Distributed Trust Management in Grid Computing Environments

DISSERTATION

zur
Erlangung des Doktorgrades
der Naturwissenschaften
(Dr. rer. nat.)

dem
Fachbereich Mathematik und Informatik
Philipps-Universität Marburg
vorgelegt von

Elvis Papalilo
aus Tirana

Marburg/Lahn, Januar 2008

Vom Fachbereich Mathematik und Informatik
der Philipps-Universität Marburg
als Dissertation am 15.01.2008 angenommen.

Erstgutachter: Prof. Dr. Bernd Freisleben

Zweitgutachter: Prof. Dr. Bernhard Seeger

Tag der mündlichen Prüfung: am 29.02.2008

To my wife, to my parents, to my brother:
Thank you for standing me by
on the way of being the one I am today!

Acknowledgments

Pursuing a Ph.D. was one of my most important goals. Anyhow, wishing it only is definitively not enough. For succeeding, I had to strive every day for getting a small step further.
During my graduation period, there have been good times and hard times, but I would always consider myself as lucky for encountering many wonderful persons that continuously stayed by my side.

My deepest and most sincere gratitude goes above all to my supervisor, Prof. Dr. Bernd Freisleben, without whom I would have probably never made this important experience in my life. I still have to meet a more diligent, dedicated, competent and open-minded person. Many thanks to Prof. Dr. Bernhard Seeger for his readiness and promptness on reviewing this thesis.

Part of the work was financially supported by DAAD through the "Stability Pact for South-Eastern Europe" project. I would like to acknowledge Prof. Dr. Betim Cico from the Polytechnic University of Tirana and to Dr. Jochen Münch from the University of Siegen for offering me the possibility to be part of this project and for their support.

Many thanks also to the Dean of the Department of Mathematics and Computer Science at Philipps-Universität of Marburg, Prof. Dr. Manfred Sommer for his welcome and for the pleasant "unberechenbare" leisure activities.

I am grateful to Thomas Friese for his support, sharing of ideas and friendship. Many thanks also to Matthew Smith for his careful reviews and constructive discussions.
Particularly, I would like to thank Markus Unterberger, Steffen Hartmann and Achim Fleige for their dedication and competence while helping me out with the implementation of the ideas presented in this thesis.
I acknowledge the members of the Distributed Systems Group for the nice moments we have spent together. For the unconditioned help, I would like to thank the members of the IT group at the Department of Mathematics and Computer Science, especially Oliver Dippel for his sincere friendship.
I am also grateful to Sylvia Pott, Martin and Annette Schneider for the many beautiful moments we have shared together.

Last, but not least, many thanks to my parents, Diana and Pavli, and my brother Erion, for continuously beliving in me and to Pranvera and Misto Margariti for their encouragement.

To my dear wife and best friend, Ela, I express all my gratitude for her precious support and for sharing with me every single moment during these years.

Abstract

Grid computing environments are open distributed systems in which autonomous participants collaborate with each other using specific mechanisms and protocols. In general, the participants have different aims and objectives, can join and leave the Grid environment any time, have different capabilities for offering services, and often do not have sufficient knowledge about their collaboration partners. As a result, it is quite difficult to rely on the outcome of the collaboration process. Furthermore, the overall decision whether to rely at all on a collaboration partner or not may be affected by other non-functional aspects that cannot be generally determined for every possible situation, but should rather be under the control of the user when requesting such a decision.

In this thesis, the idea that trust is the major requirement for enabling collaboration among partners in Grid environments is investigated. The probability for a successful future interaction among partners is considered as closely related to the mutual trust values the partners assign to each other. Thus, the level of trust represents the level of intention of Grid participants to collaborate.

Trust is classified into two categories: identity trust and behavior trust. Identity trust is concerned with verifying the authenticity of an interaction partner, whereas behavior trust deals with the trustworthiness of an interaction partner.
In order to calculate the identity trust, a "small-worlds"-like scheme is proposed.
The overall behavior trust of an interaction partner is built up by considering several factors, such as accuracy or reliability. These factors of behavior trust are continuously tested and verified. In this way, a history of past collaborations that is used for future decisions on further collaborations between collaboration partners is collected. This kind of experience is also shared as recommendations to other participants. An interesting problem analysed is the difficulty of discovering the "real" behavior of an interaction partner from the "observed" behavior. If there are behavioral deviations, then it is not clear under what circumstances the deviating behavior of a partner is going to be tolerated. Issues involved in managing behavior trust of Grid participants are investigated and an approach based on the idea of using statistical methods of quality assurance for identifying the "real" behavior of a participant during an interaction and for "keeping" the behavior of the participants "in-control" is proposed.
Another problem addressed is the security in Grid environments. Grids are designed to provide access and control over enormous remote computational resources, storage devices and scientific instruments. The information exchanged, saved or processed can be quite valuable and thus, a Grid is an attractive target for attacks to extract this information. Here, the

confidentiality of the communication between Grid participants, together with issues related to authorization, integrity, management and non-repudiation are considered. A hybrid message level encryption scheme for securing the communication between Grid participants is proposed. It is based on a combination of two asymmetric cryptographic techniques, a variant of Public Key Infrastructure (PKI) and Certificateless Public Key Cryptography (CL-PKC).

The different methods to trust management are implemented on a simulation infrastructure. The proposed system architecture can be configured to the domain specific trust requirements by the use of several separate trust profiles covering the entire lifecycle of trust establishment and management. Different experiments illustrate further how Grid participants can build, manage and evolve trust between them in order to have a successful collaboration.

Although the approach is basically conceived for Grid environments, it is generic enough to be used for establishing and managing trust in many Grid-like distributed environments.

Kurzfassung

Grid-Umgebungen sind offene verteilte Systeme, in denen autonome Teilnehmer durch die Verwendung von bestimmten Mechanismen und Protokollen miteinander kooperieren. Im Allgemeinen haben die Grid-Teilnehmer unterschiedliche Ziele. Sie treten in die Grid-Umgebung ein und verlassen sie zu unterschiedlichen Zeitpunkten, haben verschiedene Fähigkeiten zur Anbietung von Diensten und besitzen (fast immer) ungenügendes Wissen über ihre Kooperationspartner. Folglich ist es sehr schwer, den Ergebnissen des Kollaborationprozesses zu vertrauen. Weiterhin wird die Entscheidung, einem Partner zu vertrauen, von anderen nichtfunktionalen Aspekten beeinflusst, die sich im Allgemeinen nicht für jede mögliche Situation bestimmen lassen. Solche Aspekte sollten daher dem Benutzer bekannt sein, wenn eine Entscheidung getroffen werden soll.

In dieser Arbeit wird die Idee propagiert, dass Vertrauen eine bedeutende Anforderung für die Ermöglichung von Zusammenarbeit zwischen Kooperationspartnern in Grid-Umgebungen ist. Die Wahrscheinlichkeit, eine erfolgreiche zukünftige Zusammenarbeit unter Partnern zu haben, steht in engem Zusammenhang mit den gegenseitigen Vertrauenswerten, die die Partner einander zuweisen. Somit stellt das Vertrauensniveau der Partner deren Absicht dar, miteinander zu kooperieren.

Vertrauen wird in zwei Arten eingeteilt: Vertrauen in die Identität und Vertrauen in das Verhalten der Kooperationspartner. Das Vertrauen in die Identität bezieht sich auf die Verifikation der Authentizität eines Partners, wohingegen das Vertrauen in das Verhalten seine Glaubwürdigkeit darstellt.
Um das Vertrauen in die Identität zu bestimmen, wird ein Schema vorgeschlagen, das ähnlich dem von Stanley Milgram vorgeschlagenen "Kleine-Welt-Phänomen" ist. Um das Vertrauen in das Verhalten eines Partners zu berechnen, werden unterschiedliche Komponenten des Verhaltens betrachtet, wie etwa Genauigkeit und Zuverlässigkeit. Diese Komponenten werden während einer Kooperation kontinuierlich getestet und verifiziert. Auf diese Weise wird der Verlauf der vergangenen Zusammenarbeit aufgezeichnet und für die zukünftigen Entscheidungen hinsichtlich weiterer Zusammenarbeit zwischen Partnern benutzt. Diese Erfahrungen werden auch anderen Kooperationspartnern mitgeteilt. Ein interessantes Problem ist die Bestimmung des "wirklichen" Verhaltens eines Partners durch das "beobachtete" Verhalten. Es ist nicht klar, unter welchen Umständen und wie lange mögliche Verhaltensabweichungen eines Partners toleriert werden können. Die in der Arbeit vorgeschlagene Nutzung verschiedener statistischer Methoden der Qualitätssicherung bietet die Möglichkeit, das "wirkliche" Verhalten eines Partners zu identifizieren und es ständig unter Kontrolle zu behalten.

Ein weiteres Problem, das adressiert wird, ist die Sicherheit in Grid-Umgebungen. Das Grid wurde entworfen, um verschiedenen Teilnehmern Zugang zu entfernten Rechnern, Speichereinheiten und wissenschaftlichen Instrumenten zu ermöglichen. Die Informationen, die die Partner austauschen, abspeichern oder verarbeiten, könnten für andere Grid-Teilnehmer potentiell wertvoll sein. Somit stellt eine Grid-Umgebung ein attraktives Ziel für Angriffe dar, um diese Informationen zu extrahieren. Diesbezüglich konzentriert sich die Arbeit auf die Vertraulichkeit der Kommunikation zwischen den Partnern. Weiterhin werden Fragen der Autorisierung, der Integrität, des Managements und der Nicht-Abstreitbarkeit behandelt. Das vorgeschlagene Verschlüsselungsverfahren ist eine Kombination von zwei asymmetrischen Verschlüsselungstechniken, einer Variante der Public-Key-Kryptographie und einer Verschlüsselungsmethode, die auf die Benutzung von digitalen Zertifikaten verzichtet.

Die verschiedenen Verfahren zum Management von Vertrauen wurden in einer Grid-Simulationsumgebung implementiert. Die vorgeschlagene Systemarchitektur ist flexibel und kann durch separate Vertrauensprofile an domänenspezifische Vertrauensanforderungen angepasst werden. Durch mehrere Experimente wird illustriert, wie das Vertrauen zwischen Grid-Partnern aufgebaut, entwickelt und verwaltet werden kann, um eine erfolgreiche Zusammenarbeit zu ermöglichen.

Der in der Arbeit vorgestellte Ansatz zum Management von Vertrauen wurde zwar für eine Grid-Umgebung konzipiert, kann aber auch für die Etablierung und Verwaltung von Vertrauen in anderen verteilten Systemen eingesetzt werden.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

"To trust is human"

Ken Grimes

... even in Grid environments!

Grid computing is a form of distributed computing that involves resources across dynamic and geographically dispersed organizations. The parent idea for Grid computing is quite old. In the early 1970s when computers were first linked by networks, the idea of harnessing unused CPU cycles was born [61]. Anyhow, the term "Grid" was established only in the mid-90s to denote a proposed distributed computing infrastructure for advanced science and engineering [140].
According to Foster et al. [141] and [142], the Grid computing paradigm is aimed at:

- providing flexible, secure, coordinated resource sharing among dynamic collections of individuals, institutions and resources, and
- enabling communities to share geographically distributed resources as they pursue common goals, assuming the absence of central location, central control, omniscience, and existing trust relationships.

In general, Grid systems have the characteristics of uncertain environments with casual relationships based on the "good name" of participants at the moment of collaboration. The importance of a "good" or "bad name" is crucial in further decisions about the interactions between them and is influenced by their individual actions performed.
At the base of every decision for an interaction between participants, trust considerations should reside. The level of trust represents the level of intention of parties to collaborate. Trust is usually specified in terms of a relationship between a "*trustor*", the subject that trusts a target participant, and a "*trustee*", the participant that is trusted. The level of trust that the participants build on each other is crucial for constructing the beliefs and determining the level of intention of participants to establish collaboration among them.

These trust values can be accumulated and calculated based on past direct or indirect interactions.
In order to give to the participants, especially to consumers, the possibility to establish such a relationship with the other party, it is better to consider the relationships between trust and Quality of Service (QoS) (Azzedin et al. [82]). QoS properties, according to Rana et

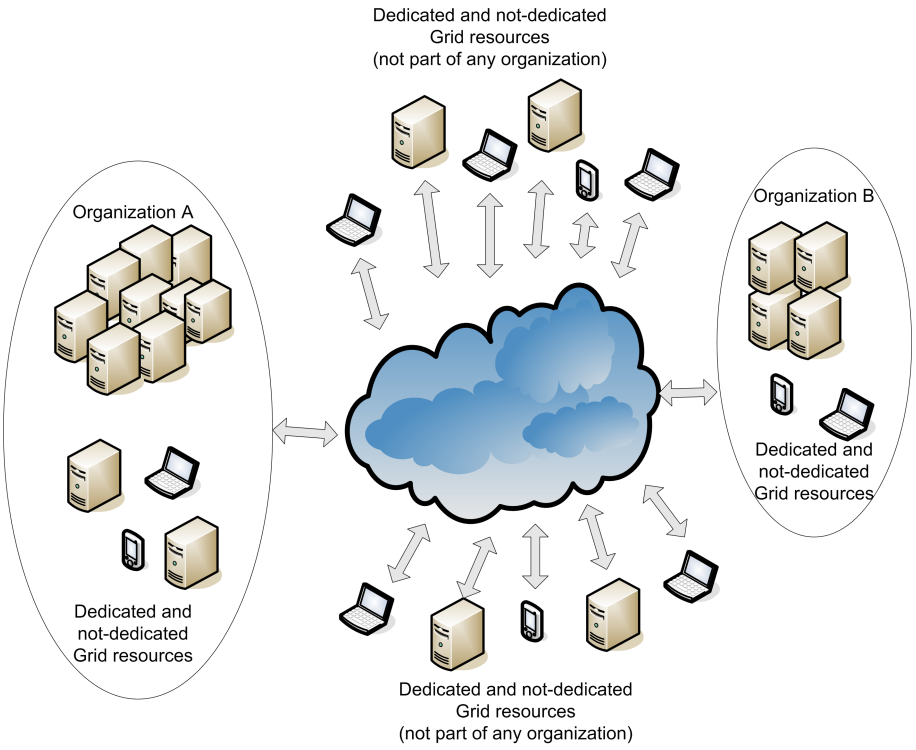


Figure 1.1: Considered Architecture for Grid Environments.

al. [73] can be considered as trust elements and their specific values at a certain moment of time should interfere with the decision of the consumer whether a provider is eligible for a certain activity or not. In a similar way, the possibility to classify consumers that want to take advantage of the offered services should also be given to the providers.
In general, such a trust management system for Grid environments, during establishment, management and evolution of trust between participants, should reflect the same procedure as during establishment, management and evolution of trust among humans. At the end one comes up with the conclusion of Castelfranchi [99] that "there is not such a big difference among feeling threatened by humans that stay next to you or behind a machine somewhere in the network; we must trust for interacting".

1.1 Motivation

With the growing size of the Grid, the number of applications that profit from this technology, varying from simple data sharing, as in Antoniu et al. [77], to expensive simulations [15] grows as well. In such a new situation, comprehensive solutions for handling uncertainties in the environment as a tool for helping individuals to expose themselves, regarding their capabilities, correctness and honesty and at the same time to know better the others in the environment are required.
It is pretty hard to have an objective opinion on the capabilities and the honesty of others in the environment. "Uncertainty" and "ignorance" are difficult to deal with and affect

the decision process when it comes to choose the right partner for/during an interaction. Furthermore, thanks also to the environment, as presented in Fig. 1.1, it is difficult to determine the real identity of the collaboration partners and their real intentions. In other words, a Grid environment deserves the determination made by Dane Skow as an "automated error amplifier" [239] in all its contexts.

1.1.1 Trust and Security in Grid Environments

The mission of Grid computing is to make possible the sharing of services (representing different kinds of knowledge and expertise) distributed across multiple institutions and multiple participants. When it comes to service integration across multiple partners, both security and trust issues become significant.
Security mechanisms tend to provide the needed protection against malicious parties in the environment. Traditional security mechanisms typically protect resources from malicious users by restricting access to only authorized users. Currently Grid security uses X509-based digital certificates [18], security assertions (SAML) [45] or role-based access management (PERMIS [42] and Shibboleth [47]).

Such security mechanisms (Fig. 1.2), although too rigid for authentication and authorization in terms of access control, lack the ability to determine how trustworthy the results obtained from a specific provider are likely to be. In Grid environments, participants need to protect themselves also from others that offer resources or services. As an example, the case when resources act deceitfully offering false or manipulated and misleading information can be mentioned. In such an example, the traditional security mechanisms are unable to offer the needed protection to the threatened participants.

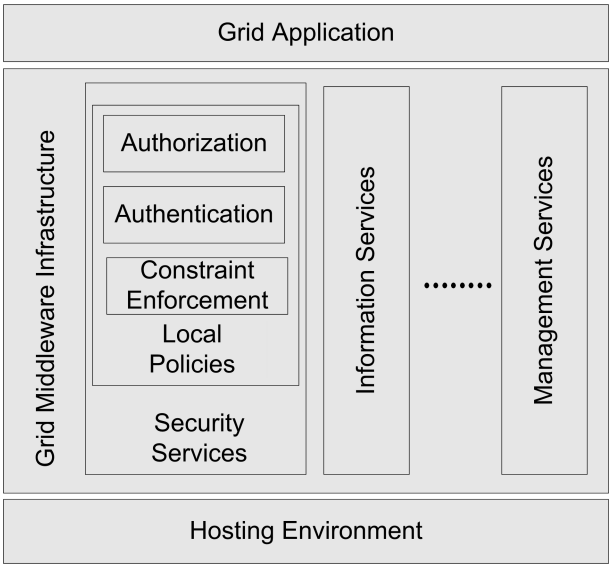


Figure 1.2: Traditional Security Mechanisms on Grids.

Trust management provides the basis to overcome these problems. It is a successful approach that helps to maintain overall credibility level of the system as well as to encourage

honest and cooperative behavior in the environment.
Trust is often linked to the identity of the participants. If the identity is uncertain, then one argues if that participant can still be trusted. However, a framework for identity does not provide information about *trust*. Participants need to somehow evaluate information received from another participant in order to determine the trustworthiness of both the information as well as the sender itself.
As a result, in Grid computing, trust management between participants should also be based on social properties of trust. Just like in Marshs' work [191], as much aspects as possible from sociology and psychology should be included.

1.1.2 Considered Trust Features for Grid Environments

In real life, when dealing in the presence of uncertainty and ignorance with possible threats, one has to rely on trust as the only mechanism for successfully dealing with inherent risks. Being conscious on present threats pushes us to interact on basis of historical evidence assigning tasks accordingly to the different levels of trust we were able to build. From Josang et al. [172] and Foster et al. [142], can be derived that trust systems are seen as most suitable for providing protection against such threats while supporting the scalability and dynamicity of the environment.
Trust in general, is seen as multifaceted and may be related to themes such as experience, optimism, intentions, risk, beliefs, transitivity, competence, reliability, mutual control commitment, etc. While managing trust in Grid environments, these features should be reflected. Socially inspired trust models are useful for Grid environments, especially for the ease of efficiency in supporting the management of trust requirements.
Another important step to be taken in a Grid environment deals with the "decentralization" and "generalization" of the notion of trust. Each of the participants should be able to decide on its own regarding its policies and the preferences set by users (applications). Although in this case more responsibility and expertise is assigned to an entity, at least each of them has a chance to manage their own trust relationships.
Two types of participants can be identified in Grid environments: *clients* and *providers* of services and resources for storage or computational purposes. It is in best interest of all participants that both clients and providers know that they are dealing with single identities in the Grid. Individual actions performed under a particular identity influence the "good name" of the identity itself and as a result the relationships of this identity with others in the community. Each participant can identify itself as a client, as a provider or could also play both roles.

For each of the participants, their *identity* and their *behavior* must be considered to establish trust among them. When trusting a participant, it is important to know which aspect one is referring to. There are instances where a participant is trusted more than the others regarding different levels of trust. There must be the possibility to specify in which aspect of trust participants are interested in and at which level. Trust towards a participant should be handled in different contexts. These contexts should be used to decide whether a participant is eligible for a certain activity or not. The overall value of trust of a participant should interfere with the decision of improving its social position among the others in the environment.
Thus, trust in Grid environments is also a social value that pushes participants to collaborate

with each other, assuming that their identities are already verified and that the kind of offered goods are of sufficient quality.
There is the need to separate different activities in which a participant is involved, and the need to consider not only the behavior of the participant in offering and requesting services or resources but the quality of goods as well. Two participants continue to collaborate or establish a collaboration with each other thanks to direct or indirect good experiences they formerly had. The bigger the level of accumulated trust of a participant, the better will be its "social value" and "position" in the environment. Thus, for each participant its "social position" in the community is important and must be determined upon the calculated level of trust that this it acquired within the community. The importance of a "good" or "bad name" is crucial in further decisions about the interactions with the participants involved.
However, apart from giving the possibility to a participant to gain a better social position in the environment and the possibility to be decorated according to the level of trust, such a system should also include sanctions to participants in case of misconduct or a lower level of participation and interest in the community.
It is also important to underline that just like in our society it is good to have a percentage of risk involved which is derived from some useful social aspects such as reciprocity and altruism. In general, Grid systems can be seen as uncertain environments with casual relationships based on the "good name" of entities at the moment of collaboration.

1.1.3 Acting on Trust

The overall decision whether to trust an interaction partner or not may be affected by other non-functional aspects that cannot be generally determined for every possible situation, but should rather be under the control of the user when requesting such a decision. In addition, while the basic functionalities of two applications could be similar, differences in application behavior could be caused by different domain specific trust requirements. Therefore, a trust system for a Grid environment should *offer flexible and easy to use components that can be configured to the specific needs of a user/provider on a per application/services requestor basis* (Fig. 1.3).

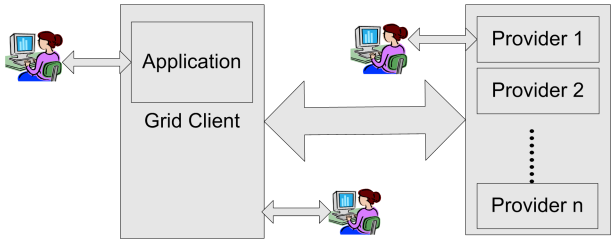


Figure 1.3: User/Application−Client−Provider.

