An agent-oriented, trust-aware approach to improve the QoS in dynamic Grid Federations

Pasquale De Meo¹, Fabrizio Messina²*, Domenico Rosaci³, Giuseppe Sarnè⁴

¹DICAM, University of Messina, 98166 Messina, Italy
²DMI, University of Catania, Viale A. Doria 6, Catania, I-95125 Italy
³DIIES University “Mediterranea” of Reggio Calabria, Loc. Feo di Vito, 89122 Reggio Calabria, Italy
⁴DICEAM University “Mediterranea” of Reggio Calabria, Loc. Feo di Vito, 89122 Reggio Calabria, Italy

SUMMARY

In this paper a distributed approach aimed at improving the Quality of Service in dynamic grid federations is presented. Virtual Organizations are grouped into large-scale Federations on which the original goals and scheduling mechanisms are left unchanged, while Grid nodes can be quickly instructed to join or leave any Grid at any time. Moreover, an agent-oriented framework is designed to observe and characterize past behaviors of nodes in terms of resource sharing and consumption, as well as to determine the trust relationships occurring between each pair of nodes. By combining trust and historical behaviors into a unified convenience measure, software agents are able to evaluate the i) advantages of node’s membership with specific any Grids, and ii) whether the actual set of Grid nodes is able to meet the original goal – in terms of resource sharing and consumption – of the Virtual Organizations. The convenience measure has been exploited to design a fully decentralized, greedy procedure, aimed at controlling the Grid Formation process. Extensive simulations have shown that the coordinated and decentralized process of Grid Formation provides a powerful mean to improve the overall Quality of Service of the Grid Federation.

KEY WORDS: Grid, Grid Federation, Grid Formation, Multi-agent System, Quality of Service, Trust System

1. INTRODUCTION

The paradigm of Grid Computing gained a large popularity in a broad range of application contexts and it is now widely acknowledged as a standard way to efficiently share computational resources [1]. As institutions and companies have joined many ad hoc Grid Virtual Organizations (VOs) [2, 3, 4], a great interest came out by authors to study and propose how these grids can collaborate to share their resources and to coordinate for solving complex tasks.

By means of Grid Federations multiple and heterogeneous grids are coupled together [5, 6, 7, 8, 9]. Research on Grid Federations is now at a mature stage and it catches the interest of more and more large communities of researchers coming from different domains like Climate [10], Science [11] and Finance [12] to only name a few. Moreover, a grid federation can be modeled in various ways, as for instance by means of a fully decentralized approach for resource allocation which does not involve in any broker of coordination [13]. In other words, each organization simply shares its clusters at large scale through peer level coupling.

*Correspondence to: Fabrizio Messina, Dipartimento di Matematica e Informatica, Università di Catania, Viale A. Doria 6, Catania I-95125, Italy. E-mail: messina@dmi.unict.it

Copyright © 2014 John Wiley & Sons, Ltd.
Prepared using cpeauth.cls [Version: 2010/05/13 v3.00]
The original approach for constituting Grid VOs is that of “flexible, secure, coordinated resource sharing among dynamic collections of individuals, institutions” [14] for a common objective, i.e. Virtual Organization are strongly constructed around some specific objectives. Our view is to provide policies, algorithms and needed automation to allow grid nodes to join with or leave automatically any VO of the federation and, similarly, each grid of the federation can decide to accept or refuse the joining request of a node. On the other hand, the grid federation interoperability model can be conceived by leaving unchanged original VO policies, such that they the achievement of the common goals remain central.

The construction of such a very dynamic scenario is really encouraged by the strong development of technologies for utility computing, mainly virtualization [15, 16], by which quick reconfiguration and deploying of virtual nodes with specific middlewares and software is efficiently supported. This approach can allow to meet various demand changes, such that the overall Quality of Service (QoS) provided within the Virtual Organization can remain stable and the original aims and autonomy of the VO remains unchanged. We refer to such a scenario, for convenience, as Open Grid Federation (G).

In the scenario described above, we can state some simple questions: given a grid node \( n_i \) (e.g. a Research Department or a firm division) and a grid \( g_j \in G \), how can we objectively measure the advantage that \( n_i \) would gain when joining a Grid \( g_j \)? Conversely, how can we decide if it is convenient that \( n_i \) leaves a grid it already joined to? More formally, in order to provide an answer to these questions, the problem can be formulated as dynamically assigning nodes to grids with the goal of optimizing the QoS offered by the grids. We will refer to this problem, for convenience, as the Grid Formation.

In dealing with the issue above, i.e. Grid Formation, we consider several aspects. First of all, we take into account the main characteristic of a Grid node not only in terms of resources, but also in terms of requests coming from its own grid users. Indeed, grid users belonging to a particular grid node will exploit its own access pattern, in terms of nature and quantity of requested resources, which depends on the goal which is behind the joining of the node itself into that particular Grid. From this point of view, balancing the supply and offer of resources can help to avoid unsatisfied requests and unallocated resources. Since the nature and quantity of resources each grid node can offer or ask on behalf of its own grid user is highly variable, predicting it is almost impossible. Anyway, observing jobs and service requests submitted by the node (i.e. its users) – in term of amount and nature of involved resources – is always possible, and a behavioral profile for the Grid node can be suitably constructed.

Another concern we take into account is the reliability of grid nodes, since not all the interactions in a grid systems can be satisfactory for all the involved parties. To this end, grid nodes should be able to rate the reliability of other nodes in automatic manner, on behalf of their own grid users. Such a rating should encode how much a node is satisfied by its past interactions with another node which has provided services on the basis of a Service Level Agreement (SLA) specified by its users. Therefore, feedbacks describing the level of QoS provided by the nodes should be properly collected and aggregated to determine the trustworthiness degree of a target node, which should be also weighted by means of the nature (i.e. relevance) of the provided services.

Given the premises above, in this paper we propose a distributed approach to help grid nodes in performing an automatic selection of Grids to join with, and to help VO admin to select the nodes to accept into its own VO. In this approach we take into account past node behaviors and the trustworthiness of a node. In particular, we consider the behavior of each node \( n_i \) in terms of costs of the resources requested/offered and ii) the trust of a node that depends by the relevances and the feedbacks of the provided services, the number of interactions performed and their freshness. The combination of trust and historical behaviors is aimed at introducing the convenience measure, which allows to quantify the advantage, for a given grid node, to join with a certain grid and vice versa. On the basis of this convenience measure, we were able to design a simple and fully decentralized procedure, with greedy characteristics, called Grid Formation (GF) to associate nodes with grids and vice versa. The aim of the GF procedure is to assign nodes to grids and vice versa by...
AN APPROACH TO IMPROVE THE QOS IN DYNAMIC GRID FEDERATIONS

means of a suitable matching problem which makes use of the convenience measure in a distributed fashion.

