can be seen in Figure 6, where the majority of agents show an increase in the average number of positive experiences. In both traces it can be seen that user agent 4, who was the most cautious again had the most stable trace and the most positive experiences. Even though the peers utilising the same resources as agent 4 were unreliable, the resource agents themselves were using trust. This meant that unreliable users were filtered out and this allowed agent 4 to utilise the resources freely. It can also been observed that the resource agents, at least for the first 50 experiences for each agent, was trusted. This behaviour can be attributed to the fact that resources were not heavily loaded in the early stages as users built trust around all those which were available.

As the users gained experience, the number of resources they were utilising decreased. At this stage, user error became amplified and they were unable to deal with the increased unreliability. Therefore, after 50 experiences, the pessimistic user agents started to see a sharp increase in negative experiences.

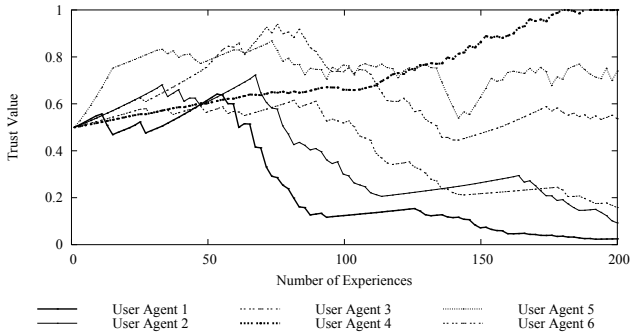

Figure 5: Trust Change for Unreliable User Agents

The number of jobs completed for the set of resources used for a typical set of resources is shown in Figure 7 and the contrary is shown in Figure 8. The average failure rate in both charts is approximately 50%. Resource agent 2 offered excellent resources with little constraint. However, as can been see in Figure 7, the failure rate was high. The general failure rate for all the resources was high in this experiment and the agents, when using trust, still found failure with the alternative resources. Additionally, the number of jobs submitted to each resource agent rose by an average of 25% when trust was utilised showing the user agents were avoiding the distrusted resource agent 2.

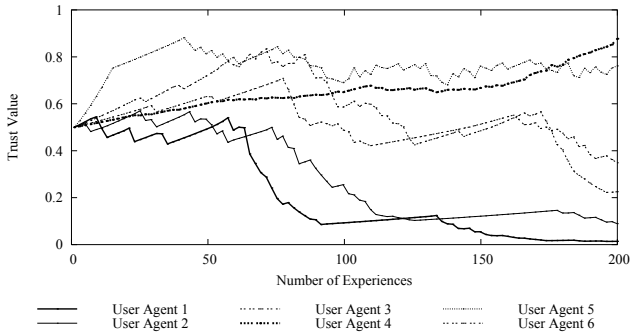

Figure 6: Trust Change for User Agents with Improved reliability

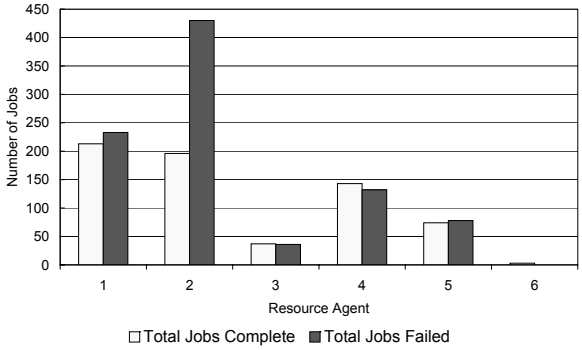

Figure 7: Completed Jobs with no Trust Model

Figure 8: Completed Jobs with Trust Model

The experiments were repeated to give 9 sets of results as shown in Figure 9. Agents with differing dispositions were used where trust model A used a mixture of agents; trust model B used a set of pessimists and trust model C used optimistic agents. Furthermore, three levels of reliability were introduced from 1 through 3 where 1 used an unreliable agent system, 2 involved an average set of agents and the agents used in 3 were quite reliable. The chart shows that the trust model performed at its best when the system was reliable and the agents were optimistic. In this case, the increase in reliability was approximately 18%. However, in one of the nine cases, the trust model caused the number of positive experiences to drop by 4%, though this was due to the fact that as the agents were pessimists, they were not building any useful relationships.

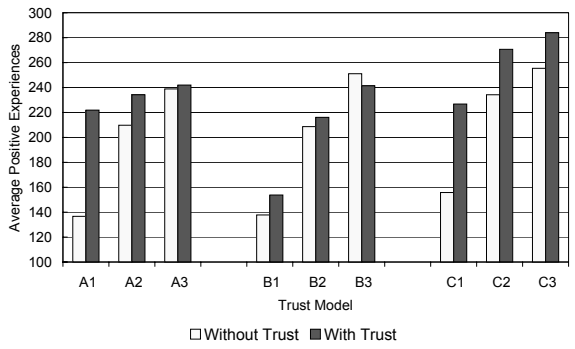

Figure 9: Model Overview

5 Conclusions and Future Work

In this paper, we have presented a trust framework, which uses trust to add reliability and performance to Grid computing. This allows users to dependably utilise those resources with increased chances of favourable usage. The results presented have demonstrated that by using trust models in multi-agent systems in the Grid computing context, a tangible benefit can be achieved by both the users and the resources. Moreover, albeit the number of positive experiences achieved by each user agent was not always significant, the resource agents were able to complete more jobs. Furthermore, the average number of jobs received by each resource agent increased, which increased throughput. Nevertheless, the negative effect of distributing requests, is that the chance of receiving an unreliable request increased and therefore the likelihood of failure, which impacts all other peers. In our current work, we are investigating in further detail how different groups of conditions set in a contract can have differing effects on the agent system when they are not satisfied, and increasing the dynamism for the change in these conditions based on past experience. Finally, the framework is being developed further to allow for integration with the middleware in the High Performance Systems Group at the University of Warwick.

Acknowledgements

This work is sponsored by funding from the EPSRC e-Science Core Programme (contract no. GRS0305801).

References

[1] A. Abdul-Rahman and S. Hailes. Supporting trust in virtual communities. In *Proc. of 33rd Hawaii International Conference on System Sciences*, pages 1769–1777, 2000.

[2] F. Azzedin and M. Maheswaran. Towards trust-aware resource management in grid computing systems. In *Proc. of First IEEE International Workshop on Security and Grid Computing*, pages 452–457, 2002.

[3] I. Foster and C. Kesselman (Editors). *The Grid: Blueprint for a New Computing Infrastructure*. Morgan Kaufmann, San Fransisco, 1999.

[4] I. Foster, C. Kesselman, and S. Tuecke. The anatomy of the grid: Enabling scalable virtual organizations. *International Journal of Supercomputer Applications*, 15(3):200-222, 2001.

[5] N. Griffiths and M. Luck. Coalition formation through motivation and trust. In *Proc. of the Second International Joint Conference on Autonomous Agents and Multiagent Systems 2003 (AAMAS'03)*, pages 17–24, Melbourne, Australia, 2003.

[6] L. Hui, P. Qinke, S. Junyi, and H. Baosheng. A mission-aware behavior trust model for grid computing systems. In *Proc. of International Workshop on Grid and Cooperative Computing 2002 (GCC2002)*, 2002.

[7] S. Marsh. *Formalising Trust as a Computational Concept*. PhD thesis, University of Sterling, 1994.

[8] R. G. Smith. The contract net protocol. *IEEE Transactions on Computers*, 29(12):1104-1113, 1980.

[9] Sun Microsystems Inc. Grid Technology Overview. *http://www.sun.com/software/grid/overview.html*, accessed June 2004.