1.1.4 To Trust or to Control?

In Grid environments, a participant needs to safeguard itself against others' "betrayal" of trust and eventual sub optimal performance and behavior. It may pursue not only to build

trust but at the same time to use control mechanisms. Trust and control mechanisms should stay in a supplementary relationship (Fig. 1.4). Control mechanisms do not "kill" trust but help to protect participants in the environment from betrayal of trust. A high level of trust should in no case mean a low reliance on control mechanisms and thus reducing it to zero.

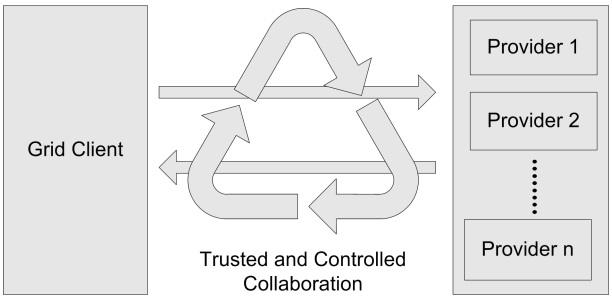


Figure 1.4: To Trust and to Control.

1.2 Contributions of the Thesis

This work offers a flexible model for establishing and managing trust in Grids as well as an analysis of the effects that trust in general and different requirements for trust can have on the environment and on personal views about the competence and honesty of others.

1.2.1 Trust Model

Consideration of Social Aspects of Trust in Grid Environments. Considering the human experience while establishing and managing trust among participants in the environment as one of the "starting points" of the thesis. A wide variety of needs and requirements for trust from the users of applications running on Grids, regarding possible collaboration partners, are considered. Trust is established and managed based on experience (personal or that of the others), beliefs and prejudice. Each of the participants is able to manage its own trust values regarding the others in the environment and also offers this experience when it is requested.

Trust Threats Analysis. Threats to trust in Grids are taken into consideration and analyzed during this work for having a better understanding on possible deceitful behavior of the participants in the environment.

Hybrid Trust Model. This thesis considers both identity and behavior trust of the interaction partners together with different sources to calculate the overall trust values regarding that specific interaction partner.

User/Application in the Loop. Trust establishment and management are adapted to user and application requirements. The trust system is configured to the domain specific trust requirements by the use of several separate trust profiles covering the entire lifecycle of trust establishment and management.

Relationship between Trust and Control. Each of the participants continuously controls the behavior of their counterparts during the entire collaboration. The following principles are applied:

- high control level on nominal trust and vice versa and
- user/application can express their preferences on the control level depending or independent of the trust level.

Detection and Prevention of Anomalous Behavior. A definition of the behavior of participants in Grid environments, as it could be derived from social sciences, is given. Statistical methods of quality assurance are used for monitoring the behavior of participants during long and short term collaborations as a tool for detecting possible deceitful behavior and trust betrayal.

1.2.2 Analysis of Effects of Trust in Grid Environments

Analysis of System Performance while Involving Trust. It is not possible to achieve a higher level of reliance on the others' behavior for nothing. The effects that trust can have on important features like system overhead, mean processing time, etc are analyzed through different simulated scenarios.

Effects of Control Levels on Trust. Many situations of normal and anomalous behavior are taken as use cases and analyzed. Different control levels (from strong to moderated control) exerted from participants and different typologies of interactions between them are configured.

1.3 Thesis Overview

The rest of the thesis is organized as follows:

Chapter 2. Here, an overview of Grid environments is presented. Current technologies applied together with security vulnerabilities are analyzed. The aim of this chapter, except from being an introduction into Grid systems, is also to show the motivation for considering trust and in part also security as research topics for this thesis.

Chapter 3. This chapter discusses the most important trust features for Grid environments together with different threats to trust that were identified during this work. The aim of this chapter is to define the notion of trust considered for designing the trust model. Part of this work was published in the Proceedings of the First International Conference on Grid Services Engineering and Management (GSEM'04), Erfurt, Germany [212].

Chapter 4. In this chapter, a discussion of the literature related to trust modeling in the Grid and in Grid-like domains, is provided.

Chapter 5. This chapter outlines the key concepts of the approach proposed to achive trust in Grid computing environments. A flexible model for managing both identity and behavior trust of Grid participants is presented. Different sources for gathering trust information are used and an important space to configure the requirements is left to the human user. The collaboration among participants is continuously monitored according to the user/application needs and accumulated experience during the past collaborations. Furthermore, statistical methods for quality assurance are used for categorizing the behavior of partners according to the observations during the current collaboration and observations done in the past, trying to find any possible deviation on their behavior. The chapter concludes with a proposal on how to enhance the security of the communication among trusted partners during future collaborations. The ideas presented in this chapter have been published in "Siegener Periodicum zur Internationalen Empirischen Literaturwissenschaft'03" [135], the Proceedings of the Fourth International Conference on Grid and Cooperative Computing (GCC'05), Beijing, China [216] and the Proceedings of the International Conference on Grid Computing, High-Performance and Distributed Applications (GADA'07), Vilamoura, Algarve, Portugal [213].

Chapter 6. This chapter presents the experimental work done in simulation environments during this thesis for:

- observing how trust evolves among Grid participants,
- what are the effects of trust in the performance of the system in general,
- what are the effects of trust on the collaboration among Grid participants,
- what are the effects of control on trust itself and on the performance of the system,
- what "success rate" has the deviating behavior of the counterparts on different control levels (model configuration) of Grid participants.

Part of the experimental results have been published in the Proceedings of the Third International Symposium on Information Assurance and Security (IAS07), Manchester, United Kingdom (best paper award) [214] and in the International Journal of Information Assurance and Security (JIAS) [215]

Chapter 7. This chapter closes the thesis giving a summary of the conclusions and the work done in this thesis. Finally, some areas of future research are outlined.

Appendices. The appendices, except of showing the details regarding the experimental scenarios, contain a discussion on why simulators were preferred upon real-life platforms for running the evaluation experiments. Furthermore, a detailed presentation of the used simulator toolkit together with the implemented changes for adapting the trust model in this simulation environment is given.

Chapter 2

Can You Trust Your Partners in the Grid?

"It's good to trust; it's better not to"

Italian proverb

2.1 Introduction

2.1.1 Reasons for Grids

Grid computing is seen as the upcoming technology for solving complex computational problems. The systems linked in the Grid, forming the virtual computational space, might be distributed in different geographical areas, run different operating systems and owned by different organizations, each with its own policies for the management of resources.

In the form we know it today, Grid computing provides a framework for exploiting the underutilized resources of different organizations (Fig. 2.1) and thus offers the possibility of substantially increasing the efficiency of *resource usage*. Through efficiently using the resources, Grid computing offers also a *cost efficient problem solving environment*.

2.1.2 Application Areas

In the following, some application areas that could profit from Grid technology will be presented. These applications are also used later on in this thesis, for deriving special trust requirements and needs that each of the applications or users of these applications can have towards their possible collaborating parties.

Data Sharing. The Grid can be considered as a large distributed data server. It is made up of a pool of servers that basically run the same application, e.g. GridFTP [26]. Each instance (session) runs independently without interfering with other instances.
The client is responsible for initiating the sessions, aggregating the results and putting together the file chunks to compose the whole file. This is a Grid application that can take advantage of the storage capacity of idle servers and available network bandwidth.

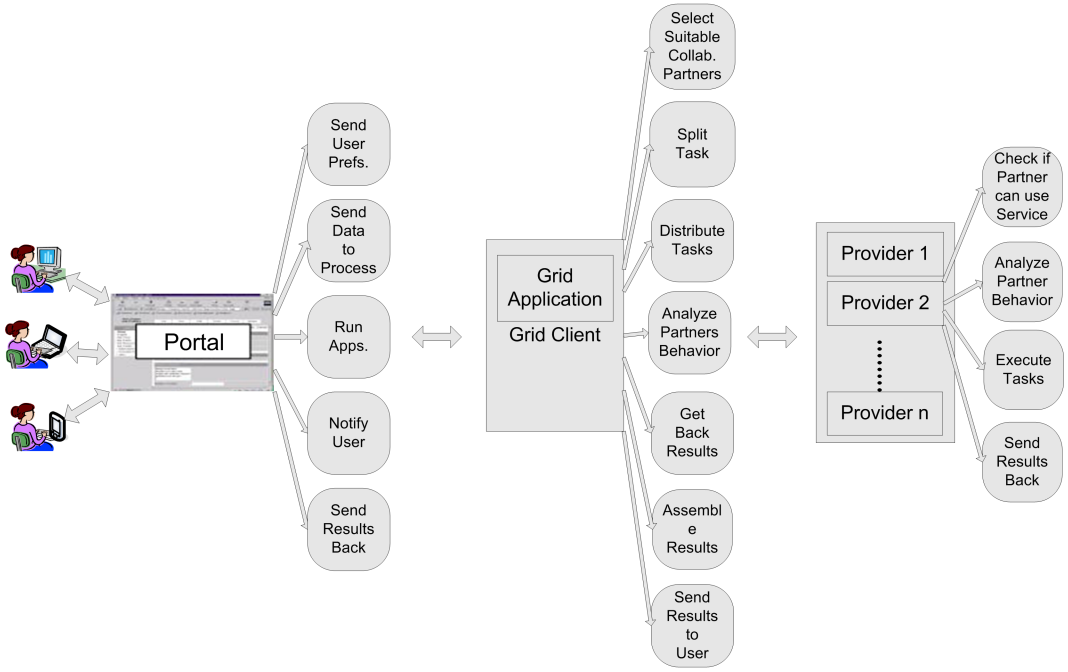


Figure 2.1: Generalization of Application Scenarios.

Media Sciences (Ewerth et al. [134]). Video content analysis is a complex task that could also profit from the use of specialized Grid applications in this field. One example could be the process of identifying "cuts" in videos. Through a temporal segmentation, the video is divided into disjoint segments called "shots". Two shots can be separated by transitional frames, but often there is an abrupt shot change (called a "cut") without such frames. To identify such "cuts", the video file is splitt and all these input video files are sent to a scheduler, which assigns the tasks (splitt video files) to the remote services that will take care of the analysis and the identification of the "cuts". After all the jobs are processed, the frame dissimilarities have to be merged together, the user is going to be notified that the work has been finished and the computed cuts list can be downloaded.

Medical Image and Video Analysis (Amendolia et al. [75]). The scenario tackles the problem of storing and processing large images, which typically requires large computational infrastructures such as distributed databases and clusters. The solution allows the use of idle storage and processing capacities in machines of the Grid to store and process large amounts of medical digital images. An example from medical imaging is finding similar cases (images) in a set of mammograms for diagnosis support. Images resulting from the mammography are passed to remote services representing image analysis algorithms which calculate similarities in image sets. These image analysis algorithms may be basically similar to those used in the identification of video cuts or in object recognition tasks within frames of video sequences. However, a more qualitative output is required in the medical scenario

compared to the video application.

Scientific Simulations (IBM Red Paper Series [137]). In this scenario, Grid computing is used to support the execution of complex system simulations in the areas of physics, chemistry, and biology. The implementation tackles the problem of intensive calculations, which demands high performance computing and typically requires large computational infrastructures such as clusters. Conventional high-performance facilities include computer clusters, super-computers and a wide range of dedicated computing devices that provide good computing power but at a high cost. Apart from the acquisition costs, such devices usually have high maintenance costs and tend to become obsolete within a rather short period of time (a few years at most), having to be replaced by new ones as the research evolves.
Summing up these ideas, research activities that perform compute intensive calculations depend on high performance computing and, as a result, on very expensive infrastructures. Such infrastructures must be also cost-effective and long-lasting. Therefore, this is the perfect scenario for Grid computing technologies.

Consulting Industry (IBM Red Paper Series [155]). This scenario describes a Grid solution for a consulting company that provides financial management services to its clients via the Internet. The application evaluates investments according to client constructs and policies defined by corporate and government rules. A significant amount of its processing time is calculation-intensive, because the application evaluates all available investment options of a particular client. Depending on the type of request, the processing can consume a large amount of CPU resources.

2.2 Grid Environments

Grid computing is a relatively new paradigm that materializes the old concept of using multiple distributed resources to cooperatively work on a single application (Schopf [232]).

2.2.1 Definitions

From its first appearance as a new paradigm, many definitions about Grid computing were given. In 1998 [140], Foster and Kesselman wrote:
"A computational Grid is a hardware and software infrastructure that provides dependable, consistent, pervasive and inexpensive access to high-end computational capabilities."
In this definition, some of the Grid properties and functionalities are also given.
Some years later, this definition also includes "the direct on demand access to computers, software data and other resources" (Foster et al. [141]).
Again, according to Foster [139], the Grid computing paradigm is aimed at:

- coordinating resources that are not subject to centralized control - The Grid aims at the integration and coordination of resources and users that live within different control domains, different administration units, different companies and addresses the issues of security, policy, payment, membership, and so forth.

- using standard, open, general-purpose protocols and interfaces - The Grid is built from

multi-purpose protocols and interfaces that address such fundamental issues as authentication, authorization, resource discovery and resource access.

- delivering nontrivial qualities of service - The Grid allows its constituent resources to be used in a coordinated fashion to deliver various qualities of service, relating for example to response time, throughput, availability, and security, and/or co-allocation of multiple resource types to meet complex user demands, so that the utility of the combined system is significantly greater than that of the sum of its parts.

All these definitions do not specify what a Grid is, but give an explanation of what a Grid should be able to do. The most convincing definition of what a Grid is comes from [22] where Grid computing is described as:
"an emerging computing model that provides the ability to perform higher throughput computing by taking advantage of many networked computers to model a virtual computer architecture that is able to distribute process execution across a parallel infrastructure."

In this work, the term Grid is used in the widest sense to describe the ability to pool and share resources in a global environment in a manner which achieves seamless, transparent, simple access to a vast collection of many different types of hardware and software resources, through non-dedicated wide area networks, to deliver customized resources to specific applications.

2.2.2 Grid Participants

To understand the Grid, one needs to understand who uses it and to which purpose it is used. According to Foster et al. [140], the classes of users that will exploit Grid capabilities together with their interests are presented on the following table:

Class	Purpose	Uses	Concerns
End users	Solve problems	Applications	Transparency; Performance
Application developers	Desktop applications	Programming models; Tools	Ease of use; Performance
Tool developers	Develop tools; Programming models	Grid services	Adaptivity; Exposure of performance; Security
Grid developers	Provide basic Grid services	Local system services	Local simplicity; Connectivity; Security
System administrators	Manage Grid resources	Management tools	Balancing local and global concerns

Table 2.1: Participants in Grids as of Foster et al. in [140].

The basic services in Grids are implemented by "Grid developers", which operate most of the time on different hardware configurations rather than on software or protocols. All

programming models are brought by "tool developers", which deal with libraries and services used later on by "application developers". "Application developers" construct high level applications and components providing the "end users" with the right infrastructure to accomplish their needs in Grids without being concerned with the fact that they are functioning in a Grid environment. The final class of users are "system administrators" which are in charge of infrastructure management assuring an effective functional behavior, at any time in the environment.

In this work, only three classes of Grid users/participants are considered, "end users", "Grid consumers" and "Grid providers" (Fig. 1.3 and 1.4). The above table could be summarized in table 2.2.

Class	Purpose	Uses	Concerns
End users	Solve problems	Grid-enabled applications	Transparency; Performance; Security; Trust
Grid consumers	Execute end users' requests	Scheduling tools; Management tools; Grid and local system services	Performance; Costs; Connectivity Security; Trust
Grid providers	Maximize returns on offered services	Management tools; Grid and local system services	Transparency; Security; Trust

Table 2.2: Participants in Grids as used in this work.

2.2.3 Grid Environment Architecture

Grid computing was born to provide users with a seamless computing environment [140] and is not only a matter of hardware connection, audience of participants or communication among them. There do exist a lot more underlying principles that form a Grid environment.

- *Multiple administrative domains and policies.* Grid participants are geographically distributed across multiple administrative domains and owned by different organizations each with its own management and usage policies.
- *Heterogeneity.* A Grid involves a variety of resources heterogeneous in nature.
- *Scalability.* The Grid has the ability to grow from a few integrated single resources or organizations to hundreds and thousands of them.

Among communities that take part in a Grid environment, it is distinguished between dedicated and sporadic participants. Each of them, alone or combined, helps to form two kinds of Grid environments:

- Dedicated Grids and
- On-demand Grids

In the following, a description of their characteristics is given.

Dedicated Grids. Concerns the case when large vendors will offer dedicated proprietary applications Grids. This kind of environment is mainly compound of static Virtual Organizations (VOs), where the participants have agreed on conditions and rules for sharing resources and using services (Fig. 2.2). VOs enable different organizations or individuals to share resources in a controlled fashion to achieve a common goal and as a result, they are emerging as fundamental entities in modern computing (Foster et al. [142]). The policies for authentication and authorization vary according to the organizations involved.

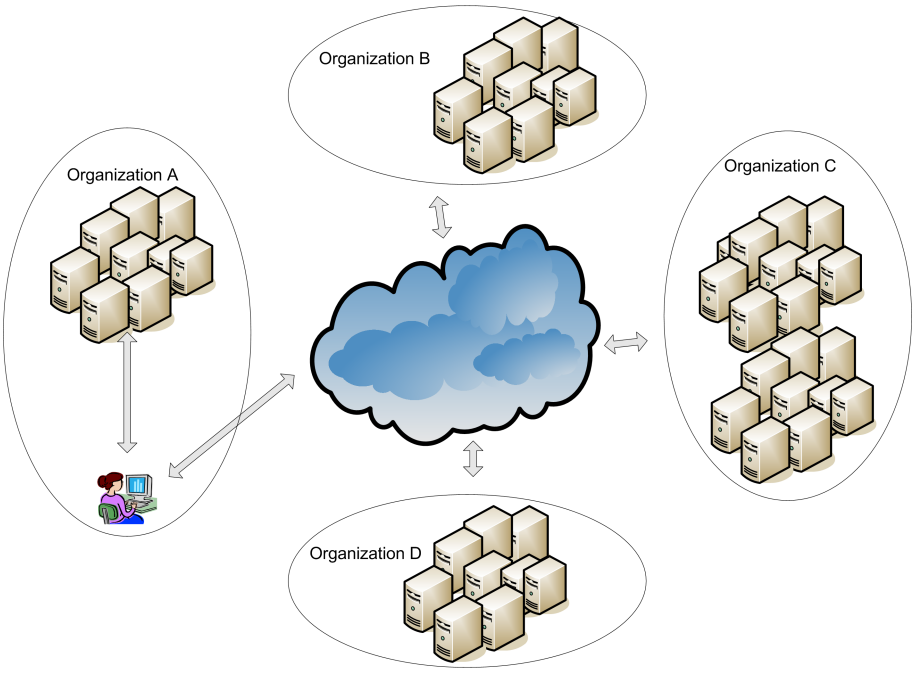


Figure 2.2: Dedicated Grids - Environment Architecture.

All participants are confident that applications that run on the remote sites, either on behalf of consumers or providers, are going to behave properly. Based on the confidence provided by this Virtual Community, members of a Dedicated Grid should, most probably, not experience the problem of an unpleasant behavior of the collaboration parties. On the other hand, Grid members are limited to those interested in the subject. The creation of such dedicated environments allows the deployment of a system that has been optimized for the intended purpose. Hence, it opens the possibility of specialized Grid environments. It means, a particular type of Grid, handled by an organization, whose aim is to provide dedicated and specialized Grid services, will be created. The audience is thus narrower with Dedicated Grids than with other types of Grids. This type of environment does not fulfill the Grid mission at its very best. Seamless access of the users is not a reality and a centralized control of the behavior of participants and their collaboration is done.

On-Demand and Interoperable Grids. This environment could also be considered as the next step of the evolution of Grid computing (Schreck et al. [233]). In this environment, a wide range of participants, with no real common interests, exist (Fig. 2.3). The Grid offers to them a seamless opportunity to share and use resources and services present in the environment.

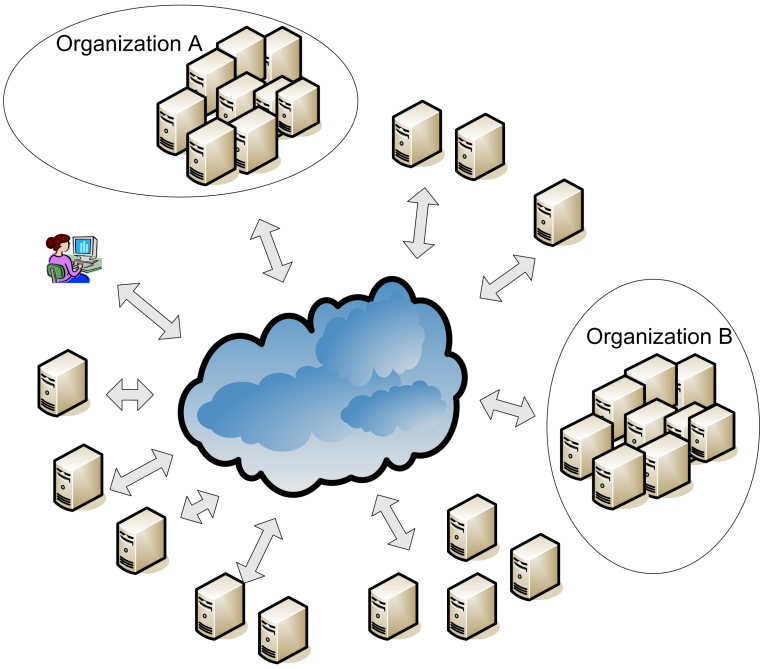


Figure 2.3: On-Demand and Interoperable Grids - Environment Architecture.