The trust model discussed in this work, which leads in the definition of a unified trust measure, is specifically designed to take into account reliability and reputation criteria, which are weighted by means of the knowledge that each node acquired in the past about its provider and the freshness of such a knowledge. Furthermore, differently from other trust systems, it is designed to derive some information about the relevance and the level of QoS of the services provided by nodes, such that it is particularly suitable to run in a Grid environment. In such a way, with respect to the provided Grid services, we derive homogeneous evaluations which do not suffer of the possible different opinions or views that each agent can have about both the same service and the same received QoS [17]. The resulting trust system is eventually able to recognize malicious behaviors in a few steps, as shown by the experiments we carried out and illustrated in Section 5. This positively influences the computation of the convenience measures by giving the opportunity to obtain good results from the application of the GF algorithm before in time that by using different trust systems.

The key contributions of this work — which started with a preliminary study already presented in [18] and discussed at the end of Section 6 — are summarized in the following points:

- a computational framework designed to capture the past behavior of grid nodes (i.e. to obtain behavioral measures), on behalf of their users, as well as to compute the trust between a pair of nodes;
- a trust model appositely conceived to work within a grid context and to be coupled with the behavioral measure to obtain a unified measure (i.e. the convenience measure).
- a multi-agent architecture for managing nodes, grids and their interactions, along with the features of the GF algorithm, a fully decentralized procedure designed to exploits the convenience measure in order to re(organize) the composition of each grid of the federation;
- a set of experimental results concerning the simulation of the trust model in a simulated Grid Federation. These have been obtained by considering some critical scenarios, such that the resilience of the trust system to malicious activities – act to trouble trust measures – has been proved.
- a set of experimental results obtained in a simulated Grid Federation of growing complexity – i.e. up to 16,000 nodes assigned to up to 40 grids – which have shown that the GF procedure is able to stabilize the convenience measures of nodes and grids in a few iterations with an increasing accuracy of up to 40% with respect to a merely random assignment of nodes to grids.

The plan of the paper is as follows. In Section 2 we formally describe the grid formation problem in an Open Grid Federation, along with the behavioral and trust measures. In section 3 we discuss a reference multi-agent architecture to support the execution of the GF decentralized procedure, which is discussed into section 4. In Section 5 we present and discuss the experimental results, while in Section 6 we compare our approach with related works. Finally, in Section 7 we draw our conclusions.

2. BEHAVIORAL AND TRUST MEASURES IN OPEN GRID FEDERATIONS

Suppose \( \mathcal{N} \) be the space of \( n \) grid nodes, and \( \mathcal{G} \) the set of Grid VO\s which include the nodes in \( \mathcal{N} \), i.e. the Grid Federation. As stated in the previous section, our Grid Federation model is conceived by leaving unchanged VO structure and policies (e.g. security, resource sharing, scheduling), such that the achievement of the common goals will remain central. Therefore, each computational Grid will provide a set of services according to the objectives and the identity of the underlying VO, maintaining its own autonomy. On the other hand, in order to get the opportunities arising from the connection of several grids, policies, algorithms and needed automation should be provided to
allow grid nodes to automatically join to or leave any VO of the federation and, similarly, each Grid of the Federation can decide to accept or refuse the joining request of a node.

Furthermore, we assume that decisional processes aimed at joining to or leaving a grid are supported, on each node $n_i \in \mathcal{N}$, by a software agent, let be $a_j$. Similarly, we can assume the “reasoning capability” of each federated grid is implemented by another software agent $A_j$ [19].

In this section we present a set of measures aimed at studying $i$) the nodes behaviors, in terms of offered and required resources, and the $ii$) overall grid behavior, which is computed on the basis of the behavior of the nodes forming the VO. In particular, three different measures are defined: $i)$ behavioral measures, $ii$) trust measures and $iii$) convenience measures, as described in the following of this section.

2.1. The Behavioral Measures

Each grid node will represent also the community of its own Grid users submitting their requests for Grid services, e.g. scientists of a research institute which joined a VO to perform a set of simulations. Grid users submit various jobs and service requests to be satisfied by the VO resource broker [20] and eventually by the local resource manager of all the grid nodes of the VO. The set of resources requested by the grid users of a specific node will be referred as “required” resources. Vice versa, offered resources are those shared by the specific node within the VO.

The definition of the behavioral measures relies on the consideration that nodes (i.e., grids) characterized by a significant need of resources (on behalf of their own users, as stated before) are interested in joining to computational grids (look for nodes) having suitable capabilities to satisfy such needs. Therefore we define the Resource Cost associated with the node $k$ (i.e. $RC_k$) as the actual cost of a service with respect to the amount of resources offered/required by $k$:

$$RC = \sum_{i=1}^{k} r_i \cdot c_i$$

where $r_i$ is the $i$-th offered/required resource expressed in resource units (for instance, $MIPS$ for CPU, $TB$ for storage, etc.) and $c_i$ is its associated unitary cost.

We also define the “historical” attitude of the generic node $n_k$ to offer or require resources within a grid by adopting the Node Behavior ($NB$) measure computed as:

$$NB_k^{(t)} = \alpha \cdot NB_k^{(t-1)} + (1 - \alpha) \cdot \frac{RC_{k,req}^{(t)}}{RC_{k,req}^{(t)} + RC_{k,off}^{(t)}}$$

where $RC_{k,off}$ is the cost of offered resources, while $RC_{k,req}$ is the cost of requested resources. The value of $\alpha \in [0, 1] \in \mathbb{R}$ determines the relevance of these two contributions. In particular, the new value of $NB \in [0, 1] \in \mathbb{R}$ at the time $t$ is computed by weighting the previous value (i.e. computed at time $t-1$) by the parameter $\alpha$, and by considering the contribution due to the new service (i.e. computed at time $t$) for which the node has been involved as provider and/or consumer weighted by $(1-\alpha)$.

In detail, the second contribution is calculated between the cost of the involved requested and offered resources, $R_{req}$ and $R_{off}$ respectively. When $RC_{req} \gg RC_{off}$, then the contribution results to be $NB \approx 1$ and it means that its own user tends to ask more resources to other nodes of its own grid. Vice versa $RC_{req} \ll RC_{off}$ means $NB \approx 0$, therefore the node is generally self-sufficient and/or it tends to offer its own resources to the other nodes.