The environment is considered to be compound of dynamic VOs. Participants can organize themselves, "on the fly", into a group in order to provide functionality and behavior that none of them individually can. Any new VO can be made available and offer its functionalities to every other participant in the environment. The dynamic organization is the key behind higher robustness and lower costs for the management of Grid systems. It is possible to perform a computation, solve a problem, or provide service(s) to a single consumer or to thousands of them. This Grid environment may consist of millions of interconnected participants. Each of them stays behind a Grid node which forms an abstraction over resources. Consumers could profit from new services, functions or even new concepts offered by providers.
The dynamicity of the environment will facilitate the deployment of new services, but at the same time raises many problems, from performance degradation to the growing of the uncertainty in the environment as the number of participants grows. In such environments *failures are the rule rather than the exception.*

None of the participants is aware of the nature of its collaboration parties and some unpleasant behavior from certain participants is expected. As a result, policies for dynamic behavior control should exist. In order to offer satisfactory solutions in such an unreliable environment, providers must also offer explicit QoS assurances regarding availability, stability, and capability [71]. The possibility to share or use resources and services should be given only to the most trusted and competent participants. This will also serve to push participants to continuously improve their behavior and the QoS of the services they offer.

In the rest of this thesis, it is refered to this kind of environment during the analysis of the different needs for trust while deploying the trust establishment and management model.

2.2.4 Grid Security Models

Security is a crucial aspect in Grids. There are many security aspects within Grids that play an important role. Among those are authentication, access control and privacy, encryption and confidentiality, integrity and non-repudiation.

- Authentication: attests the identity of collaboration parties. The authentication mechanism may be a custom authentication mechanism or an industry-standard technology.
- Access Control and Privacy: deals with the question "*which* participant can perform *what* action, *when* and *where*". The problem of access control comes up after the authentication was done successfully. Privacy deals with single preferences from participants to hide and protect data from unauthorized requests.
- Encryption and Confidentiality: constitute an additional mechanism for protecting the communication between collaboration parties and at the same time prevent the disclosure of sensitive information.
- Integrity: protects the data saved or exchanged between participants from being altered or destroyed.
- Non-repudiation: prevents participants from denying the origin of their actions.

Every participant in the Grid environment wants assurances that their data is kept unchanged, that control and privacy of the communication are assured, that his/her requests are executed accordingly, etc. In order to define a secure environment, the above aspects have to be taken into consideration together with some other additional constraints that will be treated in this thesis, like:

- Certification Infrastructure: determining the identity of participants. In such an infrastructure some of the "certification authorities" should be considered more reliable that the others.
- No Cheap Pseudonyms: preventing the participants to easily change and thus hide their real identity.
- Collaboration Monitoring: the goal is to monitor the progress of the collaboration between participants. Even the definition of policies regarding the collaboration does not guarantee a secure collaboration; therefore monitoring plays a crucial role. This

is essential to ensure that the interaction is progressing according to the needs and preferences of the collaborating parties leaving no spaces for surprises regarding the outcome of this interaction. Monitoring is important to achieve enough knowledge on the behavior of the collaboration parties in order to find out if any form of deviating behavior occurs.

- Past Behavior: archiving the monitoring results and thus achieving knowledge on the behavior of others in the environment.
- Identity and Behavior: consideration of both identity and the behavior of a single participant in the environment.
- Secure Execution: deals with running trusted (not-trusted) applications on not-trusted (trusted) Grid environments.
- Trust Notions: many of the aspects mentioned above are subsumed by trust notions. Trust can be considered as an enhancement of the security for Grid environments. Trust relationships between users, administrators, applications, services and every other Grid participant have to be considered.

The underlying security mechanisms should offer support for these trust relationships.

In the following, the security models of some of the most important Grid initiatives are presented:

- **Globus [19]:** The Globus Toolkit uses the Grid Security Infrastructure (GSI) for enabling secure authentication and communication over an open (Fig. 2.4). GSI provides a number of services for Grids, including mutual authentication, single sign-on and delegation of credentials for computations that involve multiple resources and/or sites. GSI is based on public key encryption, X.509 certificates and the Secure Sockets Layer (SSL) communication protocol. To these standards, some extensions have been added for single sign-on and delegation.
  Authentication: GSI uses certificates for authentication. A participant in the Grid is identified by a certificate, which contains information vital to identifying and authenticating the participant. A certificate contains:
  - The name of the subject. It serves to identify the participant represented through this certificate.
  - The public key belonging to the subject.
  - The identity of a Certificate Authority (CA) that has signed the certificate to certify that the public key and the identity both belong to the subject.
  - The digital signature of the named CA.

  If two parties have certificates and if both parties trust each other's certificates, then the two parties can prove to each other that they are who they say they are. This is known as mutual authentication.
  Certification Infrastructure: A third party is used to certify the link between the public key and the subject in the certificate. In order to trust the certificate and its

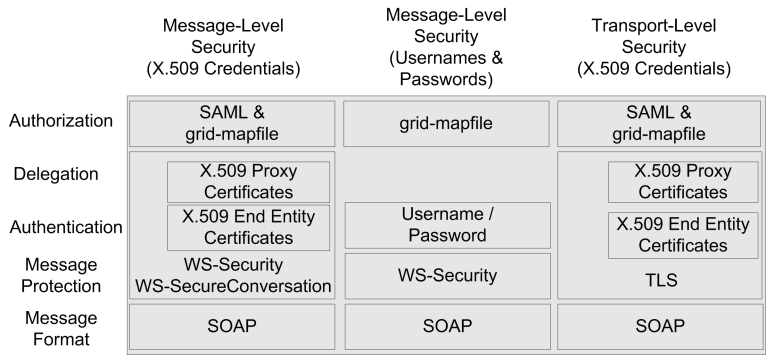


Figure 2.4: GT4 Grid Security Infrastructure.

contents, the CA itself has to be a trusted one. Furthermore, the participants themselves can generate certificates for temporary sessions (proxy certificates) if several Grid resources are needed to be used and each of them requires mutual authentication.
Encryption and Confidentiality: By default, the GSI does not establish confidential (encrypted) communication between parties.
Access Control and Privacy: GSI has an access control list. The purpose of this file is to map a GSI Credential to a local user's login name. The GSI administrator on the site can map the holder of any GSI credential to any local user name. It is up to the GSI administrator to verify that the GSI Identity is owned by and matches the local username. The access control list is a plain text file, containing a quoted GSI Credential Name (the subject of an X509 certificate) and an unquoted local user name. Each subject name in the access control list must be listed only once. However, multiple identities may map to a shared local name. It is up to the GSI administrator to ensure that the access control entries do not violate any site security policies.
Integrity: By default, the GSI provides also communication integrity. It is a configurable feature and if activated, some overhead is introduced in the system.
Trust: the only trust notion contained in GSI is represented by the "trusted CAs list", where all CAs that a participant trusts are listed.

- **Legion [33]:** In Legion, existing security standards such as Kerberos and the Secure Sockets Layer (SSL) communication protocol are integrated into the metacomputing environment. The basic concepts of the Legion Security Model (Fig. 2.5) are minimal:
  - every object (where the object may represent a file, a Legion service, a device, or any other resource) provides certain member functions (authentication, access control, delegation).
  - user-defined objects can play two security-related roles - those of the "responsible agent", RA, and the "security agent", SA.
  - every invocation of a member function is performed in an environment consisting of a triple of (unique) object names - those of the operative responsible agent, security agent, and "calling agent".

– there is a small set of rules for actions that Legion will take, primarily at member function invocation. The general approach is that Legion will invoke the known member functions, thus giving objects the responsibility of defining and ensuring the policy.

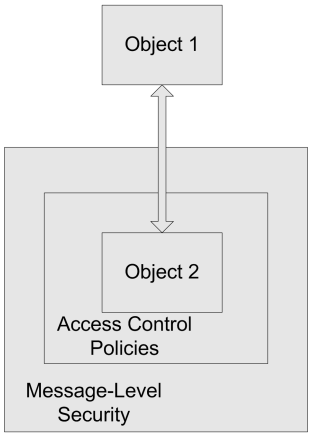


Figure 2.5: Legion Security Model.

Authentication: The Kerberos authentication protocol is fundamentally based on clients obtaining tickets for the use of services. Tickets are unforgeable tokens obtained from a distribution server through a protocol that involves the actual authentication of the user through password entry. Just like in Globus, in order to avoid the repeated entry of passwords, a special Ticket Granting Ticket (TGT) is obtained by clients. This TGT is a credential that can then be used for a limited time to obtain further tickets that are required to access individual services. Clients can obtain specially marked TGTs that can be forwarded to proxies for use within a limited time period. The use of these forwarded TGTs is the basis for employing Kerberos as an authentication mechanism within Legion. The services provided by Kerberos (e.g., obtaining forwardable user credentials) are available in other systems, such as the Secure Sockets Layer (SSL). In fact, both Kerberos and SSL can be called through a generic interface: the Generic Secure Service Application Program Interface (GSSAPI). By using GSSAPI, straightforward extensions to other systems such as SSL are possible.

Access Control and Privacy: In Legion, access is the ability to call a method on an object. Access control is not centralized in any single part of the Legion system. Each object is responsible for enforcing its own access control policy. It may collaborate with other objects in making an access decision and indeed, this allows an administrator to control policy for multiple objects from a single point.

The general model for access control is that each method call received at an object, passes through a "access control" layer before being serviced. Policies for this layer are specified as events in the configurable Legion protocol stack. According to these policies it will be decided whether to grant access according to whatever policy it implements. The default implementation provides a more sophisticated access control layer that implements access control lists and credential checking. Access control lists can be specified for each method of an object.

Collaboration Monitoring: is limited to monitoring the contribution and consumption of participants. Information on parameters such as the amount of CPU used, the number of messages sent and received, the total volume of data moved in and out of the object and the number of method invocations performed are registered. Contribution is monitored similarly.
Past Behavior: On termination, the collected information is forwarded to the instantiation manager. The instantiation manager in turn places the resource data into a host specific accounting database. Periodically the data is collected, merged into a single database and reports are generated.
The mechanism offers the possibility to gain information who is using and who is contributing resource and in what quantities.

- **Unicore [53]:** UNICORE security is based on the Secure Socket Layer (SSL) protocol and X.509V3 type certificates (Fig. 2.6). Because the participants' id and group are mapped onto his/her certificate on each UNICORE site, there is no need to use Grid-wide user identification.
  Authentication: For authentication by the gateway, each participant uses his/her

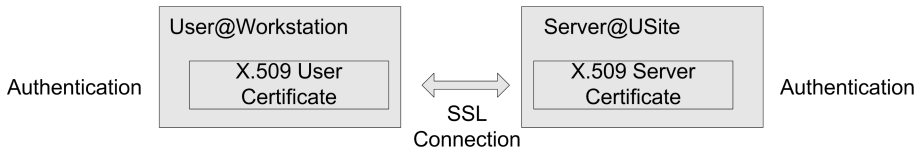


Figure 2.6: Unicore Security Model.

certificate for identification. The user interface application maintains the certificates and needs to know the Certification Authority which signs the user and the gateway certificates.
Certification Infrastructure: UNICORE has a single Certification Authority (CA) localized at LRZ (Leibniz-Rechenzentrum, Muenchen/Leibniz-Computer Center Munich) and all other centers run a Registration Authority (RA). Additionally, each participant has its own X.509 certificate that it uses to communicate with the Gateway via the SSL protocol.
Access Control and Privacy: possession of a valid certificate enables all participants to take advantage of the underlying functionalities. There are also two levels that retain full control over resources. Only participants that are registered in the login database may submit their tasks. The second allows the specification of system dependent regulation and limits for a particular machine.
Trust: is an implied notion regarding only the central CA.

- **Condor [8].** Condor security is also based on the Secure Socket Layer (SSL) protocol and X.509 type certificates (Fig. 2.7).
  Authentication: By default, Condor's authentication simply uses the participants

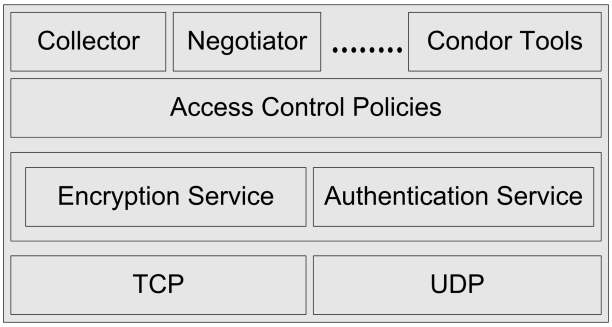


Figure 2.7: Condor Security Infrastructure.

identity (resulting from the information contained in the certificate) to determine who a participant is.
Certification Infrastructure: The role of the CA is played by the pool (VO) administrator(s).
Encryption and Confidentiality: encryption is used for data sent across the network. Condor can encrypt either the data that it sends for internal communication, or participants data, such as files and executables.
Access Control and Privacy: Certification Authority (CA) maintains an access control list as well. This list allows the administrators to control which participants can join a pool and what the rights will be.
Integrity: Condor offers the possibility to check if data sent across the network have been tampered or not using additional cryptographic data. This can be used to prevent so-called "man in the middle" attacks in which an attacker changes the data being sent without either side being aware of the modifications.

Trustworthy and reliable security is essential for all the Grid initiatives discussed above as the basis for setting-up a Grid environment which demands convenient and automatic access to available resources (Friese et al. [146] and Smith et al. [243]). There exist mechanisms and tools which provide Grid applications with the ability to access the resources they require when they require them and which also allow users and their collaborators to monitor and steer their applications wherever they are on the Grid, while at the same time it is required to maintain an accepted level of security to prevent any malicious behavior in the environment. Since the Grid spans different security domains, it should be able to operate in conjunction with these various local solutions. It cannot require a total replacement of local security solutions, but rather must allow mapping into the local environment. Enhancing security through trust consideration could be an efficient solution to this problem.

2.3 Threats on Grids

Security is one of the most challenging aspects of Grid computing (Smith et al. [244], [242] and [241]). The Grid community has successfully adapted and applied existing security approaches and mechanisms to protect Grid participants from those that could be potentially

malicious. These solutions focus primarily on authentication, access control and ease of collaboration. Authentication will identify each participant and ensure that no un-authorized parties are involved. Access control ensures that the participant is allowed to use the resources and/or services offered by remote participants. Finally, ease of collaboration is introduced to implement many of the design goals of Grid computing, such as single sign-on, virtual organizations and interaction among multiple administrative domains.
But how reliable and trustworthy are the Grid systems in reality?
From their very nature, Grids are designed to provide access and control over enormous remote computational resources, storage devices and scientific instruments. The nature of the information exchanged, saved or processed sometimes is very valuable and as a result, the Grid can be an attractive target for distilling such information. Each Grid site is independently administered and has its own local security solutions such as Kerberos [210] and PKI [43]. These solutions are built on top of different operating systems. When all participants are brought together to collaborate in this "extremely" heterogeneous environment, many security problems arise [242], [241]. In general, Grid systems are vulnerable to all typical network and computer security threats and attacks. Furthermore, the introduction of Web Service technology in the Grid (Foster et al. [143]) will bring a new wave of threats, in particular, those inherited from XML Web Services.
A comprehensive security system, capable of responding to any attack on Grid resources, is indispensable to guarantee its anticipated adoption by all interested participants independently from being individuals or organizations. The conception of this security model for the Grid requires also an analysis of the threats that do exist in Grid environments. Without such an analysis, it will be very difficult to implement the right enhancements to the current security models for Grid environments.
One of the first analysis on Grid security breaches was made by LCG group [32]. Some basic classification of security threats from the operational point of view in Grids could be:

- Misuse.
- Confidentiality and data integrity.
- Infrastructure disruption.
- Accidental.
- Web Applications and XML Web Services vulnerabilities when considering dependency on the technology.

Demchenko et al. in [125] and Navqi in [206] described some other models of vulnerabilities of Grid systems.
We separate the vulnerabilities in Grids into vulnerabilities regarding Grid resources and vulnerabilities regarding Grid services. Furthermore, one deals with models of uncertainty in Grid environments and of deceptive behavior of Grid participants.

2.3.1 Vulnerabilities of Grid Resources

Theft of Credentials and Private Keys. The authentication process is mainly based on the exchange of participants' credentials. If not properly protected, this personal information

could be an easy target to attacks with the sole aim to steal or compromise it. Single participants in the environment can impersonate certain identities and try to act maliciously without fearing that their real identity can be compromised. Some consequences of intentional impersonation are discussed further in the following sections when deceptive behavior is considered.

Data Integrity Attacks. These threats are directly related to the security offered by the physical Grid infrastructure. The integrity of the stored data, replica files and other data that are under the "execution process" is dependent on the integrity of the Grid hardware.

Denial-of-Service (DoS) and Distributed-Denial-of-Service (DDoS) Attacks. Such attacks, especially DDoS attacks are aimed at the availability of Grid services and Grid resources. Thus, authorized participants are prevented from successfully joining the Grid and participating in it.

Communication Attacks. This type of attacks regards the "leak" of information being transmitted between participants. The presence of security gaps worsens the security of the communication. As a result, the data being exchanged are vulnerable to different kind of attacks (i.e. passive wire tapping).

2.3.2 Grid Service Vulnerabilities

The deployment of Web Service technologies in Grids (Foster et al. [117]) adds new vulnerabilities and attack strategies. As Demchenko [124] states, specific for this kind of attack, from the point of view of applications and network protection, is that the malicious participants do not interrupt the normal operation of their target participant.
Based on the articles of Negm [208] and [207], Linstrom [185] and Bloomberg et al. [90], where analysis and classifications of Web Service vulnerabilities are presented, the following vulnerabilities for Grid services can be derived:

Service Interface Attacks. WSDL is the description mechanism for Grid services containing all methods and parameters used to access the service. This information on how to access the service could expose the service to possible attacks.

Credentials Tampering. Grid services, just like the Web Services, use XML credentials for authentication, authorization and session or state management. A poor implementation of the security in the Grid systems, poor key generation or key management and lack of encryption, are the causes of additional vulnerabilities like XML content theft or manipulation.

XML Content Manipulation. XML documents may be manipulated to contain malicious parsing or processing instructions (XML Schema extensions, XPath or XQuery instructions, XSLT instructions, etc). These malicious extensions make remote applications or hosting environments more vulnerable in the face of additional threats.

Variations of DoS Attacks. Many Grid applications may allow complex or voluminous XML input which, used intentionally by malicious participants in the environment, may overload the XML parsing system resulting in a variation of DoS attacks. Furthermore, XML

has the ability to include references to external documents or data types. This can be intentionally used by malicious participants to involve remote Grid sites to start (D)DoS attacks in the environment. Least, but not last, the SOAP messaging infrastructure can be also the target of typical network/infrastructure based attacks.

2.3.3 Dealing with Deception

Deception encompasses different behaviors consisting not only of "blatant lies", but also of common deceptive behavioral mechanisms including concealment, exaggeration, equivocation, half-truths and misdirection (Buller et al. [93]). In fact, in everyday life, while carrying out interpersonal relationships, one can expect to either witness or be the conveyor of a deception (Decaire [123]).
Why do participants in Grid environments deceive? The answer could as simple as "*for the same reason why humans do*", so for hiding themselves and/or planning a deceivable action. Grid environments are conceived and set up by human users to serve the human users through establishment of "interpersonal relationships" between Grid participants. Considering the security infrastructure for Grid environments discussed above, every participant in the environment has the potential to deceive the others and at the same time is vulnerable in the face of deception.
In Grid environments one can definitely talk about a "*planned deception*" in terms of Mitchell [201], carried out by a single participant or in collaboration with others.
Here, deception in Grid environments is organized into two groups; identity related deception and behavior related deception.

Identity Related Deception. In Grid environments, it is relatively simple to hide one's true identity, or to make use of someone else's identity. For such a reason, the main problems regarding identity deception deal with:

- Creation of a New Identity - The current certification infrastructure makes it easy to every participant to obtain and be represented by several identities; all that a participant has to do is ask for a certificate offering a "cheap pseudonym" (von Laszewski [76]). This certification procedure makes participants even more vulnerable. They can find themselves under the attack of malicious participants, whose identity could be no more traceable, or collaborate with others in the environment who offer mediocre services.

- Impersonation - Impersonation makes it possible for malicious participants to hide themselves behind the identity of others when acting in the environment. Current Grid security uses technologies that make use of public key infrastructures and digital signatures that can be manipulated. Poor implementation of the security infrastructure, poor key generation or key management and lack of encryption makes the credential theft and credential tampering in Grid systems possible.

Identity related problems make it even more difficult for Grid participants to have certainties on the identity of their collaboration partners and as a result to have confidence on their behavior and on the output of the entire collaboration process.

Behavior Related Deception. The current Grid infrastructure favors the presence of

malicious participants whose aim can be harming the others for deriving mainly personal profits. This deceptive behavior can be categorized as:

- Abusive Behavior - During collaboration, partners exchange information on each other and data to be processed as well. This information, especially the confidential one, can be intentionally forwarded to remote third parties or be used by single collaboration partners contrary to the owners intentions.
  Another form of abusive behavior is packet dropping. Malicious participants intentionally drop the data that should be sent to the interested participant instead.

- Mediocre Behavior - In current Grid environments, there does not exist any tool that verifies the competence and the honesty of its participants. This allows single participants, despite their modest capabilities, to compete with others in the environment. Mediocre behavior also is the intentional decrease of one's performance during the collaboration.

- Offensive Behavior - Offensive behavior are the attempts made by malicious participants to attack the others in the environment for preventing them from normal functioning, gain access to confidential information or making use of them for initializing a second attack from.

In this work, a solution that deals with the categories of abusive and mediocre behavior is presented. Proposals on how to deal with different categories of offensive behavior are also given in the next chapters of the thesis.

2.3.4 Dealing with Uncertainties

Uncertainty plays an important role in Grid environments. It has really tight connections to the information available about the environment, especially its participants. Information is the mean for gaining knowledge about the others, minimizing the risk of collaborating with deceivable partners. The more knowledge exists, the less the uncertainty. Uncertainty is viewed as a manifestation of some knowledge deficiency, while knowledge is viewed as the capacity to reduce uncertainty. In Grids, there does exist knowledge-based uncertainty. It relates to the amount of knowledge about the structure and causalities of the environment, knowledge about the real intentions of participants and reliability on the outcomes of the collaboration processes.
The following categories for uncertainties in Grids are identified:

Uncertainties on the Identity. Uncertainties regarding the identity of Grid participants relates to the simplicity of changing or manipulating their identity information. Current technology offers the possibility to the participants to identify their collaboration partners, but no assurances on their real identity. As a result, suspicions on the behavior and the intentions of single participants exist.