Since we are also interested in the evaluation of the footprint of a grid in offering or consuming resources, we define the Grid Behavior ($GB$) as the average of the behavioral measures $NB_k$ for all the nodes $n_k$ that joined to the specific grid $g_j$. More formally:

$$GB_j^{(t)} = \frac{1}{||g_j||} \sum_{k=1}^{||g_j||} NB_k^{(t)}$$
Clearly $GB_j$ assumes a value in $[0, 1]$ and represents the tendency of the whole grid $g_j$ to offer or require resources.

2.2. The Trust Measures

The second measure we introduce in our framework is based on the concept of trust. In our scope, we can assume a common definition of trust: the quantified belief by a truster with respect to the competence, honesty, security and dependability of a trustee within a specified context [21]. In the model described in this paper, the truster is the generic node $n_p$ (i.e. the associated agent $a_p$), on the behalf of its own grid user, with respect to the generic node $n_r$ (i.e. the associated agent $a_r$). In particular, we exploit the notions of reliability and reputation, which are detailed below.

The reliability is a measure of perceived trust derived by the direct knowledge due to the interactions occurred in the past. Differently, reputation can be assumed as an expected behavior based on information (i.e. indirect knowledge) on the past interactions occurred with other counterparts [22]. In other words, the reputation of a node can be considered as a measure of the trust perceived by the whole community of agents and calculated by exploiting their opinions due to their past interactions occurred with that node. So that when the direct knowledge is insufficient to make credible a direct trust measure of a node then the reputation measures become essential for evaluating trust.

In order to make clear the meaning of interactions in the proposed model, we can state that an interaction between two nodes $n_p$ and $n_r$ is represented by the process by which the resource broker allocate jobs belonging to $n_p$ users into resources of node $n_r$. Software agents acting on behalf of node $n_p$ can observe jobs submitted by the users and query the resource broker on behalf of them to store information from which it is possible to extract reliability indexes. For instance, if a bag of tasks $[23, 24]$ is submitted by grid users of node $n_p$ and allocated also in the node $n_r$, and the running times are not compliance with the SLA, software agent $a_p$ (for node $n_p$) should be able to analyze the context on which failures happened, and eventually compute a reliability index.

Let $\tau_{p,r}$, $\eta_{p,r}$ and $\rho_{p,r}$ be respectively the measures of trust, reliability and reputation that the generic node $n_p$ (i.e. agent $a_p$) computes for the node $n_r$ (i.e. agent $a_r$). The trust measure $\tau_{p,r}$ is obtained by combining the reliability ($\eta_{p,r}$) and the reputation ($\rho_{p,r}$) measures suitably weighted by the real coefficient $\beta_{p,r}$, ranging in $[0, 1] \in \mathbb{R}$, in the form:

$$\tau_{p,r} = \begin{cases} 0.5 & t = 0 \\ \beta_{p,r} \cdot \eta_{p,r} + (1 - \beta_{p,r}) \cdot \rho_{p,r} & t > 0 \end{cases}$$

(4)

Note that for new coming nodes the initial trust (i.e. reputation) is set to 0.5 in order to penalize them for not too much [25] but enough to contrast whitewashing strategies [26].

The coefficient $\beta_{p,r}$ increases according to the number of interactions occurred between the nodes and their freshness, because also the direct knowledge that $n_p$ has of $n_r$ should improve.

More formally, the coefficient $\beta_{p,r}$ is computed as:

$$\beta_{p,r} = \frac{I_{p,r}}{I_{\text{max}}}$$

(5)

where $I_{\text{max}}$ is a system threshold representing the number of interactions after which the “knowledge” that a node has about another node is maximum. Moreover, $I_{p,r}$ is incremented or decremented at each new step as described in Equation 6.

$$I_{p,r}^t = \begin{cases} \max(1, I^{(t-1)} - 1) & \text{if } \Delta T_{p,r} > \Delta T \\ \min(I_{\text{max}}, I^{(t-1)} + 1) & \text{if } \Delta T_{p,r} \leq \Delta T \end{cases}$$

(6)

where $\Delta T$ is a system threshold representing the maximum time interval between the current time-step ($t$) and the time at which the last interaction ($t_{p,r}$) occurred between node $n_r$ and $n_p$, i.e. $\Delta T = t - T_{p,r}$. Therefore, the ratio adopted in computing $\beta_{p,r}$ is to provide a different relevance to the reputation with respect to the reliability based on the experience acquired by $n_p$ about $n_r$ and the “freshness” of such an experience. In other words, the contribute of the reputation in computing trust decreases as much as the number of the interactions occurred between the two
involved nodes constantly increases. Therefore, whenever the last interaction with \( n_r \) is older than \( \Delta T \), the reputation increases in relevance, i.e., the reliability decreases in relevance (first row of Equation 6). Vice versa, whenever the last interaction with \( n_r \) is fresh enough (the last interaction with \( n_r \) is not older than \( \Delta T \)), the relevance of reliability will increase.

A fine tuning of parameters \( \Delta T \) and \( I_{\max} \) will allow to assign \( i \) different relevance of the reliability and the reputation and to increment the \( ii \) resilience to unexpected behaviors changes.

### 2.2.1. Computation of reliability

The reliability measure, \( \eta_{p,r} \in [0, 1] \subset \mathbb{R} \), is computed as:

\[
\eta^{(t)}_{p,r} = \vartheta_{p,r} \cdot \sigma_{p,r} + (1 - \vartheta_{p,r}) \cdot \eta^{(t-1)}_{p,r} \tag{7}
\]

where the parameter \( \vartheta_{p,r} \) weights in a complementary way the contributes due to \( i \) the feedback parameter \( \sigma_{p,r} \in [0, 1] \subset \mathbb{R} \) computed by \( n_p \) with respect to the service \( s_{p,r} \) provided by \( n_r \) at time-step \( t \) and \( ii \) the value of \( \eta_{p,r} \) computed at time-step \( (t-1) \). The parameter \( \vartheta_{p,r} \) is calculated basing on the relevance measure (\( \psi \)) assigned by \( n_p \) to the current service \( s_{p,r} \) and the average of the relevance measures of all services provided by \( n_r \) to \( n_p \) in the past, as follows (Eq. 8):

\[
\vartheta_{p,r} = 0.5 - \frac{\psi_{p,r} - \overline{\psi_{p,r}}}{2} \tag{8}
\]

where \( \psi, \vartheta \in [0, 1] \subset \mathbb{R} \) and \( \overline{\psi_{p,r}} \) is the average value of \( \psi_{p,r} \) computed over all the services to which node \( r \) provided a contribution. In particular, the form of the Eq. 8 has been adopted to limit the phenomenon by which some nodes could acquire positive feedbacks by means of services assuming marginal importance, e.g. simple jobs submitted in that nodes for testing purposes.

### 2.2.2. Computation of Reputation

The reputation measure \( \rho_{p,r} \) is computed by a node \( n_p \) with respect to a given node \( n_r \) as a value ranging in \( [0, 1] \subset \mathbb{R} \). Through the usual meaning of these indexes, 0 means that \( n_r \) is totally unreliable, while 1 means that node \( n_r \) is totally reliable.

Since each node (i.e. agent) has only a partial view of its community, the trust measures computed in its own might differ from those computed by including the opinions of the whole community. In particular, when the node \( n_p \) is interested to calculate the reputation of the node \( n_r \) it can ask an opinion about \( n_r \) to the node (agent) \( n_q \). We assume the agent \( a_q \), associated with \( n_q \), will provide its opinion to \( n_p \) consisting of its own trust measure about \( n_r \) (i.e. \( \tau_{q,r} \)).

Since reputation represents an indirect measure, evaluating the capability of a node of providing reliable opinions about other nodes represents an important task. To this purpose, we introduce the confidence factor \( (\omega_{p,q}) \) to weight such an opinion by taking into account the concordance between the trust value that \( n_p \) and \( n_q \) have about \( n_r \). More formally, \( \omega_{p,q} \) is computed as:

\[
\omega^{(t)}_{p,q} = 1 - \frac{\left| \tau_{p,r} - \tau_{q,r} \right|}{2} \tag{9}
\]

Consequently, the reputation \( \rho_{p,r} \) is computed by \( a_p \) as:

\[
\rho_{p,r} = \frac{\sum_{q=1}^{\left| R_{p,r} \right|} \omega_{p,q} \cdot \tau_{q,r}}{\sum_{q=1}^{\left| R_{p,r} \right|} \omega_{p,q}} \tag{10}
\]

where \( \left| R_{p,r} \right| \) is the cardinality of the set of agents which provided an opinion about \( n_r \).

Looking at Eq. 10, the confidence factor minimizes the impact of untrustworthy opinions by providing more relevance to those mentors that \( a_p \) evaluates as the most similar, like to real life.
2.3. The convenience measures

In order to measure the convenience for a node $n_i$ to join with the grid $g_j$ we define $\gamma_{i,j}$ as follows:

$$\gamma_{i,j} = |NB_i - GB_j| \cdot \frac{\sum_{n_k \in g_j} \tau_{i,k}}{||g_j||}$$

(11)

where $||g_j||$ is the number of nodes affiliate with $g_j$. In words, the convenience $\gamma_{i,j}$ for $n_i$ to join with $g_j$ increases with the difference between the behaviors of $n_i$ and $g_j$, and with the average trust computed by $n_i$ of all the nodes belonging to $g_j$.

Similarly, in order to measure the convenience for a grid $g_j$ to accept the request of a node $n_i$ to join with $g_j$ itself, we define $\eta_{j,i}$ as follows:

$$\eta_{j,i} = |NB_i - GB_j| \cdot \frac{\sum_{n_k \in g_j} \tau_{k,i}}{||g_j||}$$

(12)

The measure $\eta_{j,i}$ takes into account the behaviors of $n_i$ and $g_j$, on the one hand, and the trust computed by all the node $n_k \in g_j$ for $n_i$.

Note that, in general, $\eta_{i,j} \neq \eta_{j,i}$, i.e. the trust computed by the node $n_i$ to the node $n_j$ is not the same of that computed by $n_j$ for $n_i$, therefore Eq. 11 and 12 assume different values for the nodes $n_i$ and $n_j$.

3. THE MULTI-AGENT ARCHITECTURE SUPPORTING OPEN GRID FEDERATIONS

In the proposed model grid nodes and grid VO administrators are supported by an intelligent software agent [19] performing all the activities aimed at (re)organizing the VOs basing on the measures presented in the Section above.

We assume also that a Directory Facilitator (DF) agent is associated with the whole agent framework, and will be able to provide a yellow page service to all the other agents. This service consists of an indexing service of the names of all the node and group agents in the framework that are listed in an internal repository of the DF, by associating with each node and group agent the corresponding name.

The multi-agent architecture described in this section is synthetically represented in Figure 1. In the right part it is depicted a grid federation of 3 VOs. For each of them, a Grid Agent is indicated to manage the global information associated to the grid and to take the decisions (as discussed below in this section), while in the left part we depicted some other nodes. In order to not complicate the draw, the overlapping of VO membership is not shown.

3.1. Agent profiles

Each agent is able to build, manage and update its own profile, i.e. the knowledge needed to again execute its own activities. The profile of an agent $a_i$ associated with a node $n_i$ can be denoted by means of the tuple $P_i = \langle WD_i, RD_i, BD_i \rangle$, where:

- $WD_i$ (Working Data): set of the necessary endpoints to communicate with the DF (i.e. to contact the other agents), to interact with the resource manager of the node $n_i$ and the User Interface which offer the access to the grid to the users.
- $RD_i$ (Resources Data): data concerning all the resources used/offered in the past by the node and the agents which made available additional resources and/or consumed resources of this node.
- $BD_i$ (Behavioral Data): data concerning the behavioral measures of node $n_i$ and grid agents to which the agent has joined.
3.2. The Agent Behaviors.

Node and grid agents execute a set of tasks in coordination, which are briefly summarized here.

**The Node Agent Behavior.** The behaviors of a node agent consists of:

- updating its behavioral measures whenever its own users have consumed resources of other nodes or, vice versa, jobs of users belonging to different nodes have used its own resources;
- updating its trust measures about the other nodes of the federation, basing on the service requests performed by the grid users belonging from its own node;
- periodically updating its own convenience measures, as described in Section 2.2;
- sending behavioral and trust measures to the software agents representing its own grids, whenever it has changed;
- receiving behavioral measures from the grid agents of the Grids it has joined to;
- eventually asking to any grid agent of joining to the associated grid, basing on the convenience measure;
- eventually sending a leaving message to the grid agent of one of the grids it joined to in the past, basing on the convenience measure.

**The Grid Agent Behavior.** On the other hand, the behavior of a Grid Agent is composed by a few steps which are coupled with the correspondent steps of the node agents behaviors. More in particular, it consists of:

- receiving the behavioral and trust measures from the agents of the nodes within its own Grid;
- updating the overall behavioral measure of its own Grid whenever an update for a behavioral measure has been received from a node of its own Grid;
- periodically updating the convenience measure of its own Grid, basing on the behavioral and trust information received from the nodes of its own Grid;
- eventually asking to a node agent of joining to or leaving the VO it represents and manages, due to an updating of the convenience measure with respect to all the grid node of the VO;
• evaluating a joining requests coming from nodes of other grids;
• sending the behavioral measure of the VO to the node agents belonging to its own Grid.