Uncertainty on Current Behavior. This is the result of absence of tools for monitoring the ongoing collaboration between participants. Output of the monitoring process helps to create a history of the behavior of all collaboration partners. In current Grid environments there does not exist any model of participants' behavior where their behavior is determined

as deceivable by comparing current behavior with their typical behavior and with the set of rules and preferences established by Grid users.

Uncertainty on Future Behavior. The absence of a history regarding the past behavior of the collaboration parties makes it also difficult to reason and create a logic on possible future behavior.

In Grid environments, uncertainty is an additional problem to deal with. An adequate solution should be offered to the participants in order to reason on the others' identity and behavior as well in the presence of uncertainty.

2.4 Summary

Grid computing is evolving toward a reality with scalable, flexible and dynamic participants that provide a rich set of functionalities. The potential benefits of Grid computing are enormous and range from data sharing and hosting to health, financial, simulation, video and image processing services.
Current Grid environments can be defined as:

- **Heterogeneous:** Grid environments aggregate large numbers of independent and geographically distributed computational and information resources, including supercomputers, workstation-clusters, network elements, data-storages, sensors and services. Similarly, applications typically combine multiple independent and distributed software elements such as components, services, real-time data, experiments and data sources.
- **Dynamic:** the Grid computation, communication and information environment is continuously changing during the lifetime of a collaboration between participants. This includes the availability of the participants as well as their state, services and data.
- **Uncertain:** Uncertainty in Grid environment is caused by multiple factors, where the most important are:
  - Unpredictable and changing behaviors of participants introduced by the dynamicity and heterogeneity of the environment.
  - Possible failures, which have an increasing probability of occurrence as number of participants in the environment increase.
  - Incomplete knowledge or perhaps no knowledge at all on participants in the environment.
- **Vulnerable:** As previously analyzed, Grid systems will be vulnerable to all typical network and computer security threats and attacks. Furthermore, bringing Grid technology next to Web Services technology brings new threats inherited from XML Web Services.

Security is the key attribute that needs to be addressed within the Grid. Current Grid technologies offer authentication solutions based on the application of cryptography, PKI and X509. Current security solutions have many deficiencies which makes Grid computing yet

not so convincing for the commercial world.
Many of the security challenges in Grid environments can be addressed by considering management and enforcement of trust policies within a dynamic Grid environment. Defining a trust model is the basis for interoperability but the trust model is independent of interoperability characteristics. Trust is a complex subject relating to belief in the honesty, truthfulness, competence, reliability, etc., of the considered participants.
In a trust management system, participants which assign some degree of trust to other participants based on a combination of identity considerations, behavior observations, recommendations from other participants and references to the trust sources are involved. This is typically done by each participant resulting in its own trust policy. The degree of trust that the participants assign to each other governs the decisions that they make when collaborating with each other.
An enhanced solution needs to consider:

- **Hierarchical infrastructure for assigning certificates** - all participants are able to issue certificates for themselves and the others, but some participants have to be considered as more trusted than the others as certification authorities.

- **Continuous monitoring of the collaboration** - The monitoring process helps Grid participants to control if their collaboration partners are behaving correctly or not. It is the mean for gathering knowledge on the others' behavior and thus decreases the level of uncertainty in the environment. This process should last through the entire lifecycle of the collaboration, and the information accumulated can serve for personal future decisions or be offered to the others in the environment.

- **A probabilistic model for expressing beliefs regarding the behavior of Grid participants** - Just like as Dix in [130] argues that in several real-life situations an agent behavior may be known with a given degree of uncertainty, in Grid environments, the behavior of its participants may also be known with given degrees of uncertainty. Given that probability distributions are as sensible means to describe the uncertainty between several possible behaviors (i.e. successor behaviors of a given current one), the type of analysis involved must also be a probabilistic one. In such a probabilistic framework, the Grid participants may have the possibility to express their beliefs (degree of suspicion) regarding certain behavior properties (or the behavior as a whole) of their collaboration partners.

- **Identity and Behavior** - Identity of a participant is part of the Grid authentication and authorization process but at the same time it expresses the belief that this participant is who he/she claims he/she is.
  While making collaboration decisions, Grid participants have to consider both identity and the behavior of the partners they are going interact with. Furthermore, mapping the identity of a participant with his/her behavior may discourage malicious participants from trying to impersonate someone else in the environment and take part in a collaboration process.

- **Decentralization of the control and management** - In Grid environments, centralized monitoring of distributed participants can become impractical when the number of

participants is large, or the monitoring frequency is high. The large volume of monitoring data means that high bandwidth connections and huge hosting platforms are needed to download the data. Furthermore, using a central site for hosting behavior monitoring information, also bears security risks. It can be an attractive target for malicious participants who may want to steal or manipulate the saved information. Decentralization is an important step to be taken in Grid environments. Each of the participants should be able to manage on its own the collaboration process with others, although in this case more responsibility and expertise is assigned to a single participant.

- **Different sources for collecting information on collaboration partners** - It is important that participants make use not only of their personal experience regarding past direct collaborations with other participants, but at the same time use the experience of others in the environment together with social notions like prejudice, optimism, beliefs and transitivity.

Grid computing has the potential to evolve into something more than a commodity. In a future in which computing, storage and software are no longer objects that we possess, but utilities to which we subscribe, the most successful scientific communities are likely to be those that succeed in assembling and making effective use of appropriate Grid infrastructures and thus accelerating the development and adoption of new problem solving methods within their disciplines (Berman et al. [86]).
This makes Grid technology as the next logical step towards utility (on-demand) computing [14].

Chapter 3

Trust in Collaborative Grid Environments

"Nach dem Spiel ist vor dem Spiel"

Sepp Herberger

3.1 Introduction

All Grid participants are expected to provide a certain level of confidence in terms of security and quality of service. This will affect their "existence" (being present and active in collaborations with others) in the environment. However, pure consideration of only the security aspects like authentication and authorization is insufficient to make them less vulnerable against different threats and to ensure a successful and "undisturbed" interaction. The problems identified in the previous chapter can be somehow minimized by considering the management and enforcement of the notion of trust in the environment.
In general, trust and security are very tightly coupled (Lamsal [180]). They can even be considered as complementary to each other. The effect that different aspects of trust like reputation, honesty and credibility have in establishing trust relationships between participants in the environment has to be considered together with security.
Trust is an important aspect of Grid environments. It is the underlying concept behind every decision for collaboration.
There are a number of ways that Grid participants can establish trust among them. First, a participant interacts with the target participant(s) and learns its/their behavior over a number of interactions. In this case, the participant reasons about the outcome of the direct collaborations with others. When starting a collaboration with a new participant, it can use its beliefs about different characteristics of the interaction partners and also reason about the beliefs in order to decide how much trust should be put on each of them. Second, the participant could ask others in the environment about their experiences with the target participant(s). If sufficient information is obtained and if this information can be trusted, the participant can reliably choose its collaboration partners.
When trusting a participant, it is important to know which aspect one refers to. There are instances where a participant is trusted more than the others regarding the different levels of trust. There must be the possibility to specify in which aspect of trust participants are

interested in and at which level. Trust towards a participant should also be handled in different contexts. These contexts should be used to decide whether a participant is eligible or not for a certain activity. The overall value of trust of a participant should interfere with the decision of improving its social position among the others. Thus, trust is *a social value that pushes participants to collaborate with each other.*
There is the need to separate different activities in which a participant is involved and the need to consider not only the behavior of the participant in offering and requesting services or resources but the quality of goods as well.
Two participants continue to collaborate or set a collaboration with each other based on direct or indirect good experiences they formerly had.

Considering a decentralized and context-oriented notion of trust as well as experiences, beliefs and intentions adds a lot of flexibility to a Grid environment. Although more responsibility and expertise is assigned to every participant, they are able to manage their own trust relationships with others.
In the rest of the chapter, the most important trust features for Grid environments will be discussed. Finally, different identified threats to trust will be presented.

3.1.1 Trust Taxonomies

The notion of trust has been treated in a number of publications ranging from sociology, psychology to computer science. Though several definitions and classifications of trust already exist, there is no clear consensus on the definition of this notion. Delimiting this notion is even more critical in the domain of Grid computing, where trust plays a central role for establishing collaboration among participants in the environment. Definitions of trust are more effective when the entire process of trust establishment and evolution is considered. Each of the participants in the environment possesses an objective, underlying level of trust. This reflects exactly the degree to which they will fulfill the assigned tasks or reciprocal obligations during collaborations. However, the subject that trusts a target participant does not fully know this underlying trust value and must gather all the available information from all possible sources in order to attribute by itself a level of trust to the collaboration parties.

3.1.2 Definitions

Trust is an important aspect in every domain ranging from everyday life to social and computer sciences. Most of the activities between parties are initiated based on the sole supposition that a certain level of trust already exists between them.
Trust is usually specified in terms of a relationship between a *"trustor"*, the subject that trusts a target participant and a *"trustee"*, the participant that is trusted. Trust forms the basis for allowing a trustee to use or manipulate resources owned by a trustor or may influence a trustor's decision to use a service provided by a trustee.
In Grid environments, collaboration takes places between different participants. A collaboration partner is either a service provider (e.g. a node to host and provide a service, or a service instance running on the provider node) or a service consumer (e.g. a node that requests a service from a provider (which includes the request to deploy and perform a service at the provider), or a service instance running on the consumer node).

There are two major aspects that influence the selection or acceptance of an interaction partner:

- The *identity* of the interaction partner or more specifically the trust that one can put in the credibility of the identity an interaction partner claims to have.
- The *past behavior* of the interaction partner as an indicator for its *future behavior*. This behavior can be determined considering a multitude of dimensions, such as the accuracy of delivered results, actual costs compared to expected costs, availability of the service, response time, or fault and intrusion properties. Furthermore, the trust values might be different for different applications/services the interaction partner offers or requests.

In the following it will be discussed:

- *"what trust is"*,
- *"what trust is not"* and
- "*how trust in Grid environments can be defined*".

What Trust Is Supposed to Be. There are several attempts to define trust.
The most interesting definition is given by Merriam Webster dictionary [34], where trust is considered as:

- assured reliance on the character, ability, strength, or truth of someone or something
- dependence on something future or contingent (hope)
- reliance on future payment for property (as merchandise) delivered (credit)
- a property interest held by one person for the benefit of another
- a charge or duty imposed in faith or confidence or as a condition of some relationship
- something committed or entrusted to one to be used or cared for in the interest of another (care, custody)

According to Marsh [191], the majority of work on trust originated from sociology, (social) psychology and philosophy. The most important definitions in these domains are given by Morton Deutsch, Niklas Luhmann, Bernard Barber and Diego Gambetta.

Deutsch [126] argues that trust depends on the individuals' perceived cost and benefits analysis of the specific scenario. Such a definition involves some kind of uncertainty and at the same time indicates some optimism. If the perceived benefits were greater than the perceived harmfulness then the significance of trust in the choice would not be that big. In [127], he shifts towards confidence for defining trust. Thus, trust is the confidence that one will find what is desired from another person rather than what is feared. This shift towards confidence indicates the inclusion of hope in trust.

Luhmann [188] and [189], emphasizes that in order to live in a society, several assumptions are needed and trust is the mean for reducing the complexity of the society. The particular

situation and the particular environment have to be taken into account before taking trust decisions. Trust plays a significant role in the interaction of a single individual with the society.

Barber [85], links trust with expectations about the future:

- Expectation on the persistence and fulfillment of the natural and moral social orders.
- Expectation on "technically" competent performance.
- Expectation that interaction partners will "carry out their fiduciary obligations and responsibilities, that is, their duties in certain situations to place others' interests before their own".

Gambetta [147], defines trust as follows:
"Trust (or, symmetrically, distrust) is a particular level of the subjective probability with which an agent assesses that another agent or group of agents will perform a particular action, both before he can monitor such action (or independently or his capacity ever to be able to monitor it) and in a context in which it affects his own action.
When we say we trust someone or that someone is trustworthy, we implicitly mean that the probability that he will perform an action that is beneficial or at least not detrimental to us is high enough for us to consider engaging in some form of cooperation with him.
Correspondingly when we say that someone is untrustworthy, we imply that that probability is low enough for us to refrain from doing so."
Furthermore, he describes trust as:

- a particular expectation one has with regard to the likely behavior of others,
- something suspended between faith and confidence,
- a tentative and intrinsically fragile response to the ignorance, a way of coping with "the limits of our foresight",
- a result rather than a precondition of cooperation.

Finally, according to Grandison [154], in information technology, trust can be defined as:
"the firm belief in the competence of an entity to act dependably, securely, and reliably within a specified context."

What Trust Is Not Supposed to Be. Kaplan [178], gives his definition of what trust is not:

- Trust is not transitive - it can not be *passed* from a person to another and furthermore *preserving* the same level.
- Trust is not distributive - different persons *can not share* the same trust value and the trust on a single person *can not be transformed* in a unique value for the group.
- Trust is not associative - it *can not be linked* or *added* to another value.

- Trust is not symmetric - two partners *must not necessarily* trust each other at the same level.
- Trust is not self-exclaimed - everyone *can exclaim* his/her underlying trust level, but no one else *must necessarily* trust it; trust must be gained.

The following applies to "what trust is not" definition as well:

- Trust is not equal to reputation - reputation reflects the "social position" of an individual among all the others in the environment while trust is a property established between two individuals through direct collaborations.
- Trust is not the opposite of risk - while risk can be considered as the opposite of trust, having a certain level of trust does not mean "risk-free".
- Trust is not a substitute for security mechanisms - as previously stated, trust and security are complementary to each other.
- Trust is not as the "prisoner's dilemma" [44] propagates it - trust is established by considering different sources for getting trust information from and not a purely socio-cognitive property.
- Trust is not specified only in terms of "trusted" and "not trusted" - two individuals trust each other on an individual scale.
- Trust is not static - trust changes (evolves or regresses) over time.
- Trust is not absolute - identity and behavior are two aspects that influence the selection of collaboration partners. Furthermore, behavior of individuals needs to be evaluated considering different aspects. A trustor trusts a trustee with regard to its capability of performing specific actions or providing specific services within a certain context.
- Trust (in terms of beliefs and altruism) is not a handicap - it is good to trust, it is good to have some risk involved. *Without trust we cannot stand (Confucius).*
- Trust is vulnerable - vulnerability is a weakness that can be exploited (Kaplan [178]). Alternatively, a weakness in a trust establishment and management system could be exploited in order to violate trust[1].
- Trust is not an exclusive property - trust is not assigned; everybody in the environment can earn it.
- Trust is not enough - a high level of trust should in no case mean "blind trust" and as a result a low reliance on control mechanisms.
- Trust is not cost-effective - trust is a complement to security and means control. Control mechanisms could in a certain sense raise the individual costs of the participants.

[1]The threats to the trust in Grid environments are treated at the end of this chapter.

Definition of Trust. In this thesis, a multi-faceted definition for trust is considered:
"Trust is the extent to which every participant in a Grid environment, in a specific moment of time, with an evidence of relative security regarding the identity and the behavior of their counterparts, is willing to interact with them, even though unexpected negative outcomes could result from the entire collaboration process."

When trusting a participant, it is important to know which aspect one refers to. There are instances where a participant is trusted more than the others regarding to different levels of trust. There must be the possibility to specify in which aspect of trust participants in the environment are interested in and at which level. Trust towards a participant should be handled in different contexts. These contexts should be used to decide whether a participant is eligible for a certain activity and the overall value of trust of a participant should interfere with the decision of improving the social position among the others.
Trust in Grids still preserves its social value. It pushes participants to collaborate with each other, having certainties that the kind of requested goods and services are offered with a sufficient quality level (Fig. 3.1).

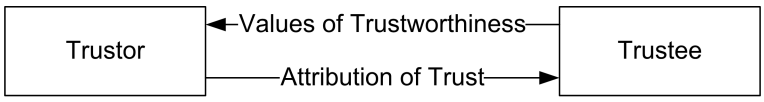


Figure 3.1: Trustor-Trustee Collaboration.

Considering Gambetta's description of trust ([147]), it is further concluded that trust is the *"moral prize"* that the trustor assigns to the trustee at the end of the collaboration regarding its behavior during collaboration and at the same time it is *"the precondition"* for future collaborations.

3.1.3 Classification

Trust in this work is classified principally into identity trust and behavior trust:

Identity Trust (I^T). Every participant in the environment is known through its identity. Trust on identity reflects the confidence of the trustor on the declared identity of the trustee.

Behavior Trust (B^T). During the collaboration, it is expected from every participant in the environment to behave dependably, securely and reliably. Behavior trust reflects the confidence of the trustor on the expected behavior of the trustee.
Behavior trust depends on different aspects, such as type of connection, type and quality of

services offered, processing power, quality of resources, etc. In different situations, Grid users can have different needs with respect to the overall capabilities of single participants in the environment, e.g.:

- the user may want to use some computing power in the shortest time possible at the lowest cost,
- the user may want to take advantage of any special service with a certain quality of service offered by one or more participants,
- the user may want to save critical data at some remote sites expecting the the data will not be manipulated.

Overall Trust. The trust T that an interaction partner X has for partner Y is influenced by both identity trust T^I and behavior trust T^B:

$$T_X(Y) = T_X^I(Y) \cdot T_X^B(Y) \tag{3.1}$$

The relationship between overall trust, identity trust and behavior trust is expressed in Fig. 3.2.

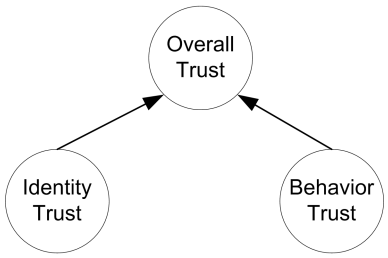


Figure 3.2: Overall Trust.

3.2 Trust Dimensions

While sources of trust are from where trust cues originate, dimensions of trust are the operational attributes to which they contribute (Bailey et al. [84]).
In the following, dimensions of trust in Grid environments will be discussed.

3.2.1 Attraction

Attraction among Grid participants involves the overall capabilities that they offer in the environment. Independent of the role a participant is playing at a certain moment of time, "attractiveness" is specified in terms of "appearance", determined primarily by its interface and functionalities exclaimed. These elements are very important since a Grid participant that "looks good", is also as a reasonable partner for collaborating with.

3.2.2 Identification

Trust in Grid environments should also be linked to the identity of the active participants. If the identity of a participant is uncertain, as discussed in the previous chapter, it is difficult to fully trust that specific participant.
The relationship between trust and identification in Grids has not been explored in depth yet. Since in Grid environments there is the possibility for the participants to possess a signed certificate which is validated entitling them to collaborate with others in the environment, further knowledge of the identity is not required. However, possibilities for further investigation on the "real" identity of the trustees must be offered, in order to fulfill explicit requirements of trustors.

3.2.3 Control of Collaboration Partners

The relationship between trust and control is of a great importance for the notion of trust and for modeling and implementing control mechanisms regarding the behavior of participants in Grid environments.
Trust and control assume each other's existence, but they do not necessarily coexist in every scenario. According to Möllering [203], trust produces control and control produces trust. Control is trust-building when trust in trustee's autonomous willingness and competence would not be enough. Both trust and control exist in a reflexive relationship to each other, establishing the means of positive expectations. For a trustor, when coming to the behavior trust or overall trust of a trustee, control completes and complements trust. Furthermore, the control:

- determines whether a collaboration was successful or not,
- deals with the possible deviations and unexpected events in order to cope with them.

Castelfranchi and Falcone, in [102], argue that control requires new forms of trust:

- trust in the control itself,
- trust in the trustee as for being monitored and controlled,
- trust in possible authorities, etc.

Depending on the circumstances, control makes trustee more reliable or less reliable. As a result, control can either decrease or increase trust.

3.2.4 Competence and Reliability

Competence implies that the trustee possesses the knowledge, expertise and ability to fulfill the needs of the trustor (Chopra and Wallace [107]). A related attribute is credibility (Doney et al. [132]), the degree to which information provided by the trustee can be believed.
Competence in Grid environments is incorporated by the attributes of:

- correctness - the participant delivers the proper outputs or payments,
- availability and accessibility - the participant is up and running whenever it is needed.

Reliability of a collaboration partner is measured in terms of dependability, predictability or consistency. It is directly related to past experiences and prior collaborations.

3.2.5 Satisfaction

The trust that the trustor places on the trustee during (or at the end) of the collaboration increases the ultimate benefits the trustor gains from collaborating with the trustee. In a certain sense, not only does the trustor gain the desired outcome from the collaboration, but the trustor also gains the "satisfaction" of having evidence that the trustee is capable to fulfill its needs, behaving accordingly to the trustor's expectations, showing itself as trustworthy.
Only when a trustor, according to the gathered evidence on past collaborations, establishes a belief that a trustee is capable of performing its required functions and will act on the trustor's best interests, the trustor can fully expect that the trustee will fulfill the trustor's needs to perform a specific task.
A trustor's continuous sense of satisfaction, regarding the collaboration process with a specific trustee, will have a positive effect on trustor's perception of effort and performance expectancy toward the trustee.

3.2.6 Commitment

The higher the level of commitment, the greater the probability a participant will achieve a better outcome. Expected benefits coming out of the collaboration normally exceed the costs associated with maintaining this kind of relationship between them (the very aim of a Grid participant for joining the environment).
Commitment during the collaboration results in an increase of the trust that the trustee has already established to the trustor.
The accumulated trust is only an evidence of commitment during past collaborations and does not give any warranties for the commitment of the trustee in future collaborations. It merely reflects the trustor's expectations regarding the behavior of the trustee in the future.

3.2.7 Intentions

Intentions are not characteristic of machines in Grid environments; they are not characteristic of computers at all at their present stage of evolution. Intentions, however, do become "insidiously implanted" in Grid participants. Developers, administrators and users in Grid must be careful to properly define their goals and objectives. If the participant's purpose is perceived to be somehow hostile, then distrust will result.

3.2.8 Expertise

Expertise is related to the skills, capabilities, or knowledge of a collaboration party. In general, experts are perceived as being more trustworthy than the others (Brainov et al. in [91] and Peters et al. in [220]). In Grid environments, the assessment of expertness is related, to some extent, to breadth of knowledge, depth of knowledge, etc, offered by the specific participants for the solution of specific problems. An optimal solution to a difficult problem is most impressive with respect to the capabilities of the party/parties involved in the collaboration.