The activities described above, performed by node and grid agents, are formalized in a procedure called GF, i.e. Grid Formation, in Section 4.

4. THE DECENTRALIZED PROCEDURE FOR GRID FORMATION (GF)

In this Section we describe a decentralized procedure named GF (Grid Formation), which is composed by a few activities based on the measures defined in Section 2. The activities of the GF procedure differs for node agents and for grid agents, respectively.

To this purpose, let $T$ be the time elapsed between two consecutive executions of the GF procedure, that we call step. Moreover, we assume that agents can query a distributed database named $GR$ (Grid Repository) on which the list of the grids of the open federation is maintained.

4.1. The GF procedure performed by the node agents.

Let $X_n$ be the set of the grids which the node agent $a_n$ is affiliated to, where $|X_n| \leq N_{MAX}$, and $N_{MAX}$ is the maximum number of grids which a node agent can join with. We suppose that $a_n$ stores into a cache the grid profile $p_g$ of each grid contacted in the past (see Section 3) and the timestamp $d$ of the execution of the GF procedure for that grid. Finally, let $\xi_n$ (a timestamp) and $\chi_n \in [0, 1]$ be thresholds fixed by the agent $a_n$.

The GF procedure performed by the node agent $a_n$ is summarized by the following, ordered steps:

I - A set $Y$ of $N_{max}$ grids from $GR$, so that $X_n \cap Y = \emptyset$ is randomly selected.

II - For each grid $g \in Z = (X_n \cup Y)$ such that $d_g > \xi_n$, a message to the agent $a_g$ asking its profile $p_g$ is sent.

III - For each received $p_g$, the convenience measure $\gamma_{n,g}$ (see Section 2.3) between the profiles of the node agent $a_n$ and the grid $g$ is computed.

IV - The list $L_{good}$, with all the grids $g \in Z$ such that $\gamma_{n,g} > \chi_n$, is updated.

V - A second list $L'_{good}$ is built by inserting a number $k = min(N_{max}, |L_{good}|)$ grids of $L_{good}$ with the greater value of $\gamma_{n,g}$.

VI - For each grid $g \in L'_{good}$, if $g \notin X_n$, a join request to $a_g$ together with the profile $p_n$ is sent.

VII - For each $g \in X$, $g \notin L'_{good}$, then the agent $a_n$ deletes its node $n$ from $g$, i.e. a message to $a_g$ in order to leave the grid $g$ is sent.

4.2. The GF procedure performed on the grid agent side.

Let $K_g$ be the set of the nodes affiliated to the grid $g$, where $|K_g| \leq K_{MAX}$, being $K_{MAX}$ the maximum number of nodes permitted by the administrator of the grid. Suppose that the grid agent $a_g$ stores into its cache the profile $p_u$ of each node $u \in K_g$ and the timestamp $d_u$ of its acquisition. Moreover, let $\omega_g$ (a timestamp) and $\pi_g \in [0, 1] \subset \mathbb{R}$ be thresholds fixed by the agent $a_g$.

The tasks comprised into the GF procedure and performed by the grid agent $a_g$, are triggered whenever a join request by a node agent $a_n$ (along with its profile $p_n$) is received by $a_g$, as follows:

I - For each node $u \in K_g$, such that $d_u > \omega_g$, a message is sent to the node agent $a_u$ to require the profile $p_u$ associated with $u$.

II - The convenience measure $\eta_{g,u}$ (see Section 2.3) for each node $u \in K_g \cup \{n\} –$ where $n$ is the node which has asked to join with the grid $g$ – and the profile of the grid $g$ is computed.
III - The list of good candidates $K_{good}$ storing all the nodes $u$ such that $\eta_{g,u} > \pi_g$ is updated.

IV - A second list $K'_{good}$ stores a number $s = \min(K_{max}, |K_{good}|)$ of nodes belonging to $K_{good}$ having the greater value of $\eta_{g,u}$.

V - For each node $u \in K_g$, if $u \notin K'_{good}$, the grid agent $a_g$ deletes $u$ from $g$ and notifies $u$. Clearly, if $n \in K'_{good}$, its request to join with $g$ is accepted.

5. EXPERIMENTS

In order to test the capability of the model along with the GF algorithm, we performed a set of simulations aimed at testing the capabilities of the trust model and the GF algorithm, which, as stated in Section 4, makes use of the convenience measures described into Section 2. For this aim we conceived ad hoc simulators, which allowed us to collect a set of results which are presented and discussed below.

In particular, experimental results are presented into two parts. The former contains experiments aimed at “stressing” the trust system in order to test its effectiveness in providing a valid unified trust index. The latter discuss the performance of the GF algorithm discussed in Section 4 by simulating the overall proposed framework.

5.1. The Trust model

In order to test the effectiveness of the proposed trust model we carried out some experiments involving a population consisting of 1000 nodes which intensively interact for requiring/providing an opinion. Each node is initially joined with a random number of Virtual organizations comprised between 10 and 20, with a maximum number of nodes per Grid which is 50. Each experiment consisted of 50,000 epochs, where in each epoch 250 nodes (i.e. the 25% of the overall nodes) send Grid jobs which eventually lead in requiring computational resources to other nodes of its own grid. They also ask an opinion to up to 125 nodes chosen in a random way among the set of node belonging to their own grids (i.e. the 12.5% of the entire population). At the end of each experiment, each node used resources of another node about $12 \div 13$ time in average.

As previously described into Section 2, a trust value equal or greater than 0.5 identifies an honest node and, vice versa, a trust value smaller than 0.5 is associated with an untrusted node. When a simulation starts, all the nodes are assumed to be not malicious and trusted, i.e. they received an initial trust value of 0.5. Clearly, the higher the number of trusted and malicious nodes are identified during the simulation, the higher the accuracy of the model.

A variable number of i) malicious nodes (i.e. nodes which provide untruthful opinions about other nodes), and ii) unreliable nodes (nodes providing unreliable services for the jobs allocated into its own resources) has been considered for each experiment, as follows. In particular, we simulated three different scenarios, A, B and C, as described in the table I.

<table>
<thead>
<tr>
<th>Scenario</th>
<th>Untrusted nodes</th>
<th>Malicious nodes</th>
<th>Behavior</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>10% or 50%</td>
<td>NO</td>
<td>Always reliable or always unreliable.</td>
</tr>
<tr>
<td>B</td>
<td>10% or 50%</td>
<td>10% or 50%</td>
<td>Always always reliable and honest or unreliable and malicious.</td>
</tr>
<tr>
<td>C</td>
<td>10% or 50%</td>
<td>10% or 50%</td>
<td>Building positive reputations on services with low relevance for cheating on those with high relevance.</td>
</tr>
</tbody>
</table>

Table I. Scenarios simulated for the trust system

As described in table I, scenario A describes a grid federation on which the 10% or 50% of the nodes assume a untrusted behavior (i.e. they are unreliable in providing services). Scenario labeled
Figure 2. MBE vs epochs that for A) uniform behavior, B) alternate behavior C) misleading opinions in presence of a percentage of the 10% and 50% of malicious nodes.