3.2.9 Dynamicity

Dynamism is used to describe how active a Grid participant is. With respect to the trustee, a trustor's opinion is affected by its perceptions of a trustee's responsiveness, i.e. meaning how quickly it is engaged in a collaboration process and how quickly it works towards the solution of assigned tasks.

3.3 Trust Sources and Formalisms

Trust is a multidimensional value that can be derived from different sources. First, prior to any collaboration, trust has an initial value that reflects a trustor's general predisposition or willingness to trust. This predisposition reflects to some extent the trustor's collaborative attitude (Mayer et al. [196]). Second, trust may be based on knowledge and inference (attribution) of the trustee's abilities, traits, goals, norms, values and circumstances. This knowledge may be based on the trustor's own direct experience with the trustee. Third, trust may be based on the reputation that the trustee has in the environment. This kind of trust is formed through the collaboration in the past between the trustee and others in the environment. A fourth source of trust could be the socio-cognitive trust as discussed by Castelfranchi and Falcone in [100] and [101]. This type of trust formalizes the concepts derived from human social interaction and its relationship to trust. Ideas from different fields like psychology or sociology are synthesized. In this case, it makes use of outcomes from examination of the behavior of deceptive individuals and the behavior of collaborative individuals or collaboration of all individuals (group collaboration). The empirical basis for defining the elements for assessing trust for making trust decisions and how trust among individuals evolves over time are provided.

3.3.1 A Social Network Model for the Relationships between Participants in Grids

Experience is an important factor that influences the trust between participants in Grid environments. It is actually the result of the interaction between participants.
We distinguish between three types of experiences:

Direct Experience. If a trustor assigns some tasks to a trustee, the quality of the outcome of the execution of these tasks reflects the trustor's experience with the trustee. This type of experience is called *direct experience*. Observations of the trustee's behavior, through recording the outcome of the collaboration process, are essential for a subjective evaluation of its trustworthiness.
The term *direct trust* refers to the trust that is built from direct collaborations (that is, using observations by the trustor itself). Even in the case when the participants are unknown at the beginning, a collaboration can be established (depending on the security policies). In this case, an initial value can be assigned by default, which will later on be changed by the trustor on the basis of the outcome of the direct experience with the trustee.

Third Parties' Experience. The quality of the experience itself is important. This can be considered as subjective, meaning that different participants may not estimate the quality

of the experience with a specific collaboration partner at the same level. The experience that third parties in the environment have had with a specific participant is called *third parties' experience* or *recommendations*. Recommendations from third parties provide the possibility for trust regarding unknown participants to be propagated. Recommendations are based purely on the recommender's personal direct experience. Obtaining recommendations becomes more important in cases where no direct experience with a specific target participant exist. Thus, recommendations allow participants to consider collaboration even with unknown entities.

Experience with Recommenders. The third parties' experience is *per recommender* and as such it is possible to associate a measure of trust in the opinion of the recommender itself. In cases where recommendations affected the trustor's decision to collaborate with the trustee, the quality of those experiences will also serve to evaluate the quality of each individual recommendation to arrive at the formation of an opinion regarding the recommenders. Let us consider the case when a participant X (the trustor) assign its tasks to a participant Y based on the recommendations by a participant Z (third party). Here, X relied on Z's recommendation. If the collaboration between X and Y was evaluated as successful according to the point of view of X, than Y is most probably suitable for future collaborations. The outcome of the collaboration with Y affects X's opinion about Z's future recommendations. In other words, direct experience of X with Y, affects X's opinion regarding the recommendations of Z. This is the experience with recommenders.

The relationships between Grid participants can be modeled as a social network (Fig. 3.3), where participants in the environment are represented through a set of vertices and relations between them are represented through a set of edges.

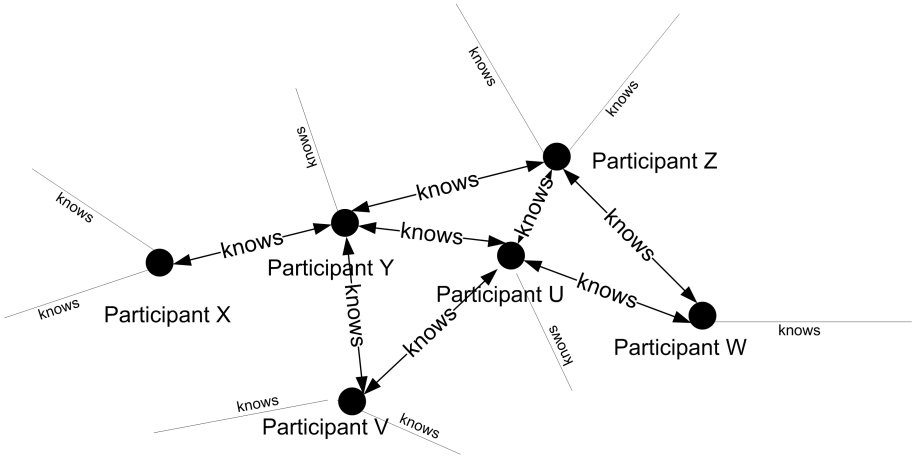


Figure 3.3: Social Network Model for the Relationships Between Participants in Grids.

3.3.2 Optimism, Intentions, Beliefs and Risk

Social trust includes characteristics like optimism, belief in collaboration and confidence that collaboration parties can resolve their differences and "live" a satisfactory "social life" together.
According to Patil et al. [218], optimism is the positive expectation a participant has for another participant or an organization based on past performance and truthful guarantees. Trust is about expectations in future collaborations. Participants' intentions behind collaboration play an important role in a successful outcome of the entire collaboration process.

Trust is considered to be an epistemic notion (Christianson and Harbison [108]) consisting of beliefs (Castelfranchi and Falcone [103]).
Statements about trust regarding individual participants in the environment are statements of personal beliefs held by others. The reasons for holding them do make such beliefs true in the real world (Patil et al. [218]).
Trust values regarding past collaboration partners involve one's beliefs derived from the available evidences, intentions and readiness for undergoing possible risks. One should not expect Grid participants to behave monotonously on these basic parameters involved in trust formation. A generic framework for managing trust among participants should provide means and methods for the participants themselves to express their beliefs while avoiding unintentional transitivity of trust (Christianson et al. [108]).

The extent to which a participant can be optimistic is threatened by the risk associated with the unsuccessful or undesired outcome of the collaboration. Risk and trust helps for making decisions in uncertain collaborative environments. Participants in the environment continue to collaborate with unknown participants even though they are aware of risk existence. It is because of their intentions behind the collaboration and their beliefs in evidences associated with the collaboration with specific participants as well.
Trust means in no way certainty. Some uncertainty is always present and some probability of failure must be considered. A participant in the Grid environment must accept this fact and run such a risk. Thus, when a trustor trusts a trustee there are two risks:

- the risk of total failure - outcome of the collaboration is disastrous.
- the risk of wasting the efforts - not only fails the trustor on achieving its goals but also wastes other resources (e.g. monetary).

Relying on another party for establishing a collaboration with may be considered as a risky process. It implies always the presence of some uncertainty, but it also requires some predictability of the other party and inevitably some degree of trust in it.

3.4 Trust Properties

Trust is necessary to enable Grid participants to collaborate with each other. Every trust relationship has basic properties to which it adheres.
Trust, which reflects a subjective degree of individual belief about other participants, is dynamic and non-monotonic (Marsh [191]), meaning that personal experiences may increase

or decrease one's degree of trust. Trust beliefs are also influenced, as previously shown, by reasoning, social stereotypes, communication and spreading of reputation (Bacharach and Gambetta in [83]). To trust means, one is vulnerable to the outcome of a decision and risks and uncertainty are inevitably involved with it. With the above definitions of trust it is possible to identify the properties of trust. Properties of trust include subjectivity, ability to reduce complexity, non-transitivity, context and ability to be measured. The identified properties of trust, which are included in the construction of the trust model presented in this thesis, are introduced in the following:

- Trust is a knowledge gaining process - trust is a process of individual learning to trust others in the environment.

- Trust is not an ever flourishing property - some problems can be tolerated by the participants, but when a certain threshold is reached, trust can flip to distrust. For the trustee, fixing the individual problem will not lead to a regain of the lost trust on the trustor's side.

- Trust is always accompanied by risk - during the collaboration, the presence of some uncertainty on the outcome of the collaboration process with a specific participant is presumed, but at the same time some degree of trust is also needed in it.

- Trust is dynamic - experience and knowledge about the collaboration party is accumulated with time, and as a result, the degree of trust that the trustee builds at the trustor is under constant re-evaluation and changes with time.

- Trust is subjective - one property of trust that is important in social networks is its subjectivism. Two participants often have very different opinions about the trustworthiness of a third participant in the environment. Trust is based on a trustor's prior direct experiences and gathered knowledge. *Trust is inherently a personal opinion*. Trust on a trustee ranges from complete distrust to complete trust. When knowledge on the trustee is lacking at all, then this case can be defined as "ignorance" and the trustor simply is not able to make a categorization regarding the trust on a trustee.

- Trust is asymmetric - the asymmetry of trust reflects a specific type of personalization. For two participants involved in a collaboration, trust is not necessarily identical in both directions. Because participants perceive the experiences differently, it is understandable why they may also trust each other differently. According to Hardin [157], trust is mutual, meaning that each participant has some trust for the other, but there do often exist differences in how much they trust one another. Asymmetric trust can arise in any relationship, as supported from Golbeck [149], and representations of trust relationships in models of social networks must allow for these differences.

- Trust relationship - participants in the environment are interested to deal with single identities. Thus, the one-to-one relationship is considered as best suited for Grid environments.

- Trust is measurable - the trust level is a measure of belief in another participant, in its honesty, competence, security and dependability.
  In the environment, some participants may be trusted more than others. In order to

have an accurate view on the level of trust regarding a specific participant, this has to be a continuous value. However, there is still a problem relating to the representation of ignorance (or lack of knowledge) with respect to trust.

- Trust is compositional - there are a number of ways that a Grid participant can establish trust with its counterparts. The participant can interact with the target participant(s) and learn its/their behavior over a number of interactions. In this case, the participant reasons about the outcome of the direct interactions with others.
  When starting an interaction with a new participant, meaning no information on previous behavior exists at all, its beliefs about different characteristics of these interaction partners can be used, in order to decide how much trust should be put in each of them. Furthermore, the participant could ask others in the environment about their experiences with the target participant(s). With information from many people, there is simply more reasoning and justification for the belief (Golbeck in [150]). If sufficient information is obtained and if this information can be trusted, the participant can reliably choose its interaction partners.

3.5 Relationship between Trust and Quality of Service

QoS refers to the ability of a trustee to provide network and computation services that each trustor's expectations for timeliness and performance quality are met. According to Quality Management and Quality Assurance Standard [57], QoS is defined as "*the totality of characteristics of an entity that form its ability to satisfy stated and implied needs.*"
There are many different usages of the word quality. The most important are:

- "conformance to requirements" - leads to the idea that "*quality costs less*" and
- "degree of excellence" - which implies that "*quality costs more*".

There are several dimensions of QoS described in the literature like Chatterjee et al. [105], Vendatasubramanian et al. [254], Rana et al. [73] and Maximilien et al. [194] which include parameters like accuracy, precision and performance. For a QoS dimension to be supported, it means that participants request or specify a level of service for one or more attributes of these dimensions and the underlying control mechanisms should be capable of keeping those services at the requested levels. The requirements for quality are an *expression* of the needs or their translation into a set of quantitatively or qualitatively stated requirements that enable the realization and examination of the collaboration processes.
The overall assessment of the QoS is performed by Grid participants themselves, since they are the ones able to evaluate the efficiency of the services offered and the grade of fulfillment of their requirements.
The collected information gives to the participant the means for assessing the various quality characteristics of its collaboration parties. It is used later on during the categorization of the behavior of its partners for continuing collaboration only with those that fulfill its requirements.

QoS assures not only the behavior of single Grid participants through the monitoring of

QoS parameters, but defines also how the whole environment manages given tasks. QoS deals with a range of expected behaviors of individual participants which only as a whole define the completion of the service a user (or an application) demands. In this context, it is important to map user's expectations and preferences into the system parameters and capabilities.
Trust in Grid systems can be defined as having confidence that a interaction party will offer the desired QoS, behaving as expected. Trust management is the process of deciding what participants are to be trusted to complete what actions and if the interested participant can be allowed to use the services offered or not. A trust system for Grid environments should offer flexible and easy to use components that can be configured to the specific needs of a user, considering different roles (consumer or provider) that a platform could play in different moments of time.

Abstracting the common attributes from the variety of demands that the human user, aiming at an optimal level of QoS, on a per case basis, places to the participants in the environment, the components of the behavior trust could be derived from the parameters of QoS like: reliability (correct functioning of a service over a period of time), availability (readiness for use), accessibility (capability of responding to a request), cost (charges for services offered), security (security level offered), performance (high throughput and lower latency), etc.
Each of these parameters can be directly measured or broken up into measurable elements, in order to offer the possibility to create a history with data from past interactions among collaborating parties in Grid environments.

Fig. 3.4 gives a view of the behavior trust elements considering different roles (consumer or provider) that the participants play at certain moments:

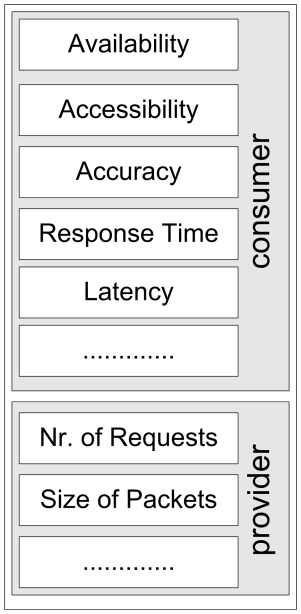


Figure 3.4: A General View on Behavior Trust Elements in a Grid Environment.

The QoS requirements can be divided into:

- Quantitative requirements - here, parameters like collaboration reliability, accessibility and availability of services offered, performance, responsiveness, cost, etc., are included. These parameters can be described in terms of measurable values such as bandwidth, failure percentage and absolute times, given also the nominal values and tolerances determined at the beginning of the collaboration.
- Qualitative requirements - here, parameters like dependability, efficiency, flexibility, robustness, interoperability, security, etc., can be mentioned. These requirements are not expressed in absolute values but they are comparable and always subject to participants' perception and evaluation.

Managing trust between parties in a collaboration means somehow also quality management. The establishment of a quality management system helps to identify if the collaboration parties were committed and the output of the collaboration process was not under the requested standards.

3.6 Trust Metrics

Trust metrics define the measure of amount of trust that the trustor attaches to the trustee. Quantitative metrics, qualitative metrics, or a combination of these two should be considered to measure trust levels.

3.6.1 Trust Values

Any trust relation is associated with a value that represents the expectations or the strength of beliefs. In the literature, there is no real consensus regarding the representation of trust. According to Ding et al. [129], the level of trust the participants establish to each other can be either boolean or numeric. Aberer et al. in [67], use a sort of "binary" trust. Participants are either trustworthy or not. Another alternative for representing trust values is to make a categorization of trust regarding a participant. Azzeding et al. in [79], use this approach. Trust is categorized as (very low trust level, low trust level, medium trust level, high trust level, very high trust level, extremely high trust level). A similar schema for the trust categorization is given by Golbeck in [151]. Trust values are specified on a scale of 1 to 9 where (1: Distrusts absolutely, 2: Distrusts highly, 3: Distrusts moderately, 4: Distrusts slightly, 5: Trusts neutrally, 6: Trusts slightly, 7: Trusts moderately, 8: Trusts highly, 9: Trusts absolutely).
However, the usage of these semantics does not give an objective view on the real trust of a participant (trust is subjective; different participant may use different categorization for the same experiences). Furthermore, this approach gives a loss of sensitivity and accuracy, since comparisons become coarse grained with no way to distinguish between participants' behavior.
In order to have a more accurate trust assessment, the numeric values are seen as most suitable, although even in this case, there does not exist any proper agreement (Ding et al. [128]). In eBay [10], participants receive feedbacks with one of the three values (+1, 0, -1) for their

reliability during each collaboration. Marsh, in his thesis [191], represents trust as a continuous variable using as scale for trust values [-1, +1]. He states that trust can have threshold values that vary between individuals and situations. A participant will have a positive threshold value, above which another participant is trusted and a negative threshold value, below which they will not trust a specific participant. Waguih [255], considers a numeric trust system consisting of nine grades ranging from absolute distrust to absolute trust within [-1, 1]. It proposes that a participant is categorized acording to: -1 (Absolute Distrust), -0.75 (High Distrust), -0.5 (Moderate Distrust), -0.25 (Slight Distrust), 0 (Ignorance), 0.25 (Slight Trust), 0.5 (Moderate Trust), 0.75 (High Trust), 1 (Absolute Trust).
Sloman [240], expresses the range of trust levels as integers in [0, 100] where high trust is represented in [90, 100], low trust in [5, 20] and a default initial trust in [0, 50]. Negative values represent distrust.

In this work, probabilities are used as parameters of subjective belief (denoted as confidence levels) to denote the trust values. The probability for a successful future interaction among partners is considered as closely related to the mutual trust values the partners assign to each other. These values vary in $[0, 1] \subset \Re$, where 0 means that the other partner is not trusted at all or there are uncertainties due to the lack of information (the condition of "ignorance"), and 1 means that it can be fully trusted and gives certainties on the success of the interaction that is going to take place.
In this way:

- trust measurement (identity and behavior trust) is reduced to a normalization of the measured value to range $[0, 1]$.
- these confidence parameters are an evidence of past experiences and in a certain sense reflect the risk associated and the expectations for the future.
- according to the user specific needs and requirements it is easier to automate the decision whether the trust established with a certain participant would be enough to establish a collaboration with.

3.6.2 Metrics on Clients' Side

In the following, some of the QoS/Behavior Trust elements, as seen from the clients' side will be presented (client is represented through X and the provider through Y).
This list is subject to extension as soon as new elements are considered as necessary to assess the behavior (trust) of a provider during the collaboration.

- Availability - represents the state when a provider is present and ready to fulfill the client's request. It could be measured as:

$$T_X^{Availability}(Y) = \frac{N_{total\_success}^{Availability}(Y)}{N_{total\_verif.}^{Availability}(Y)} \tag{3.2}$$

  where $N_{total\_success}^{Availability}(Y)$ represents the number of successful verifications of the availability of the provider (service) and $N_{total\_verif.}^{Availability}(Y)$ represents the number of total verifications of the availability of the provider.

- Accessibility - represents the state when an available provider is capable of serving the clients' requests. It could be measured as:

$$T_X^{Accessibility}(Y) = \frac{N_{total\_success}^{Accessibility}(Y)}{N_{total\_verif.}^{Accessibility}(Y)} \tag{3.3}$$

  where $N_{total\_success}^{Accessibility}(Y)$ represents the number of successful verifications of the accessibility of the provider (service) and $N_{total\_verif.}^{Accessibility}(Y)$ represents the number of total verifications of the accessibility.
  A non-available service is implied to be not accessible as well.

- Accuracy - reflects the provider's competence, reliability, commitment and intentions. It can be measured as:

$$T_X^{Accuracy}(Y) = \frac{N_{total\_success}^{Accuracy}(Y)}{N_{total\_verif.}^{Accuracy}(Y)} \tag{3.4}$$

  where $N_{total\_success}^{Accuracy}(Y)$ represents the number of total correct responses (resulting after verification) received from a provider (service) and $N_{total\_verif.}^{Accuracy}(Y)$ represents the number of total verifications of the accuracy of the responses from that provider (service).

- Response time - represents the time taken for a provider to complete a client's request and return a response. It can be measured as the time between sending a request to a provider (service) and receiving a response from it.

- Latency - intended to measure the speed with which a provider can process a given request. Possible measures can be conducted using the time when the request reached the service and time when the service finished processing the request.

- Throughput - represents the number of completed service requests over a time period (Ran [224]). Throughput is related to latency and in a certain sense influences the response time; the higher the throughput, the shorter the response time. It is also true that as the number of concurrent clients increases, the provider's throughput is multiplexed amongst a larger number of connections and hence the response time increases.

- Packet dropping - if an available and accessible provider received a task to complete, but for a certain reason did not return any kind of results, then it is supposed that the task assigned by the client was dropped. It could be measured as a fraction of the total number of responses the client should have received from a specific provider:

$$T_X^{P\_Dropp}(Y) = \frac{N_{total\_success}^{P\_Dropp}(Y)}{N_{total\_verif.}^{P\_Dropp}(Y)} \tag{3.5}$$

  where $N_{total\_success}^{P\_Dropp}(Y)$ represents the number of total responses the client received from a provider and $N_{total\_verif.}^{P\_Dropp}(Y)$ represents the number of responses that the client expected in total from that specific provider.

- Packet dropping duration - represents the "duration" of drop events from a provider. It can be measured as the number of consecutive packets dropped in each event.

- Bandwidth - represents the amount of data a specific provider instantaneously receives and transmits. It can be measured as the average number of bytes (or packets) able to be received and transmitted in a second.

3.6.3 Metrics on Providers' Side

In the following, some of the QoS/Behavior Trust elements, as seen from the providers' side, will be presented (provider is represented through X and the client through Y). This list, just like in the case when client was the trustor, is subject to extension as soon as new elements are considered as necessary to asses the behavior (trust) of a client during the collaboration.

- Client availability - just like in the case when the client was the trustor, availability represents the state when a client is present and ready to receive the provider's response. Formula (3.2) could be used for measuring client availability.

- Client accessibility - again just like in the case when the client was the trustor, accessibility represents the state when an available client is capable of accepting the result coming from the provider. Formula (3.3) could be used measuring client accessibility. Once again, a not available client could be considered as not accessible as well.

- Number of concurrent requests - represents the number of simultaneous requests that a client can make to a provider. Considering a service-oriented Grid environment as in Srinivasan et al. [246] or in the thesis of Friese [145], the client could assign more than one task simultaneously to a provider e.g. simply by invoking different instances of the same service offered by that provider. In case the number of concurrent requests is somehow limited by any pre-contract[2] (Web Services Agreement Language [54]) between the client and the provider, respecting this clause is "a must" for the client and part of its behavior during the collaboration. It can be measured as:

$$T_X^{Conc\_Req}(Y) = \frac{N_{total\_success}^{Conc\_Req}(Y)}{N_{total\_verif.}^{Conc\_Req}(Y)} \tag{3.6}$$

  where $N_{total\_success}^{Conc\_Req}(Y)$ represents the number of successful verifications (client respected the agreement) and $N_{total\_verif.}^{Conc\_Req}(Y)$ represents the total number of verifications of the number of concurrent requests made from the client.