B extends scenario A because the nodes which assume an unreliable behavior, will assume also a malicious/dishonest behavior in providing recommendations to other nodes. In particular, nodes which assume a malicious behavior will provide always misleading opinions by assigning a trust value which is close to null to reliable nodes, and a trust value of about 1 to untrusted nodes. Finally, scenario C differs from B because malicious nodes assume a more sophisticated behavior, aimed at building a positive reputation by means of services having low relevance. They will cheat this reputation in order to have assigned services have high relevance. Furthermore, in this experiment the ratio of low to high relevant services was fixed to 1:4.

In order to analyze experimental results, we measured the percentage of nodes which correctly recognize untrusted nodes. For sake of simplicity, in the following we will refer to this measure as MBE (Malicious Behavior Estimation).

Figure 2 shows the results of a set of experiments carried out by simulating three different scenarios in terms of average MBE. The curves labeled A10 and A50 describes the trend of the MBE over the various epochs, where A10 refers to the case 10% of nodes assumes an untrusted behavior, and A50 for the 50%, as reported in Table I. Moreover, curves labeled B10 and B50 describes the trend of MBE, when 10% and 50% of nodes have a behavior specified in the scenario B. Finally, the curves labeled C10 and C50 refer to the case 10% and 50% of nodes assume the behavior specified for the scenario C. Finally, we avoided to represent the results produced in absence of malicious nodes because their are practically overlapped with the curve A10.

By analyzing the obtained results, we can note as each experiment puts in evidence a significant resilience with respect to the presence of malicious and unreliable nodes into the Grid. In particular, in presence of a uniform behavior (curves labeled A10 and A50) the 50% (i.e. 90%) of the node population stably recognize all the malicious nodes after about 1500 (i.e. 3500) epochs. If collusive or trouble activities act to mislead the reputation measures are introduced, the obtained results (curves labeled B10 and B50) show an initial very little gap, with respect to the previous case, due to the wrong values of the malicious opinions but it becomes null around the 4,500-th epoch. Indeed, this gap is maximum when the simulation starts (see Sections 2) because i) the confidence factors weighting opinions are usually maximum and ii) the reputation has the maximum impact on the trust measures. Accordingly with the increased knowledge of each node about its environment their confidence factors usually decrease similarly to the relevance of the reputation in computing trust measures. Finally, in presence of dynamic behaviors (curves labeled C10 and C50) the influence of trouble behaviors is clearly evident. However, also in this case the gap in terms of MBE with respect the other experiments is filled up and the 90% of the node population stably recognize all the malicious nodes after about 15,000 epochs.
Figure 3. MBE and number of interaction carried out with the malicious node 103 vs epochs.

In figure 3 we couple two measures: i) the percentage of the nodes which are correctly recognizing the behavior of the malicious node no. 103 (scale for y-axis in the left side), i.e. the value of MBE for that particular node; ii) the number of services assigned to the node 103 over the epochs on behalf of the considered nodes (y-axis in the right side). In particular, any curve labeled $x$ ($x = 1 \ldots 8$) represents the percentage of nodes which “interacted” at most $x$ times with the target node. From these results we can conclude that the number of interactions (i.e. submitted jobs which have been assigned to the malicious/unreliable node 103) needed to the trust model to recognize the malicious nature of the target node is very low, therefore the trust system is effective in the short term.

5.2. The GF Algorithm

The experiments described here are devoted to test the effectiveness of the proposed framework and, in particular, of the GF algorithm presented in Section 4. To this purpose we exploited another simulator, written in JAVA[27].

A first experiment has been conducted on three simulated scenarios consisting of 4000, 8000 and 16000 nodes respectively, and a fixed number of 40 grids. Moreover, 10% of nodes had an untrusted behavior. In simulating such scenarios, all the parameters have been randomly generated by a uniform distribution in their respective domains. Initially, we randomly generated the profile of each node which stores its orientation to consume/off resources and its honest or malicious behavior, the number of grids which each node can be assigned (i.e. $N_{MIN} = 2, N_{MAX} = 10$) and the number of nodes for grid (i.e. $K_{MIN} = 10, K_{MAX} = 80$). These values and those of all the other parameters have been set based on sensitivity analysis we performed.

As a measure of the internal convenience of a grid $g_j$, we introduced the concept of Average Convenience ($AC$) computed on the basis of the convenience measure described by Equation 12 in the Section 2. More in detail, the Average Convenience $AC_j$ is the average of all the measures of convenience $\eta_{j,i}$ computed by the grid $g_j \in G$ for all its affiliated nodes $n_i \in g_j$. Therefore, to measure the global convenience of all the grids belonging to $G$ in our simulated scenario, we computed the mean ($MAC$) and the standard deviation ($DAC$) of all the $AC_j$:

$$MAC = \frac{\sum_{g_j \in G} AC_j}{\|G\|} \quad DAC = \sqrt{\frac{\sum_{g_j \in G} (AC_j - MAC)^2}{\|G\|}}$$

**Average convenience.** The contribute of the GF algorithm in improving the performance of the Grid Federation increases as much as the convenience measures quickly tend to become stable and, as previously specified, it mainly depends on the trust model. Therefore, in order to measure the
advantages introduced by the GF algorithm, we set the simulator to start the execution of the GF algorithm at the 100-th epoch, when the level of stability of the convenience measures is still low due to limited number of performed interactions (each node provided about 15 services† in average), and executed for 20 epochs, because, as will show below, MAC and DAC values reach stable values very quickly.

The initial values for the MAC and DAC measures when the GF algorithm starts to run are summarized into Tables II-a and II-b, along with the relevant parameters:

<table>
<thead>
<tr>
<th>Sc.</th>
<th>Nodes</th>
<th>Grids</th>
<th>MAC</th>
<th>DAC</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>4000</td>
<td>40</td>
<td>0.156</td>
<td>0.0032</td>
</tr>
<tr>
<td>2</td>
<td>8000</td>
<td>40</td>
<td>0.156</td>
<td>0.0033</td>
</tr>
<tr>
<td>3</td>
<td>16000</td>
<td>40</td>
<td>0.155</td>
<td>0.0035</td>
</tr>
</tbody>
</table>

Table II. MAC, DAC and parameters for the 1st experiment (effect of the GF algorithm on the convenience)

![Graph showing MAC and DAC variation vs epochs](image)

The results of the simulations for each of the three scenarios, in terms of MAC (DAC) with respect to the epochs, are shown in the left (right) part of Figures 4-6.