- Size of sent packets - once again, in case the size of the packets the client is allowed to send to the provider is somehow limited by any pre-contract [54] between the client and the provider, respecting this clause is again "a must" for the client and part of its behavior during the collaboration. It can be measured as:

$$T_X^{Packet\_Size}(Y) = \frac{N_{total\_success}^{Packet\_Size}(Y)}{N_{total\_verif.}^{Packet\_Size}(Y)} \tag{3.7}$$

[2]Actually, each of the considered metrics, either from client or provider side, could be subject to a preliminary agreement between the client and the provider. Each of the metrics specified in the agreement is "a must" for the party, otherwise it has not behaved accordingly. Trust values are always influenced from the resulting behavior (positively or negatively).

where $N^{Packet\_Size}_{total\_success}(Y)$ represents the number of successful verifications (client respected the agreement) and $N^{Packet\_Size}_{total\_verif.}(Y)$ represents the total number of verifications of the size of packets sent from the client.

- On-time payment - represents the number of payments made at the right time by the client. The time of payment can be specified in an agreement between the client and the provider. It can be measured as:

$$T_X^{On\_Time}(Y) = \frac{N^{On\_Time}_{total\_success}(Y)}{N^{On\_Time}_{total\_verif.}(Y)} \tag{3.8}$$

where $N^{On\_Time}_{total\_success}(Y)$ represents the number of successful verifications (client respected the agreement) and $N^{On\_Time}_{total\_verif.}(Y)$ represents the total number of verifications of the payment times.

- Due payment - regards the due amount of the reward the client owes the provider for the services offered. It can be measured as:

$$T_X^{On\_Time}(Y) = \frac{N^{On\_Time}_{total\_success}(Y)}{N^{On\_Time}_{total\_verif.}(Y)} \tag{3.9}$$

where $N^{On\_Time}_{total\_success}(Y)$ represents the number of successful verifications (client respected the agreement) and $N^{On\_Time}_{total\_verif.}(Y)$ represents the total number of verifications of the payment times.

- Response loss - if an available and accessible client, for a certain reason was not able to receive the response sent by the provider, then it is supposed that the response got lost. It can be measured as a fraction of the total number of responses the provider sent to the client:

$$T_X^{P\_Loss}(Y) = \frac{N^{P\_Loss}_{total\_received}(Y)}{N^{P\_Loss}_{total\_sent}(Y)} \tag{3.10}$$

where $N^{P\_Loss}_{total\_received}(Y)$ represents the number of total responses the client received from a provider and $N^{P\_Loss}_{total\_sent}(Y)$ represents the number of responses that the client expected in total from that specific provider.

- Response loss duration - distribution of the "duration" of loss events, measured as the number of consecutive packets lost in each event.

3.7 Trust Threat Analysis

Trust is vulnerable and anything or anyone that can exploit any vulnerability constitutes a threat (Kaplan [178]).
It is related to the fact that participants have a degree of freedom to disappoint the expectations. It is also related to the limits of anyone's capacities ever to achieve a full knowledge of others, their motives, their responses to endogenous as well as exogenous changes (Gambetta [147]).
Threats define events, the occurrence of which could have an undesirable impact (Ozier [211]).

Threat analysis includes the identification of possible threats that may adversely impact the target participant. Without threat management and assessment, the participants may fail to objectively establish trust among them.
The potential for harm caused by the presence of threats could be also answered from the four questions posed by Ozie while analyzing and assessing risk (Ozier [211]):

- What could happen? (What is the threat?)
- How bad could it be? (What could the impact or consequence be?)
- How often might it happen? (What is the frequency?)
- How certain are the answers to the first three questions? (What is the degree of confidence?)

Central to the notion of trust is the condition of ignorance or uncertainty as defined in the forth question. If no uncertainty could exist in the environment, then the trust information the participants gathered on each other could be more reliable. Reliability on trust information is the confidentiality that one should have regarding the offered experience and the current behavior of others in the environment.
The two primary types of adversaries in Grid environments, able to put reliability of trust at risk are fraudulent and malicious participants. They are primarily distinguished by their goals in the environment. Fraudulent participants wish to have a considerable profit for their "mediocre" contribution or achieve a better "social" position in the environment to the detriment of the other participants. The goal of malicious participants on the other hand, is to cause harm to either specific targeted participants in the environment or to the environment as a whole.
To accomplish their goal, both types of participants are willing to exploit any vulnerability (Negm in [208] and [207], Lindstrom [185] and Bloomberg et al. [90]) and any type of coalition with other participants (Jennings et al. [122]).
A threat model is normally used to describe a given threat and the harm it could do to the participants in the presence of vulnerabilities. The construction of the threat model is important for the construction of the overall trust management system. In the evolution of computational Grids, security threats were overlooked in the desire to implement a high performance distributed computational system (Naqvi [205]). This applies to the threats on trust as well. The conception of a comprehensive trust management system for the Grid environments requires a realistic threat model. Otherwise, without such a threat model, there is the risk of wasting time and effort implementing the trust management system.
In the following, trust threat profiles help to identify the specific threats that put the reliability of trust information at risk in Grid environments.

3.7.1 Abusive "Gossiping"

Participants have the possibility to exchange their personal direct experiences. This should not be seen as an obligation for the participants, but merely as a possibility to exchange information on others and thus helps to reduce the level of the uncertainty in the environment. Each of the participants should independently decide whether to consider this kind of information and at what degree.

In this case, the possible threat to the trust information offered to participants is abusive or even malicious "gossiping" by fraudulent or malicious participants with the sole aim to:

- discredit participants in the environment - for certain targeted participants or for everybody else, low trust values are offered to interested parties, or
- support certain target participants for an undeserved profit - higher trust values, meaning a greater competence are offered regarding certain target participants in the environment.

3.7.2 Deceiving Trust

A participant can gather information and learn the behavior of its partners over a number of direct interactions. In this case, the participant reasons about the outcome of the future interactions with these participants. This trust information can be abused by fraudulent or malicious participants. The following threats to this type of information are identified:

- From "*High*" to "*Low*" - A higher level of trust gives some assurances on the competence and the reliability of the target participant. The existence of this general principle in Grid environments bears some disadvantages. Fraudulent and malicious participants can use it in order to deceive their interaction parties. At the beginning, they could fulfill the expectations of their interaction parties and offer services of high quality, reasons for which they were chosen among the others in the environment, but as soon as their they have reached a "*high social position*" in the environment, they start acting differently, lowering the quality of their offered services.
- Trust Manipulation - In current Grid environments, for each of the participants it is very easy to change or manipulate their identity information. Current technology offers the possibility to the participants to identify their interaction partners, but no assurances on their real identity. As a result, suspicions on the behavior and the intentions of single participants exist. This problem becomes serious especially in cases when malicious participants impersonate the identity of highly trusted participants and try to collaborate with others in the environment.
- Stealing Trust Information - In general, Grid systems will be vulnerable to all typical network and computer security threats and attacks.
  The nature of the trust information saved (direct experiences) can be considered as valuable and as a result an attractive target for malicious participants for distilling valuable information from (e.g. a list of most trusted participants in the environment could be extracted in order to attack them or hinder their normal activity).

3.8 Summary

Trusting a participant means believing that when offered the chance, it is not likely to behave in a way that is damaging to the others.
So far, Grid technology has been primarily used by researchers. These users have tried to benefit at maximum from being able to share resources in the Grid and have no intention of harming the resource owners or fellow users. This is all about to change. The number of

Grid users is growing fast. The growing size and profile of the Grid require comprehensive security solutions as they are critical to the success of this technology. A comprehensive security system, capable of responding to any attack on Grid systems, is indispensable to guarantee its anticipated adoption by the users and the resource providers as well.

Considering the uncertainty of Grid environments where all the participants are free to disappoint the others, free enough to avoid a risky collaboration and free enough to consider a specific collaboration party as an attractive option, trust gains a great relevance, especially with respect to the level of reliability that the participants should assign to each other.
In practice, it is not possible to know in advance whether a certain participant can be trusted or not. This deduction should be done considering both identity and the behavior of the collaboration parties. Identity trust is concerned with verifying the authenticity of the collaboration partner, whereas behavior trust deals with the trustworthiness of that party.
The overall behavior trust of an interaction partner can be built up on several factors. Considering the relationship between quality of service (QoS) and trust, different QoS properties like availability and accessibility of the service, accuracy of the response provided by the service, response time, cost of the offered services, security etc., can be considered and modeled as behavior trust elements that a consumer uses to rate a provider. In a similar way, the total number of (concurrent) requests coming from a consumer or the size of the packets received from it can be considered as behavior trust elements from the point of view of a provider.
These factors of behavior trust should be continuously tested and verified. In this way, it is possible to collect a history of past collaborations that can be used for future decisions on further collaborations between partners. This kind of experience can also be shared as recommendations between participants. Considering the social value of trust, some socio-cognitive trust decisions should also be included regarding the future behavior of the collaboration party(parties).
The level of trust represents the *evidence* a participant was able to gather about its collaboration parties in the past, the *beliefs* of this participant regarding the behavior of these partners in future collaborations and as a result the level of *intention* of participants to collaborate with each other.
A trust management system for Grid environments should be flexible enough to reflect all discussed trust properties and contain flexible and easy to use components that can be configured to the specific needs of the users on per case basis.

Chapter 4

Related Work

"We must respect the past and mistrust the present,
if we wish to provide for the safety of the future."
Joseph Joubert

4.1 Introduction

Trust is used in different domains. McKnight and Chervany [197] offer a typology of trust and a classification of the existing research on trust in domains like sociology, psychology, management, economics, and political sciences. Trust is thereby classified conceptually into six categories:

- disposition - the trustor is inclined to trust,
- situation - the trustor trusts only in particular scenarios,
- structure - the trustor trusts impersonally the structure the trustee is part of,
- belief - the trustor believes the trustee is trustworthy,
- intention - the trustor is willing to depend on the trustee,
- behavior - the trustor voluntarily depends on the trustee.

A mathematical model of computational trust should be capable of expressing all such aspects, as well as further notions of primary relevance in computing, e.g. that trust information is time dependent and varies very rapidly. Also, it should be sufficiently general to allow complex structures representing combinations of different types of trust.

The related work regarding trust modeling is grouped into four main categories:

- General models of trust.
- Identity-based trust models.
- Behavior-based trust models.
- Other approaches to trust management.

4.2 General Models of Trust

General models of trust are frequently motivated by work in security and policy representation. Trust and security are related, dependent concepts with different purposes.
Policies are aimed toward the expression of *when*, *what* and *how* trust in a participant is determined, without considering the problem of trust establishment and many important elements of trust management. Furthermore, some of them assume that trust is established through an external trusted system. Here, languages like XACML (eXtensible Access Control Markup Language) from OASIS [62], or SAML (Security Assertion Markup Language) [45] should be mentioned.
Moreover, the work of Carbone et al. in [96], contributes to the division of trust in different types, yielding a policy language that aids the control over trust decisions.
An approach, similar to role-based access control [28], where access decisions are based on the roles that individual users have as a part of the organization where they take part, is the one described by Kagal et al. [176]. A trust management framework that uses a system of rights and delegations, as well as digital certificates to facilitate trust management is proposed. Trust management involves developing security policies, assigning credentials to participants, verifying that the credentials fulfill the policies, delegating trust to third parties and reasoning about participants' access rights. This approach allows the delegation chain in which users are able to delegate their rights to other users that they trust. Once users are given certain rights, they are responsible for the actions of the users to whom they subsequently delegate those rights and privileges.

Josang et al. [165], [167], [166], [168], [170], [169], [173] and [171], proposed a subjective logic for beliefs. His approach can be used to model and rationalize beliefs relating to levels of trust in a system. Through the proposed subjective logic, an opinion value along three axes: belief, disbelief and uncertainty is computed. He describes a scheme for combining opinion values and a protocol for initiating a trust relationship and evaluating trust values.

Some other contributions in the trust domain are presented as a result of related research in game theory. Buskens [94], offeres a variant of the "Trust Game". It is analogous to the prisoner's dilemma, but set in a market scenario. The game theoretic approach is used for measuring a type of trust from the graph of a social network.
An approach that uses also game theory is that of Brainov and Sandholm [91]. The authors show that underestimating trust hurts all participants involved and a mutual level of trust maximizes the utility. Once again, the game defined by this work is a market-based scenario, where buyers and sellers are the main actors. In this work, the fact that trust could be considered as an alternative for dealing with uncertainty is underlined.

Platform for Content Selection (PICS) [238] provides rules that together form a kind of filter between the web documents and their viewers based on policies. It was developed by the World Wide Web Consortium [56] to protect primarily children from pornography on the Internet. PICS offers some rating that determine the appropriateness of a target internet page. All PICS-compliant applications should be able to read the labels and together with the user-defined filtering rules decide whether a specific document can be accepted or rejected.

Another approach that can be classified in the category of general trust models is the "Free Haven" system [16]. It describes a design for a publishing system aiming to a distributed, anonymous, persistent data storage which is robust against malicious attempts by others in the environment.

4.3 Identity-Based Trust Models

The majority of the so-called "trust models" based on the identity of the involved participants deal primarily with authentication. The Public Key Infrastructure (PKI) [69] authenticates the owner's identity using digital certificates. A digital certificate is issued by a certification authority and verifies that a public key is owned by a particular participant [209]. It does not handle the policies regarding what involved participants are allowed to undertake.
Two other known certificate systems are PGP [209] and X.509 [138]. They make use of digital certificates based on public key cryptography. That is, these two models can be used to guarantee the identity of the originator or the recipient of an object. The PGP trust model assumes *no centralized or hierarchical relationship* between certification authorities [154]. In PGP systems, the user generates a pair of keys, a public and a private one, associated with his unique ID (name, e-mail address). PGP's digital certificates are used primarily for privacy and authentication relating to e-mail type of applications between human users. On the other hand, X.509 is a *strictly hierarchical* trust model used for authenticating web transactions (i.e., authenticating the user or the web server) by offering a digital certificate as a proof of identity. In X.509 framework, everyone will obtain certificates from an official certification authority (CA). Certificates contain more information than PGP. They contain the names of the signature schemes used to create them and the time interval in which they are valid. The certification authority *does not guarantee* the trustworthiness of the owners of the keys. They simply authenticate the owner's identity. This is necessary in order to establish access or provision rights for their bearers. The policy governing what the owners of the keys are permitted to access is not handled by the certificate infrastructure, but is left up to the trustor which may trust others, may validate their certificates or even trust third parties to validate certificates.
It is important to underline that none of these models can be used as a single trust model for all domains. PKI implementations contain no systematic and reliable methods for obtaining evidence about participants involved e.g. in an Internet transaction (Josang [174]). PGP lacks official mechanisms for the creation, acquisition and distribution of certificates and X.509 may lead to unnatural business alliances between competing companies as a result of rigid hierarchical structure, violating the natural establishment and propagation of trust. Additionally, some applications, such as the reference information distribution systems need certificates to have a lifespan longer than is currently allowed by each of the schemes separately.

Other approaches provide encryption methods for ensuring a trusted communication among partners. The Secure Sockets Layer (SSL) [158] uses application encryption for Web browsers. It is a protocol oriented towards protection of the data exchanged between Web browsers and Web servers, ensuring the provenance of the data, their secrecy and integrity. The approach used by IP Security (IPSec) [89] ensures the secrecy and the integrity of the exchanged data

through the implementation of network layer encryption and authentication providing an end-to-end identity-based trust solution. Another example in this category is the Kerberos protocol [179], which uses a third party to facilitate the exchange of credentials between users and computers. Kerberos is not oriented towards the determination of access rights, but simply enables two parties to securely exchange easily verified credentials.
A distinct problem that comes out during credential exchange is that of possible loss of privacy, since the partners are obliged to reveal their identity. Winslett et al. in [267] and [266], have focused on the trade-off between privacy and trust establishment. In their work, trust is established only thanks to the revelation of a certain number and type of credentials. Another approach, based on these principles is TrustBuilder [260]. Trust is established only if a sufficient number of credentials are revealed. Furthermore, some types of credentials affect trust more than others. Zheng et al. [271] offer a scenario where agents play in a variation of the prisoner's dilemma. Trust is measured as the amount of cooperation between two participants and the types of credentials include resumes, text-chats, and pictures of players. The type of credential, in their work, affects the amount of trust or distrust received.
PolicyMaker [88] is especially convenient in systems that include anonymity as a security requirement. It is a unified approach for specifying and interpreting security policies, credentials and relationships that allow direct authorization of security-critical actions. It is a tool in the development of services whose main goal is privacy and authenticity. PolicyMaker expresses security credentials and policies without requiring the application to manage a mapping between personal identity and authority. It also offers possibilities of expression of conditions under which an individual or an authority is trusted and the conditions under which trust may be deferred. It specifies what a public key is authorized to do (evaluates whether a proposed action is consistent with local policy). Policies are trust assertions made by the local system and are unconditionally trusted by the system. Credentials are signed trust assertions offered by other entities whose signatures must be verified before using the credentials.
KeyNote [87], is also used for specifying local security policies and security credentials that can be sent over an un-trusted network. KeyNote accepts as input a set of local policy assertions, a collection of credential assertions and a collection of attributes (action environment) that describes a proposed trusted action associated with a set of public-keys. Applying assertion predicates to the environment, it decides consistency of actions with a local policy. The result of the KeyNote evaluation process is an application-defined string, the simplest response being "authorized".
Another approach from IBM, the Trust Establishment Policy Language [29] and [162], similar to PolicyMaker, states that the underlying trust implications involved in an e-business transaction can be solved using certificates. It is a role-based access control model that uses certificates, a Java-based Trust Establishment module and a Trust Policy Language (TPL). Certificates can be issued by various participants, vouching for a specific participant in a particular role (buyer, seller or both). The "Trust Establishment module" validates client certificate and maps a role to the owner of the certificate. TPL is used to specify local policy which defines what a role is permitted to do. This framework can be used to define policies for all applications whose users are allowed to undertake only those actions assigned to them according to their roles.
Rule-controlled Environment For Evaluation of Rules and Everything Else (REFEREE) [109] provides both a general policy-evaluation mechanism (for Web clients and servers) and a lan-

guage for specifying trust policies. It places all trust decisions under explicit policy control. In the REFEREE model, every action, including evaluation of compliance with policy, happens under the control of some specified policy. That is, REFEREE is a system for writing policies about policies, as well as policies about cryptographic keys, PICS label bureaus, certification authorities, trust delegation, etc. It is based on PolicyMaker and considers a PICS label as the stereotypical web credential and uses the same theoretical framework as PolicyMaker to interpret trust policies and administer trust protocols, which are represented as software modules. Like PolicyMaker and KeyNote, REFEREE is a recommendation-based, query engine so it needs to be integrated into a host application. It evaluates requests and returns a value (true, false or unknown) and a statement-list, which is the justification for the answer.
Germano [148], distributes trust relationships and reputations to each participant. They can decide whom to trust based in the specifics of this architecture, where users certify other users' public keys without the need of a Certification Authority. Communication between two users takes place only if a link can be established between them.

All these approaches can be considered as "hard security approaches" to trust (trust is either present or absent). Trust is here defined as the output of the identity and authorization verification process, thus, after credentials and their claimed association are verified.

4.4 Behavior-Based Trust Models

4.4.1 Direct Experiences

The first formal computational model of trust has been presented by Stephen Marsh in his Ph.D. thesis [191]. He investigates the notions of trust in various contexts and develops a formal description of its use with distributed intelligent agents. "An imperfect understanding, a plethora of definitions and informal use in the literature and in everyday life" with regard to trust is addressed. In his model, Marsh proposes a set of variables and a way to combine them to arrive at a single continuous value of trust in the range $[-1, 1]$, where:

- -1 implies *complete distrust* on the target agent,
- 0 implies *lack of knowledge* regarding the target agent and
- 1 implies *full trust* on the target agent.

Marsh identified three types of trust (basic, over all contexts; general, between two people and all their contexts occurring together and situational, between two people in a specific context). He also identified time as being relevant to the variables used to comprise trust. Furthermore, he includes aspects of trust such as:

- Competence - the *level of experience* that agents have with each other,
- Group membership - an agent's level of trust towards agents from the *same group*,
- Agent disposition - an agent's *inclination towards cooperation* and
- Reciprocation - the *modification of behavior* based on a recent history of cooperation.

Although the work is considered as valuable for the definitions presented, it displays difficulties with the separation of the concepts of trust and distrust. Marsh also does not consider the possibility for the agents to share their accumulated trust information.
Following his work, some researchers have attempted to model the properties of trust and reputation in a computational setting. Resnick et al. [226] describes reputation as "important for fostering trust among strangers", creating a clearer picture for reputation. Their work outlines those features of reputation that make it valuable for Internet applications.

Castelfranchi and Falcone [136], express the idea that a higher level of reputation is useless without knowledge of the context in which that reputation was earned. The authors deal with a dynamic aspect of trust: *"does the observed behavior represents also the real intentions of that agent regarding future collaborations?"*. This model is a clear example of a cognitive trust model. The basis of their model is the strong relation between trust and delegation. Delegation finds place only if a specific set of beliefs and goals (the mental state that Castelfranchi and Falcone identify with trust) are met. The basic beliefs that an agent needs are:

- Competence belief - the belief of the trustor that the trustee *is capable* of performing the task.
- Dependence belief - the belief that it *is better to rely* on that specific trustee for performing the task.
- Disposition belief - the belief that the trustee *will* actually *perform* the task. It can be further divided into:
  - Willingness belief - the trustee intends to perform the task and
  - Persistence belief - the trustee is stable in its intentions while performing the task.

  Relevant is also the degree of importance of the goal that is going to be achieved through the delegation of the task to the trustee. The resulting degree of trust is obtained by multiplying the degree of those beliefs and goals useful to the trust relation. If this value exceeds a given threshold and it is also the best solution of all the available solutions, then the decision to delegate is taken.