The results clearly show that the GF algorithm introduces a significant increment of the convenience of the grids, that after a period of only 10 epochs achieves a stable configuration in which $MAC = 0.281$ and $DAC = 0.0035$ in the first scenario, $MAC = 0.295$ and $DAC = 0.0036$ in the second scenario and $MAC = 0.305$ and $DAC = 0.0041$ in the third scenario. Therefore, we have obtained an improvement of about a 44% (resp. 47, 49%) in average convenience of the grids in the first (resp. second, third) scenario, while the standard deviation from this compactness value remains small enough. We also observe that the performances of the algorithm improves when the number of nodes increases too.

**Average convenience vs no. of Grids.** In a second experiment, in order to investigate the effect of an increasing number of grids on the algorithm effectiveness, we have considered other three simulated scenarios having a fixed number of 16,000 nodes, and three different numbers of grids, namely 80, 160 and 320, respectively. Also in this case we have simulated a random configurations

---

†Here a service can assume the meaning of a bag of jobs/tasks submitted to the local resource manager by the resource broker, as specified in various parts of the paper.
for the grid. The initial values for the compactness measures are shown into tables III, along with the remaining parameters (shown in the table in the right), which remained unchanged.

The results of the simulations for each of the three scenarios, in terms of $MAC$ ($DAC$) with respect to the epochs, are shown in the left (right) part of Figures 7-9.

<table>
<thead>
<tr>
<th>Sc.</th>
<th>Nodes</th>
<th>Grids</th>
<th>MAC</th>
<th>DAC</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>16000</td>
<td>80</td>
<td>0.158</td>
<td>0.0037</td>
</tr>
<tr>
<td>2</td>
<td>16000</td>
<td>160</td>
<td>0.161</td>
<td>0.0039</td>
</tr>
<tr>
<td>3</td>
<td>16000</td>
<td>320</td>
<td>0.165</td>
<td>0.0040</td>
</tr>
</tbody>
</table>

GF starts at 100 epoch 10 80 2 10

Table III. MAC, DAC and parameters for the 2nd experiment (increasing number of Grids vs convenience)
We remark that the GF algorithm improves its performances in term of convenience when the number of grids increases. In particular, after a period of 10 epochs, it achieves a stable configuration in which $MAC = 0.311$ and $DAC = 0.0043$ in the first scenario, $MAC = 0.327$ and $DAC = 0.0046$ in the second scenario and $MAC = 0.350$ and $DAC = 0.0052$ in the third scenario. As a result, we have obtained an improvement of about a 49% (resp. 51, 43%) in average convenience of the grids in the first (resp. second, third) scenario, while also in this case the standard deviation from this convenience value remains small.

![Figure 7. Experiment 2, 16000 nodes: Variation of MAC (left) and DAC (right) vs epochs in the 1st scenario, with 4000 nodes.](image)

![Figure 8. Experiment 2, 16000 nodes: Variation of MAC (left) and DAC (right) vs epochs in the 2nd scenario, with 8000 nodes.](image)

6. RELATED WORK

Grid federation extend the classical Grid architecture by connecting computational nodes of different Grids. Coupling a great number of computing nodes at large scale enforces to engage
several challenges for coordinating different grid policies, e.g. resource discovery, task scheduling, QoS, fault tolerance and security issues [28]. Therefore, the approach of constituting federated Grids leads to develop a variety of different approaches to perform, e.g., inter-grid scheduling, resource allocation and coordination of grid brokers. In the following we discuss the approaches which, in our knowledge, gave a contribution in the area we have addressed in our work, i.e. resource allocation and trust systems in grid federation, discussing the original aspects of our approach.

Finally, at the end of this section we briefly discuss how this work is partially based on the preliminary study presented in [18].

**Task Scheduling and resource allocation in Grid Federations.** Economical approaches [29] have long been used to allocate computational resources in distributed environments in a scalable and adaptable way [30, 31]. In particular, auctions mechanisms, which are essentially based on the equilibrium between resource demands and supply, have shown their capability in efficiently allocating resources. Prices are established by the market and take into account QoS requirements [32]. Similarly, authors of [13] proposed to model a Grid Federation as an economy driven large scale scheduling system. Indeed, they designed an algorithm able to improve the efficient allocation of jobs on cluster resources across the grid federation and to satisfy the QoS constraints prescribed by the resource consumers. An analogous approach is adopted in [33] where a “computational economy methodology” for a federation of grid agents is discussed. More in detail, a QoS scheduling of distributed clusters of resources in a cooperative fashion is implemented. DRIVE [34] is another proposal for a distributed economic meta-scheduler for a trustworthy allocation of the cost of resources across the members of a Virtual Organization. A major trait of DRIVE is its ability to support a wide range of topologies from small scale local Grids through global Grid Federations. Resources are allocated by outlining the requirements of each service and then negotiated by using a two stage contract. In the first stage the service is allocated and negotiated but the resources availability is not guarantee, while the final contract stage is performed by a further SLA negotiation which fixes all the terms of the services. In this way, the allocation latency is reduced and the negotiation participation is supported.

In [35] Grid Federation is devoted to construct an utility computing infrastructure. The work is based on the adoption of the Globus Toolkit [36] and consolidated standards in order to provide a full meta-scheduling system which shows flexibility and scalability properties. The authors state that the proposed solution exhibits many advantages (security, scalability, etc.) and good performances, particularly in presence of intensive computing applications. In [37] the performances of cooperative, semi-cooperative and non-cooperative grid resource allocation mechanisms, based
on the game theory, are compared and analyzed by exploiting an extended set of experiments. The adopted simulation model has been realized around large Grid sites connected together as a Federation, which accepts job to be scheduled by a main portal giving the access to the Federation.

Authors of [38] focused on scheduling strategies for HTC (High throughput Computing) applications applied to Federated Grids. They classify infrastructures forming a Federated Grid as internal or external, in order to apply scheduling policy which allocates resources internally, i.e. into its own Grid infrastructure. On one hand, their approach is devoted to save time and communication bandwidth by exploiting the location of the internal resources and the membership to their respective resource domains. They present three novel algorithms for inter-grid scheduling in presence of enterprise, partner, or utility Grid infrastructures. Policies presented in their approaches are interesting because they are aimed to limit overload of external resources, such resource sharing between different Grid organizations is facilitated.