Another approach is the one presented by Jonker et al. [164]. They focus on another aspect of trust dynamics, more precisely on how positive and negative experiences can change negative and positive trust, respectively. The outcome of this work suggests that *trust does change according to the type of experiences and that distrust may be harder to overcome*.

4.4.2 Third Parties' Experiences and Hybrid Trust Models

Trust in E-Commerce. Trust plays a crucial role in computer mediated transactions and processes. Online service provision commonly takes place between parties who in general are strangers to each other. Furthermore, the environment where the service consumers and providers act, often offers insufficient information about their counterparts. This forces the provider to accept the risk of dealing with a non-correct consumer, but especially it forces the consumer to accept the risk of prior dealing with a dishonest provider of goods (i.e. to pay

for services and goods before receiving them; consumer finds himself in a vulnerable position, etc.). The consumer generally has no opportunity to see and try products he is going to buy (contrary to the provider that knows exactly what he is going to get once the money has been paid). This information asymmetry is considered to be mitigated through the use of trust and reputation in such marketplaces. Even if the consumer (provider can also be included) does not know what he gets during the transaction, he can still be confident that it will be what he expects as long as he trusts the provider.
Reputations are effectively used in electronic marketplaces like eBay [10] as a measure of the reliability of involved participants. With eBay, buyers and sellers can express their negative (-1), neutral (0) or positive (1) votes for each other after each transaction. Votes so collected are used by the system to provide cumulative ratings of users, that are made known to all participants. Reputation is considered as a global property and these models use a single value, that is not dependent on the context it was earned. The only source of information used to build the value of reputation is the information coming from the others that has previously interacted with the target participant. Unfortunately there does not exist any kind of mechanism for dealing with false information provided.
Reputation systems like CNET [6], EPINIONS [11] and ALLEXPERTS [2] compute reputations based on the feedbacks of experts and reviewers. OpenPrivacy [37] introduces reputation services that can be used to create and calculate results from accumulated reputations.
Other models, like the one proposed by Halberstadt et al. [156], use social factors such as reciprocity that according to them constitutes the social motto *"be nice to others who are nice to you"*.

Mui et al. [204] have proposed a computational model where the concepts of trust and reputation are separated in electronic marketplaces. According to them, reputation is the "perception that an agent creates through past actions about its intentions and norms" and trust is "a subjective expectation an agent has about another's future behavior based on the history of their encounters". They express reputation as a probability of success ranging in $[0, 1]$. In this model, no effects of deception are considered. Furthermore, no reference to the minimum value of success that will push the agent to accept a counterpart exists.

Although some approaches, like the one discussed above, are offered for online marketplaces, the providers can still hide the quality of the goods they are going to offer. *The high level of information asymmetry creates a perfect situation for frauds, misuse and also a market for lemons* (Akerof [70]).
A general discussion of trust on the Internet is given by Friedman et al. [144], where characteristics of trust in online interactions are outlined. One of the key points presented there is that simply performing a task is definitely not the same as providing good service of high quality. This is a problem of automated reputation systems for electronic marketplaces that fail to capture this difference.

Trust in Multi-Agent Systems. Trust is a fundamental concern in multi-agent systems. It is considered to lie at the core of all interactions between collaborating agents that operate in uncertain and constantly changing environments. Given the particular features of these environments, trust components and the ensuing systems are increasingly being conceptualized, designed and implemented.

Considering relationships between agents, Jennings et al. [223] expect that agents exhibit a specific behavior in an interaction based on reputation from various sources. They focus on combining the sources of reputation and refer to direct experience as confidence. In some later work, [78] they claim that the context and the roles of interacting agents can somehow determine the rules of trust. In particular, the general relationships of trade, dependency, competition and collaboration can be mentioned. A main disadvantage is the fact that trust is implied to exist whenever it is believed that an agent will not gain at the disadvantage of another agent.

Another model presented by Jennings et al. in [159] and [160], combines multiple sources of trust such as reputation, context-based rules and credentials. In cases where no reputation or other sources of trust regarding a particular agent exist, endorsements of trust from other agents are used. In this model, the strong belief that the agents will report their trust information truthfully exists.

In [64] and [65] Abdul-Rahman et al. focus on providing a system where each agent is merely enabled to make trust decisions rather than automating the entire process. The main contribution of this work is the effort to decentralize the trust decision process.
The trust model they present later on in [66] uses four degrees of belief to express the trustworthiness of the agents:

- very trustworthy,
- trustworthy,
- untrustworthy and
- very untrustworthy.

For each of the partners and the contexts, the agent maintains a tuple with the number of past experiences in each category. Considering direct interactions, trust on a partner in a given context is equal to the degree that corresponds to the maximum value in the tuple. They make use of different trust categories:

- which aspect is trusted,
- a scale of trust values on recommendations and
- direct trust values (related to one context).

The biggest problem of this approach is that it is not possible to differentiate between lying agents and those who in a certain sense "think different". The model gives more relevance to the information coming from those agents with a similar point of view (agents that have a similar perspective in a given context). The model is intended to evaluate only the trust on the information given by recommenders. Direct experiences are used to compare the point of view of these recommenders with the direct perception of the agent and then are able to adjust the information coming from them accordingly.

The trust model proposed by Schillo et al. [231] considers a hard type of trust (the result of an interaction between two agents is a boolean impression meaning either good or

bad). Furthermore, the degree of satisfaction for interacting agents is not considered. They propose a Prisoner's dilemma set of games with a partner selection phase. Each agent receives the results of the game it has played plus the information about the games played by a subset of all its neighbor players. The result of an interaction is the impression on the honesty of the partner and its behavior according to the normal prisoner's dilemma actions (cooperation or defection). The model assumes that the recommenders never lie, but they can hide information in order to make other agents appear less trustworthy. No information is given about how to combine direct experiences with recommendations. The trust value is a subjective property assigned particularly by each individual and it does not depend on the context.

The model Zacharia [268] deals with direct information and recommenders' information. The reputation value is a subjective property assigned particularly by each individual. Direct experiences are limited to the use of the most recent experience with the agent that is under evaluation. More importance is given to the recommendations. A similar structure to Schillo et al. [231] is used. Ratings are represented from a directed graph, where nodes represent agents and edges the information on the most recent reputation rating given by one agent to another. The agents who have been rated directly by the generator of the graph have a reputation value equal to the rating value. In this model, the reputation value does not depend on the context and no special mechanisms are provided to deal with inappropriate recommendations. Furthermore, the reputation value assigned to an agent regarding its recommendations, compounds also its reliability.

Another approach that combines direct experiences with the reputation information from others is ReGreT [228] and [229]. The model adopts a sociological approach for computing reputation in societies of agents trading well defined products inside an e-commerce environment. The main contribution is the vision of reputation through:

- Individual dimension - the effect of personal past experiences with a given agent.
- Social dimension - refers to reputation inherited by individuals from the groups they belong, that is the reputation of the group that an individual belongs to also influences the reputation of the individual.
- Ontological dimension means that the reputation of an agent is compositional. The overall reputation is obtained as a result of the combination of the agent's reputation in each context.

A great importance is given to the freshness of the information. Computations give a fixed high relevance to recent rates over older ones according to a time dependent function. In REGRET, a minimum number of interactions are required for evaluating the reputation of a certain agent. REGRET does not handle the problem of cheating among agents. Rates are obtained cooperatively and the competition in the environment is not considered at all.

The work of Yu et al. [263], [264] and [265] describes a decentralized solution to reputation management. Agents are allowed to actively determine trust using reputation information they receive from other agents. This is a further effort towards the construction of a decentralized trust management system. The information each of the agents stores, reflects a

general value of the quality of the interaction with the partners. Each of the agents defines an upper and lower threshold for what they consider as trustworthy, unclear and untrustworthy agents. Then, using the saved information together with Dempster-Shafer theory of evidence, an agent can calculate the probability that a specific partner most probably is going to offer a service assigned to one of these groups. If the difference between the probability that the service belongs to the first and latest group is greater than a threshold for trustworthiness, the target agent is considered as a trusted one. Once again, only the most recent experiences with a specific partner are considered. When third parties' experience is considered, a similar graph to that of Schillo et al. [231] is used. If any of the acquaintances has any information on the target agent, they will send it to the interested agent, otherwise they send referrals to the other agents that can be queried to obtain either the needed information or further referrals. If the referrals that finally give the desired information are not far away to a depth limit in the chain, then the information will be taken into account.
The model does not offer any possibility for the simultaneous use of direct and third parties' experiences. If any information exists thanks to direct experiences that constitutes the only source of information that is considered to determine the trust of the target agent. Third parties' experience is considered only in the absence of direct experiences with the target agent.

The model presented by Sen and Sajja [235] is primarily concerned with the robustness of reputations that an agent receives. The agent selects those partners that have the highest reputation in a group of agents. The main contribution of the model is the capability of each agent to dynamically adjust the size of the group it is going to select a partner from, despite the existence of a set of cheaters within the population of agents. Cheaters are assumed to cheat consistently. To decide if a partner is good or not, the model uses the number of positive and negative answers received from recommenders. Knowing the number of recommenders and how many of them are cheaters, the model provides a mechanism to calculate how many agents should be queried to be sure that the likelihood of selecting a good partner lies upon a certain value. Direct experiences are used only as recommendations and not used in combination with third parties' experience in order to obtain a final reputation value.

In the work of Carter [97], the reputation of an agent is based on the degree of fulfillment of roles assigned to it by the society. If the society judges that they have met their roles, they are rewarded with a positive reputation, otherwise they are punished with a negative reputation. The authors identify five roles:

- Social information provider role - the degree of connectivity of an agent with its community (users of the society should regularly contribute new knowledge about their friends to the society). Every recommendation made by a user has a weight associated to it which indicates the strength of the recommendation.

- Interactivity role - the degree of usage of the system from a particular user, in comparison with others (users are expected to regularly use the system, otherwise without this participation it is assumed that the system becomes useless).

- Content provider role - the degree of affinity of the offered information to the real interests of the users (users should provide the society with knowledge objects that

reflect their own areas of expertise; users that create information agents related to their areas of expertise will produce higher quality content related to their interest than those who do not).

- Administrative feedback role - (users are expected to provide feedback information on the quality of the goods offered by the system).
- Longevity role - the degree of having a high average reputation (users should be encouraged to maintain a high reputation to promote the longevity of the system).

The user's overall reputation is calculated as a weighted aggregation of the degree of fulfillment of each of the above roles and the weights are entirely dependent on the specific society.

Maximilien et al. [193], [194] and [195] offer a multi-agent framework (WSAF, Web Services Agent Framework) coupled with a QoS ontology. Agents provide methods that let consumers set their QoS preferences and rank services. They establish a QoS ontology, however, there are no details about metrics where the consumers have to base on and do not consider how truthful providers in their QoS advertisements could be. It is not shown how agents scale with the increasing number of services in the environment and how agencies scale with the changing number of QoS properties. Furthermore, they evaluate only service providers from the consumer's point of view and do not offer any possibility for the providers to evaluate their consumers.

Trust in P2P Systems. Peer-to-Peer networks are networks in which all peers cooperate with each other to perform a critical function in a decentralized manner [200]. Usually, there is no centralized control authority in P2P systems and all peers are both consumers and providers of resources and share information and services directly without intermediary peers. Since peers are heterogeneous in providing services and do not have the same competence and reliability, considering trust is a necessity for such environments.
Damiani et al. [120] proposes the XRep protocol. It makes automatic voting using feedbacks of the users regarding the best host for a given resource possible.
The same idea is used by the authors in the P2PRep system [115], [119]. It is a set of protocols and algorithms for sharing reputation information with peers. It is a self-regulating system that uses a distributed polling algorithm by which resource requesters can assess the reliability of a resource offered by a participant before initiating the download from that specific participant.

PeerTrust [261] and [262] is another approach that aims to develop a trust mechanism for P2P networks. Peers can quantify and compare the trustworthiness of other peers and perform trusted interactions based on their past direct experiences without trusted third parties. The model defines a general trust metric that combines the total number of transaction a peer performs and the credibility of the feedback sources in addition to the feedback a peer receives through its transactions with other peers. A limitation of the model is that it has no mechanism for preventing dishonest peers from cheating via collusion, where a group of peers secretly agree or cooperate especially for illegal or deceitful purposes. In addition, this approach has no mechanism for filtering out and isolating dishonest peers from the reputation network. A peer gathers feedbacks from other peers regardless of their honesty. This practice

allows dishonest peers to damage and influence the feedback-based reputation network.

The EigenTrust [177], using PageRank algorithm [92], computes a global reputation value for each peer. It is intended to decrease the number of downloads of non-authentic files in a P2P file-sharing network. Reputation is simply the quality of a peer's uploads, where as quality, parameters like the successful upload of a file are implied. To every peer, a unique global trust value, based on the peer's history of uploads is assigned. By having peers use these global trust values to choose their partners from whom they download, the network effectively identifies malicious peers and isolates them from the network.
The most important design considerations, relevant to the reputation system are:

- The system should be *self-policing*.
- The system should *maintain anonymity*.
- The system should *not assign any profit to newcomers*.
- The system should have *minimal overhead*.
- The system should be *robust* in the face of malicious collectives of agents.

In the EigenTrust system, peers are assumed to be organized under a distributed hashtable (DHT). The reputation values for any given peer are managed by a unique set of "score managers" which are allocated using multiple hashes into the DHT space. The benefit of this approach is that there is no need for a central authority to map hashed node values into actual nodes. Before communicating with a node, the requesting node should access the score managers for the destination node. Similarly, once the file transfer has completed, those same score managers should be updated with the transfer status. However, the authors focus mainly on the mathematics aspects of trust calculation and do not provide any approach on how this information is going to be distributed within the system.

Wang et al. [257] and [256] use "Bayesian Networks" for enabling peers to develop trust and reputation, especially with respect to the competence and capability of peers to offer high-quality files and valuable recommendations in a P2P file sharing application. The model also computes a global reputation value for each peer. Every peer develops a naive Bayesian network for every other peer it has interacted with and modifies its corresponding Bayesian networks after each interaction. When a peer has no experience with another one, it can ask other peers to make recommendations for it. Such recommendations are partitioned in two groups, recommendations from trustworthy peers and recommendation from unknown peers and are combined by taking a weighted sum.

In [67], Aberer et al. distinguish trust among:

- provider peers,
- recommender peers offering recommendations on provider peers and
- recommender peers offering recommendations on other recommenders.

They consider the authenticity of a recommendation before it can be used. In their model, a negative feedback is stored as an evidential recommendation called a "complaint". The evidential recommendation mechanism works well against the attack from peers issuing fake recommendations without real interactions, however, it cannot detect and stop peers from lying in recommendations based on interactions that have really happened. Furthermore, since this model is entirely based on negative feedbacks, the peers do not have any possibility to develop a positive reputation.

Poblano [106] is a JXTA-based effort to establish a reputation based decentralized trust model. Specifically, Poblano is used as the trust model for distributing signed certificates among peers in JXTA. The trust spectrum neither requires nor prohibits the presence of a PKI (Public Key Infrastructure). A peer is connected to at least one peer group, which is a dynamic set of peers that have agreed upon a common set of policies and services. Peer group's membership is motivated by keyword interest.
This approach is used to perform reputation guided searching or to securely distribute signed certificates among peers and as such, it is based on recommendations. For calculating the cooperation threshold between two peers, a user-centric approach was chosen. The user interaction is required to tune the software and to make choices. Access control based on direct observations is not the main concern of this work.
The major drawback of Poblano is that the peers do not consider the latest information available to them for discovering compromised peers.

Trust Management in the Semantic Web. The Semantic Web is also a large, unsupervised system to which everyone may offer his contribution. Trust decisions in such environments can be a transitive process, where trusting one piece of information or information source requires trusting another associated source. Much research has focused on authentication of resources (work on digital signatures and public keys), however, just because a person can confirm the source of documents, does not also mean that the content of the offered documents has to be trusted. Confidence in the source or author of a contribution is important, but trust and reputation in this case could help diminish the uncertainties.
One of the early approaches of Stweart [248] and later on of Stewart et al. in [249], deals with the trust transfer between hyperlinks on the Web. They examine how much trust, in the context of a consumer trusting a business for purchasing a product, is transferred from a trusted Web resource to an unevaluated one. The transfer is evaluated through the combination of different types of links, types of resources and types of trust in the known sources.

Advogato [58], provides a trust metric, called "Advogato maximum flow", for discovering which users are trusted by members of an on-line community and which are not. The authors compute the maximum network flow over a web of trust to find trust between any pair of entities. An advantage of this approach is that it is very robust to noise and even attacks altering the given web of trust. However, trust is computed by a centralized community server and there always exist some highly trusted users. Furthermore, the model does not provide support for weighted trust relationships.

Golbeck et al. [152] and [153], describe how trust/reputation can be applied to a person for a specific subject area. A trustor trusts a trustee in relation to a certain area by such a

degree or that the trustee has a certain reputation in a specific area. The trust values are measured on a scale from one to ten where one means absolute distrust and ten means absolute trust. In their model, participants *assign* trust values to others. Trust is then calculated by a transitivity function, which considers the trust along the hops down the network path. Using the same principles, in [192], it is dealt with the problem of controversial (trusted and distrusted) participants. The key contribution of their work is the evidence that the globally computed trust value in a web of trust, for a controversial user may not be as accurate as a locally computed value due to the global disagreement on trust for that user. This work addresses the use of trust within those systems where the set of commonly rated items between users is sparse (i.e. Epinion community [11]). That situation leads to a breakdown in correlation-based recommender system algorithms and their work explores how incorporating even simple binary trust relationships can increase the coverage and thus the number of recommendations that can be made.

In general, trust management models for semantic web do not consider context and as a result do not differentiate between "context specific trust". Ding et al. [128] try to present a method for computing trust, considering also the domain of knowledge. They make use of the trust of participants in the ability of the others to recommend based also in their similarity. As a result, they offer a data model that computes trust as:

- trust in the domain knowledge of participants,
- trust in their ability to recommend others,
- more participants trusting the others similarly and
- more participants being trusted similarly from the others.

However, the model bears a high level of abstraction regarding the notion of trust.

4.5 Trust in Grid Computing Environments

In Grid environments, trust management was first discussed by Azzedin et al. [79], [81] and [80], who make an important contribution defining the notion of trust through *identity trust* and *behavior trust* and also separating the Grid domain into a *client domain* and a *resource domain*. They have studied the importance of trust in Grid environments and have shown how the computing performance in Grid can be improved by using the concept of trust and avoiding the large computational overhead incurred by the security infrastructure.
The work on trust management in Grid environments deals with only one of these dimensions of trust.

4.5.1 Identity-Based Trust Models

The Grid Security Infrastructure (GSI) [18] from the Globus Alliance uses the X.509 certificates as its authentication mechanism for security. Authentication is one of the mechanisms helpful in implementing certification trust. In this PKI, highly trusted participants known as certificate authorities (CA), issue X.509 certificates where essentially a unique identity name and the public key of an entity are bound through the digital signature of that CA. One of

the challenges encountered in key management include the need of users of having different credentials, according to the roles they play and projects they take part, thus, different CAs need to be trusted. While PKI could handle this situation by signing the same public key into several different certificates, in practice the user may end up with numerous key pairs to manage. To link these different identities, the notion of federated identities has been developed, as shown by Linn et al. [186].
GSI provides necessary mechanisms needed for authentication, but does not handle all the security management issues and especially trust management issues. Although it uses the PKI infrastructure to establish the identity of other collaborators, this identity does not provide any information about the likely behavior of the participants.
Another proposal for handling authorization in Grids is considered in the Globus Toolkit Gridmap file [20]. This file holds a list of the authenticated distinguished names of the Grid users and the equivalent local user account names that they are to be mapped into. Access control to a resource is then left up to the local operating system and application access control mechanisms. This approach neither allows the local resource administrator to set a policy for who is allowed to do what, nor does it minimize his/her workload.
The Community Authorization Service (CAS) [219] was the next approach by the Globus team to improve upon the manageability of user authorization. CAS allows resource owners to grant access to a portion of his/her resource to a VO and then let the community determine who can use this allocation. The resource owner thus partially delegates the allocation of authorization rights to the community. This is achieved by having a CAS server, which acts as a trusted intermediary between users and resources. Users first contact the CAS asking for permission to use a Grid resource. The CAS consults its policy (which specifies who has permission to do what on which resources) and if granted, returns a digitally self-signed capability to the user optionally containing policy details about what the user is allowed to do. The user then contacts the resource and presents this capability. The resource checks that the capability is signed by a known and trusted CAS and if so maps the CAS's distinguished name into a local user account name via the Gridmap file.

Chang et al. [104] use certified code as an upholder of trust. The authors deal with a simple notion of trust, considering only the correct running of some code, on behalf of the clients, on the remote providers. The intrinsic properties of native code that is going to be run are examined. It serves to determine if the code tries to perform any illegal operation either intentionally or non-intentionally. Once the testing of the code is done, it is assumed to be trustworthy. The authors claim that the certified code can be run even in trustless environments without considering any additional trust mechanism.

Hwang et al. [161] discuss the problem of propagation of trust in Grid computing using PKI based trust models. They make use of bridge CA certification model, which interconnects various PKI islands by a central authority of cross certification. Each PKI domain sets up a cross-certificate with the central bridge CA. Then a trust path is formed from a sequence of intermediate CAs. Once trusting a bridge CA, one must trust all other certificates in the environment. The model is suitable only for investigating identity trust in Grid environments although no protection against certificate theft or impersonation are provided.

Ma et al. [190], propose a new type of mobile process in Grids, *Grid traveler*. It is able

to move across VOs in order to coordinate the use of resources and access control. For this purpose, the authors propose a security infrastructure, G-Pass, for managing credentials of the travelers. This infrastructure implements a kind of trust model that supports simple credential verification and transfer. An important feature of this infrastructure is the inability of the holders of such "passports" to manipulate the contained information. Each page is defined as a contract, on which at least two identities are required to provide a digital signature and to claim their responsibilities.

In [252], Tie-Yan et al. try to aid the process of authentication. They propose a model consisting of two tiers. In the upper tier, trust among different virtual organizations belonging to different Grid domains is managed and in the lower tier is managed the trust among participants that belong to the same Grid domain but to different virtual organizations. No information regarding the notion of trust assigned to identities of participants and VOs is given. Furthermore, no means for protection against malicious activities are offered.