Interlinking of Grids through peering arrangements, aimed at enabling Inter-Grid resource sharing, is proposed in [39]. The authors focus on the fact that the resulting large scale infrastructure is potentially able to grow in a sustainable way. For this aim, they also propose a reference architecture, including inter-grid Gateways, and a set of mechanisms, some policies to allow the internetworking of Grids.

In the above contributions, the problem of optimizing the QoS provided by the Grid nodes is central. Summarizing the scope of those approaches, it can be stated that they take into account the result of an utility function, coming or not from an explicit economical model, which leads to assign a value (let be real or virtual) to a strategy or algorithm to schedule tasks and/or assign resources to Grid users, i.e. activities devoted to manage and control the Grid Federation. On the other hand, the main novelty of our approach is that VOs identities and goals are preserved, such that the federated Grids is fully compatible with the original goals of the federated Virtual organizations. Indeed, the presented approach includes mechanisms aimed at improving the global utility offered by each Grid to its users, which constitute the primary goal of every Virtual Organizations [1]. This is performed by defining the problem of Grid formation in the dynamic context of Open Grid Federation. The proposed solution is defined around the concept of “convenience” (convenience measure).

**Trust systems for Grid VOs and Grid Federations.** In proposing our solution to the grid formation problem, we relies on a distributed trust system which, in fact, constitutes a key contribution in the computation of the convenience measure. In this field some past approaches aimed at managing resources and task allocation by exploiting reliability and reputation information, often synthesized in a single unique trust measure [40, 41, 42, 43, 44].

Furthermore, in order to deal with unreliability of computing resources at large scale, decentralized Grid overlays are proposed together with a reputation-based Grid workflow scheduling algorithm in [45]. The proposed algorithm allows the Grid Federation to dynamically adapt itself to changes occurring in resource conditions and in dealing with unsuccessful job execution events or resource failures. The scheduling algorithm is based on global and local information about reputation, which are considered statistical properties, and are obtained by exploiting feedbacks automatically computed on the basis of the results of each performed transaction.

The opportunities provided by trust systems are exploited also in [12], on which a novel framework, called Trust-Incentive resource Management, is designed to take into account the management of grid resources by introducing values of prices, trust incentives, and a weighted voting scheme. Providers set the price according to demand and supply, and consumers maximize the surplus upon budget and deadline, while the weighted voting scheme secures the grid by declining the join requests coming from malicious nodes. In this approach the trust model is very simple, as the trust score is obtained by considering only the number of direct and indirect experiences. Authors present also a set of experimental results confirming improvements in terms of resource allocation efficiency, system completion time, and aggregated resource utilization.

The reputation management system presented in [46] – called “Network of Favors” – is interleaved in a Grid context in order to implement a one-to-one resource sharing credit between the resource providers. In particular, a node A which receives from B a request to use its own
resources, will calculate the reputation of a node B by using the values of “favors” A has received from B and vice versa. Similarly to [45] only the overall success or fail in providing a service is considered. Authors addressed different strategies against to malicious behaviors: i) identity changes are contrasted by considering in the reputation also the past interactions history by differentiating long-known nodes by newcomers and by using cryptography techniques; ii) the work validity is assured by avoiding reply strategies; iii) denial-of-service attacks are warded off by using sandboxes with restricted access to the underlying machine, and no network accesses. Nevertheless, such an approach is similar to that adopted by eBay, therefore it suffers of the same vulnerabilities [47].

In all the approaches discussed above, trust and reputation are exploited in the context of computational grids in order to allow better management and sharing of computational resources (e.g. assigning resources and services and/or monitor specific behaviors). Such approaches implement automatic mechanisms to introduce feedbacks in front of services, and simple strategies which only consider the number of transactions carried out successfully. The trust model we designed for our proposal allows software agents to evaluate relevance and feedback of provided service. In addition, it allows to consider all the aspect involved in the service provisioning, such that the result is a fine-grained evaluation of the resulting QoS.

Similarly, when direct and indirect reliability measures are adopted to construct the trust score, any of these approaches modulate their reciprocal relevance, on the basis of the knowledge hold by a node about another one. Moreover, in computing reputation measures only our proposal provides a different weight to the opinions of the less affordable nodes. Differently by our proposal, in order to contrast malicious activities we may refer to the proposal in [45], which implements some suitable strategies in the trust system. As shown in the experimental section, our trust system is able to effectively contrast the effects due to an high presence of malicious nodes in the system and quickly allows their identification.

As stated in Section 1, the approach described in this work is partially based on a preliminary study presented in [18]. In particular, although the definitions of the behavioral measures are similar, that presented in this work is generic, i.e. it takes into account all the possible cost components. The trust model defined in [18], which is a significant part of the convenience measure, was only a short and preliminary version of that presented in this work. Indeed, it considered only the feedbacks provided by the grid users, while in this study we take into account two distinct measures, reliability and reputation, which are carefully designed and discussed; in the present work, the trust model includes several (counter)measures in order to take into account the freshness of trust information, and to limit malicious behavior of software agents. The experimental section presented in this work is almost new, as an extensive set of results is presented and discussed to validate our approach. The extensive discussion about related work – contents of this section – is original and based on the model and experimental results presented in the previous sections.

7. CONCLUSIONS

In this work we proposed a model to improve the QoS provided by grid nodes in an open, dynamic federation of Grid VOs. We defined the reference context as Open Grid Federation and we dealt with the problem of Grid Formation, by which nodes should be assigned to grids with the goal of optimizing the overall QoS provided within its own grid, is supported by software agents which compute behavioral and trust measures on behalf of grids and nodes. The GF algorithm, designed at this purpose, is aimed at balancing demand and offer of resources in order to avoid unsatisfied requests and unallocated resources. It makes use of a unified measure, i.e. the convenience measure, resulting from two different measures: i) the nodes (past) behaviors, in terms of costs of the resources requested/offered and ii) the trustworthiness of nodes, which is based on data automatically collected by software agents assisting Grid nodes. In particular, trust measures we take into account in our work include a fine grained evaluation of the relevance of the the services mutually provided within each Grid VO. The result is an ad hoc trust model which is suitable to be combined with behavioral measures to quickly obtain reliable convenience measures. At the best of
our knowledge, the presented approach is the only framework which uses a trust system to manage the affiliation of a node with a grid.

In order to measure the effects exploited by the application or our proposals, we performed extensive simulations. As shown in the experimental results, the algorithm is able to quickly and dynamically assigns nodes to grids and vice versa, even in presence of large grids, by solving a suitable matching problem in a distributed fashion.

REFERENCES

29. Sutherland I. A futures market in computer time 1968; .
34. Chard K. Drive: A distributed economic meta-scheduler for the federation of grid and cloud systems 2011; .