These trust mechanisms do not consider *behavior trust* and thus, the definition for trust, as presented in the previous chapter, is partially covered. No means for monitoring trust relationships are offered. In addition, these trust models and trust management applications do not recognize the need for participants to learn from past experiences and the experiences of others in the environment in order to dynamically update their trust levels [154].

4.5.2 Behavior-Based Trust Models

Azzedin et al. [79], [81] and [80] present a formal definition of behavior trust. Several aspects of trust values are considered as part of their model:

- The trust values decay with time - as time passes, if no more collaborations have found place among participants, trust decays.
- Trust relationships are based on a weighted combination of the direct relationship between domains as well as on the global reputation of the domains.
- The trust model should stimulate organizations to sanction entities who are not behaving consistently in the Grid environment and who break trust relations.

Behavior trust in this model is limited to a general abusive or abnormal notion of behavior of the participants during the interaction and no real metrics for trust are offered.

Alunkal et al. [74] propose to build an infrastructure called "Grid Eigentrust" that addresses reputation-based service selection in Grid environments. They use a hierarchical model in which participants are initially connected to institutions which then form a VO. The trust algorithm that defines these trust tiers is the major contribution of the work. They conclude with the realization of a "Reputation Service", however, without providing mechanisms that automatically can update trust values. Furthermore, they do not address the authorization/authentication of Grid users to the services.

In [184], Lin et al. use the belief, disbelief and uncertainty to weight the trustworthiness of the collaborating parties. The authors deal with a general notion of behavior trust that

is established before interaction takes place among participants. They develop a trust management architecture for addressing security issues in Grid environments based on subjective logic. They use the belief, disbelief and uncertainty to weight the trustworthiness of the collaborating parties.
Identity trust, although considered in the model, is implied to belong only to the authentication process, without offering any possibility to measure it. The authors deal with a general notion of behavior trust. Trust information from direct and recommenders' sources are used.

Zhao et al. [270] propose the notion of *trust-based scheduling* for peer-to-peer Grids. Using either a *quiz* or *replication* strategy they try to identify non-correct responses from the partners. Thus the behavior of the participants in the environment is identified with the accuracy of the results coming from the "provider peers" while it is assumed that every peer in the system has a persistent identity. When using replication, a similar model to Seti@Home [46] is considered. The validity of the computed results is checked by letting different participants work on the same data unit. At the end, if more than the half of the incoming results is the same, the correct response is established. This technique results in a significant increase of the total computational costs to solve a particular problem, since the overall workload is multiplied by the replication factor used to reassign subtasks. If there is a service charge collected by a service provider for every processed work unit, this also leads to a considerable increase of the overall cost. Even the usage of particular test challenge for which the consumer knows the correct result could sometime turn out as not reliable, since a malicious provider may be able to guess the correct answers to the set of challenges, as the computational resources of the consumer for test preparation may be very limited.

A Grid environments as a multi-agent system is considered by Dyson et al. [133]. In this environment the following type of participants take place:

- user agents, which use the resources that are available to them.
- resource agents, which offer resource access have the goal of ensuring that the resources under their management are utilized to their maximum capacity.
- negotiating agents, which could be either user agents or resource agents. They match potential resources with the requests that are submitted from the user agent.

A particularity of this work is the evaluation of the user agents as *"pessimistic"* or *"optimistic"* ones according to their disposition on assigning initial trust values for the resource agents. Over time, trust values related to past experiences will become inaccurate and outdated. However, this model deals with a general notion of behavior trust and no trust metrics are offered. This model bears also a great level of abstraction regarding the notion of trust in general and how trust is managed and updated from participants.

Finally, Patel et al. [217], develop a probabilistic approach for managing behavior trust also in agent-like Grid systems. They do concentrate more on the accuracy of the trust values coming from third parties (third parties' experience). A participant is considered as trustworthy only if it has a high probability of fulfilling its obligations during the interaction.
In a later work, Teacy et al. in the TRAVOS model [250] and [251] try to cope with inaccurate reputation sources. They use a similar reputation representation to that of Beta Reputation

System [163]. A distinctive feature of their model is that recommendations coming from inaccurate sources are not given the same importance as those received from accurate sources. In order to mitigate the effects of deceptive advices, TRAVOS scales opinions by a factor derived from comparing past opinions with the outcome of interactions. In their work, they suppose that the agents do not change their behavior and this can be seen as a big handicap for this model.

4.6 Other Approaches to Trust Management - TCPA Initiative

While social-oriented trust models pertain to the reasoning and information gathering ability of participants, there is a second approach to building trust concerning the implementation of new technologies instrumented for security (Trusted Computed Initiative [111], [110], [112], [113], [114]). Through this technology, it is expected that participants interact with each other only if they can be considered as *"trusted"*, where according to the definition at Wikipedia [52], in technical sense, *"trusted"* does not necessarily mean the same as *"trustworthy"* from a participant's perspective. Rather, it means that a certain participant can be trusted more fully to follow its intended aim without any deviations from the expected behavior.
This approach takes a distributed-system-wide approach to the provisions of integrity protection for resources. The following notions are at the core of the technology:

- Trusted Platform Module (TPM) - a hardware module for conformed operation and secure storage. It is designed to perform computations which cannot be subverted by the platform owner, including the system administrator. These computations involve also some public key cryptographic operations (decryption and digital signature generation using a private key in the TPM), system status measurement and secure storage. This module is going to be installed at every participant's site

- Core Root of Trust for Measurement (CRTM) - at the participant's site, during the boot time, the TPM measures the system's data integrity status. The measurement starts from the integrity of BIOS, then that of the operating system and finally to applications. With CRTM, is intended to establish a desired environment by loading only well behaved systems.

- Root of Trust for Storage - the measured integrity of an executable is represented by a cryptographic checksum of the executable. This is then securely stored in a TPM. The TPM component called Platform Configuration Register (PCR) holds this data in an accumulative formulation. The stored environment status is maintained until system reboot.

- Remote Platform Attestation - cryptographic challenge-response mechanisms are going to be used. A participant can then evaluate whether its remote partners have the desired behavior. Remote platform attestation is the most significant and the most innovation element of this technology.

The challenge, for the Grid is to find the means of integration with this technology which has to support the significant components of Grid infrastructure in the most seamless manner as

possible. It is necessary to support the whole lifecycle, from provisioning and commissioning of Grid nodes, software deployment, authorization of participants and so on. Support for fine-grained mandatory access control will also require integration with the authorization approaches previously discussed.
This approach although still under development has already encountered a strong contradiction [1]. Furthermore, the implementation made from Microsoft on Windows Vista [55], showed itself as vulnerable by Rutkowska [227].

4.7 Discussion

Trust management remains a significant area of research in different domains of computer science. The various approaches can be divided into two main categories:

- approaches focused on trust aspects of a security infrastructure in particular with regards to the authentication and authorization of users
- frameworks for trust management focused on the analysis and quantification of trust and trust services.

The former are relatively well understood because relate to PKI infrastructures. However, these models define specific situations for their trust, e.g. X.509 specifies trust only in context of creating reliable certificates, PGP for key introduction etc. They make use of the term "trusted", but did not explicitly define what being "trusted" means and do not relate to the discussed trust requirements. The same problems exist also for the models that use policies for managing access and information rights for "trusted" participants.
The second category has also been subject of a great number of studies, however, apart from agreeing on the importance of trust, little agreement on what trust really is, how trust can be characterized and how can it be measured has been achieved. The following aspects are agreed into the various studies on trust:

- relationship to different sources of information that influence its value,
- orientation toward different contexts that include the activities being performed, the parties engaged in the interaction as well as other contextual elements of the transactions,
- relationship to different characteristics of the parties involved in a collaboration or the activities being performed such as their competence, honesty, correctness, etc and
- its quantification, meaning that trust is measurable. Unfortunately, no consensus has been reached yet on the desired metrics for its quantification.

Many of the proposed approaches differ significantly in their definitions and computational methods for trust. Thus, it is not surprising to find that many of the above models, although claiming to handle various aspects of trust, have failed to define what trust is. Trust is mainly represented as a subjective belief in another participant's honesty, where honesty is primarily associated to the accuracy of the results coming from the collaboration partners.
Another common flaw with the majority of the proposed approaches is that they are used to identify a static form of trust. Trust is mostly evaluated only at the start of a collaboration

considering only direct experiences and information from third parties. Using only these information sources, however, imposes several addition concerns that deal with the poor direct experiences or with the subjectivism of third parties' opinion.
The majority represent trust as the probability of a binary event, that is, the probability that a partner will cooperate or defect. However, by modeling a partner's possible actions simply as cooperation or defection, they ignore the effects that quality of service provided by that partner may have on the assessment of its trust.
The trust formation phase is characterized by the leap from ignorance to the belief in another's (un-)trustworthiness, taken with caution and within a certain time-span. This phase, however, is absent from all of the models surveyed. Furthermore, none of the models has properly considered the "dilemma" of new participants entering an already well established environment. Such participants may never be given the opportunity to have their trustworthiness assessed. Sometimes, distrust is considered as the alternative to ignorance. In general, the models that support distrust can be considered as *pessimistic* as none of them specifically give untrustworthy participants the chance to prove the contrary through some additional experience.

The presented trust models for Grid environments bear also a lot of limitations. Most of them make use only on part of the trust sources, more precisely only on direct and third parties' experience. To the participants, the possibility to choose among different trust sources when gathering trust information on the future collaboration partners is not offered. All the models take into consideration only one type from the considered approaches; either identity or behavior trust. A collective characteristic is the lack of mutual verification of parties involved in an interaction. Although in the trust models that manage identity trust of the participants, parties involved in an interaction make a mutual verification of their identities at the beginning of the collaboration, there is still need for a continuous mutual verification during the collaboration. This verification affects the behaviors of the parties involved.
None of the approaches in the field of Grid computing analyzes and classifies the behavior of the participants to conform to the real social meaning. There is still no specification on what the behavior of Grid participants really is and few metrics regarding the measurement of behavior trust are offered. Only a general notion for the behavior of the participants is specified, leaving out different contexts that determine the behavior of Grid participants.

Trust management should be dynamic, having a notion of learning and adaptation. It should be able to adapt to the changing conditions of the environment in which the trust decision was made. Human users should also be able to incorporate their own preferences in the decision-making process. Intentions and the behavior of the participants in the environments are subject to changes and evolution and thus, there is a need to monitor trust relationships to determine whether the criteria on which they are based still apply. This could also involve the process of keeping track of the activities of the partners involved during the collaboration and of determining the necessary actions needed when any of the participants misuses the trust of their partners.

4.8 Summary

In this chapter, the related work done for modeling and managing trust in some areas of computer science was presented. First, an overview of the models for identity trust was given. In none of the presented approaches there was a relationship to behavior trust and thus as stand-alone, these models are not suitable to assess the trust of the participants in the Grid environments. Second, the current behavior-based trust models were presented. They were divided into:

- models that consider only direct experiences and
- models that consider either third parties' experiences only or hybrid models where the direct and third parties' experiences are considered as well.

Third, a discussion on the collective features and limitations of the presented approaches was done.

In the next chapter, the trust model for Grid environments will be presented.

Chapter 5

Towards A New Approach to Trust in Grid Environments

"You may be deceived if you trust too much,
but you will live in torment if you don't trust enough."
Frank Crane

"Distrust itself is very expensive"
based on a quote of Ralph Waldo Emerson

5.1 Introduction

There are a number of ways that a Grid participant can establish trust with its counterparts. Firstly, it can interact with the target participant(s) and learn their behavior over a number of interactions. In this case, the participant reasons about the outcome of the direct interactions with others. When starting an interaction with a new participant, i.e. no information on previous behavior exists at all, it can use its beliefs about different characteristics of these interaction partners and reason about these beliefs in order to decide how much trust should be put in each of them. Secondly, the participant could ask others in the environment about their experiences with the target participant(s). If sufficient information is obtained and if this information can be trusted, the participant can reliably choose its interaction partners.

In the previous chapter it was shown that the presented approaches to trust, especially in Grid domain bear limitations. The main limitations identified are:

- only part of the trust sources are used,
- either identity or behavior trust are considered,
- no definition of identity trust exists,
- there exists an abstract and general notion of behavior and behavior trust; participants are either "collaborative" or "defective",
- lack of continuous mutual verification of the partners involved in an interaction,

- still uncertainties regarding the "real" behavior of participants (reflection of their intentions; malicious and mediocre behavior is favored).

As a result, there is a need for a flexible trust model whose properties reflect the requirements of Grid applications and the preferences and needs of their users in a heterogeneous, dynamic, uncertain and vulnerable Grid environment.
The work presented in this chapter is designed to fulfill this goal for Grid environments, but at the same time to be generic enough to be considered in many other similar domains.

The notion of trust used follows the statement of the chapter 3:

"Trust is the extent to which every participant in a Grid environment, in a specific moment of time, with an evidence of relative security regarding the identity and the behavior of their counterparts, is willing to interact with them, even though unexpected negative outcomes could result from the entire collaboration process."

A high degree of trust in a participant would mean that it is likely to be chosen as an interaction partner. Conversely, a low degree of trust would suggest that the participant is no more selectable, especially in the case when other, more trusted interaction partners are available. In this way, the trust model aims to guide a participant's decision making process regarding how, when and who to interact with. However, in order to do so, the trust management model initially requires participants to gather some knowledge about their counterparts' characteristics or "behavior". This can be achieved in many different ways: through inferences drawn from the outcomes of multiple direct interactions with these partners or through indirect information provided by others in the environment that have had similar experiences. The direct interactions make it possible to consider methods by which participants can learn or evolve better strategies to deal with honest and dishonest participants such that payoffs are maximized in the long run. It makes up the trust information that is going to be shared with others in the environment. For every participant, in order to consider third parties' experience, the ability to develop methods to reliably acquire and reason about the information gathered from them is offered.
The overall decision whether to trust an interaction partner or not may be affected by other non-functional aspects that cannot be generally determined for every possible situation, but are rather considered to be under the control of the user(s) when requesting such a decision. In addition, while the basic functionalities of two Grid applications could be similar, differences in application behavior could be caused by different domain specific trust requirements. Therefore, in this trust management system, flexible and easy to use components that can be configured to the specific needs of a user on a per case basis are offered.
In the following, a trust model that can be classified at the individual level as learning based, reputation based and to a certain point socio-cognitive based is proposed.

The trust model includes identity and behavior trust of the interaction partners and considers different sources to calculate the portion of trust (overall or partial trust value) for an interaction partner. Participants, independently of the role they are playing for the moment, have the possibility to monitor each others' behavior during the entire collaboration.
Finally, a system architecture for collecting and managing multidimensional trust values is

proposed. It consists of two main components, a trust engine and a verification engine. The trust engine manages trust values and offers partner discovery and rating functionality to higher level applications. The verification engine handles the verification of the behavior of the parties during the collaboration and generates the necessary feedback for the trust engine regarding the partner. The proposed system architecture can be configured to the domain specific trust requirements by the use of several separate trust profiles covering the entire lifecycle of trust establishment and management. Thus, this model is considered to be generic enough to support social-oriented trust management within Grid computing and other similar scenarios.

5.2 Key Concepts

In the following subsections, a summary of the key concepts of the trust model, some of which have already been treated in the previous chapters and also been published by Papalilo et al. in [212], [216] and [213], is presented.

5.2.1 Decentralization

In an interview, Patti Maes [221], declared:
"We always think of intelligence as a centralized thing. We view even our own consciousness as centralized. It's called the homunculus metaphor - that there's a little person inside our brain running things. But it's more likely that intelligence is decentralized and distributed."

Instead of delegating the problem of determining trust to some centralized node, a distributed approach which involves decentralization and collaboration is proposed. "Decentralization" of the notion of trust is an important step to be taken in a Grid environment. Each of the participants should be able to decide on its own regarding to its policies and actions. Furthermore, the structure of relationships between participants in the environment must be allowed to evolve naturally. Although in this case more responsibility and expertise is assigned to an entity, at least each of them has a chance to manage its own trust relationships.
Decentralization is also considered to improve the management of trust (Blaze et al. [88]).

5.2.2 Participants

Participants considered in this work include:

Grid End User(s)[1]. Grid end users are those application users who wish to access or use Grid resources and services in Grid-enabled infrastructures for solving their problems belonging to different domains, from media sciences (Ewerth et al. [134]), medical video and image analysis (Amendolia et al. [75]), problem-solving environments that help set up parameter study experiments (Abramson et al. [68]), mathematical packages augmented with calls to network-enabled solvers (Casanova et al. [98] and Czyzyk et al. [118]), consulting industry (IBM [155]) and so on.
For the end users, building or modifying Grid applications remains a difficult and time-consuming task and thus, excluded. They typically place their requirements on the Grid-

[1]Referred now on as "user(s)".

enabled tools, in terms of reliability, predictability, confidentiality, usability, costs, performance, security and trust.
The construction of applications that can meet these requirements in Grid environments represents a major challenge (Foster et al. [140]).

Grid Consumer(s)[2]. Grid consumers take care of users' requests. They make use of Grid and local system services, discovery mechanisms, scheduling and management tools. Consumers interact with providers on behalf of users for executing their requests. They are in charge of managing the entire collaboration process and hand the outcome to the users.
A user could contact a consumer either directly or through a Grid portal like Globus-COG [7], or GPDK [24].
Consumers' requirements normally reflect the users' requirements.

Grid Provider(s)[3]. Grid providers constitute the environments that host access to the services or storage resources. Providers aim to maximize the returns on offered services. They make use of management tools, Grid and local system services.
Providers' requirements normally reflect their owners' requirements, i.e. profit, transparency, security and trust.

Certification Authorities (CAs). A CA could be a third party dedicated to the distribution of certificates (i.e. Grid Certification Authorities [21]), or every other participant in the environment. CAs are used to certify the link between the public key and the subject in the GSI-like certificate [18] giving a possibility to identify Grid participants.

5.2.3 Trust Metrics

Trust measurements regarding identity and behavior are reduced to the normalization of the measured values in the range $[0, 1]$.
Probabilities are used as parameters of subjective belief (denoted as confidence levels) to determine the trust values. The probability for a successful future interaction among partners is considered as closely related to the mutual trust values the partners assign to each other during the collaboration.
These values vary in the range of $[0, 1] \subset \Re$, where:

- 0 means that the other partner is not trusted at all (the condition of "distrust", resulting from the partner's deviating/other than expected behavior during past collaborations), or there are uncertainties due to the lack of information (the condition of "ignorance"). Assigning a trust value that equals 0 to a collaboration partner when no information on its behavior exist, expresses not only the "ignorance" but also the prejudiced "distrust" regarding the behavior of a unknown partner during the collaboration.
- 1 means that it can be fully trusted and gives certainties on the success of the interaction that is going to take place.

These confidence parameters are an evidence of past experiences and reflect also the risk associated and the expectations for the future.

[2]Referred now on as "consumer(s)".
[3]Referred now on as "provider(s)".

5.2.4 A Model for Small Worlds in Grids

It is believed that almost any pair of participants in collaborative environments (real world included) can be connected to one another by a short chain of intermediate acquaintances. This phenomenon could be seen as valuable for bringing some order to the Grid environments as considered in this thesis.
In the following, the "small world" phenomenon and a view on how to introduce this phenomenon in Grid environments are presented.

The "Small-World" Phenomenon. The phenomenon has its roots in experiments performed by the social psychologist Stanley Milgram in the late 1960s [199]. The aim of his experiments was to trace out short paths through the social network of the United States. He asked participants of the experiment to forward a letter to a "target person" living near Boston, with the restriction that each participant could advance the letter only by forwarding it to a single acquaintance that the current holder knew on a first-name basis. The letters were distributed over a random selection of people over a certain territory. The strategy followed by most of the people was to pass their letter to someone they presumed was more likely to know the person to whom the letter was ultimately addressed.
Milgram kept track of the letters and the demographic characteristics of their handlers through receiving intermediary reports on the letters' reception. A reasonable number of letters eventually reached their destination, with a median chain length of about six[4]. Milgram concluded that six was therefore the average number of acquaintances separating any two people in the entire world.
Further work conducted by Watts and Strogatz [259], introduced the "small-world network" concept which is based on "six degrees of separation". They have shown that introducing just a few distant connections in a regular graph is enough to drastically cut the step-length and yet still retain high levels of clustering. In other words, such "semi-random" networks are very likely to show the "small-world" effect, with apparently unrelated clusters joined to others via few steps. Watts and Strogatz made use of three real-life networks: film actors, the electric Grid of the western US and the nervous system of a nematode worm. Their model lies between random and regular networks.

Another concept, used for describing "collaborative distances", is that of the "Erdös number"[5] [13]. Paul Erdös has Erdös number 0. Every person that has a joint publication with him has the Erdös number 1, others that have a joint publication with them have Erdös number 2 and so on. Thus the Erdös number of every person is just the distance from that person to Paul Erdös in the collaboration graph (in which two authors are joined by an edge if they have a joint publication).
Amongst all mathematicians who have a finite Erdös number, the numbers range up to 15, the median is 5, the average Erdös number is 4.65 (according to the Erdös Number Project); and almost everyone with a finite Erdös number has a number less than 8 [12].
Notions similar to the Erdös number are used for defining the Oracle of Bacon which links all co-actors of Kevin Bacon [39], the Kasparov number which links all co-players of the chess

[4]This situation has been labeled and referred to as "six degrees of separation".

[5]Honoring the mathematician Paul Erdös, one of the most prolific writers of mathematical papers; had 485 co-authors.

player Garry Kasparov [31], the Black Sabbath Game which links musicians in rock bands [3], the Oracle of Baseball which links baseball players [40] and so on.

Modeling the "Small-Worlds" Phenomenon in Grids. Milgram's experiment [199] led to two main discoveries:

- first, he proved the existence of short paths in social and collaborative environments,
- the second deals with the fact that people in society, with knowledge of only their own personal acquaintances, were collectively able to forward the letter to a distant target so quickly.

Similar conclusions came out from the work of Watts and Strogatz [259].

However, in this thesis, use of a much simpler model of "small-worlds" for Grid environments is made. Two types of "collaboration graphs" are considered:

- certification graph (Fig. 5.1) - links the participants on basis of the authorities that issued their certificates

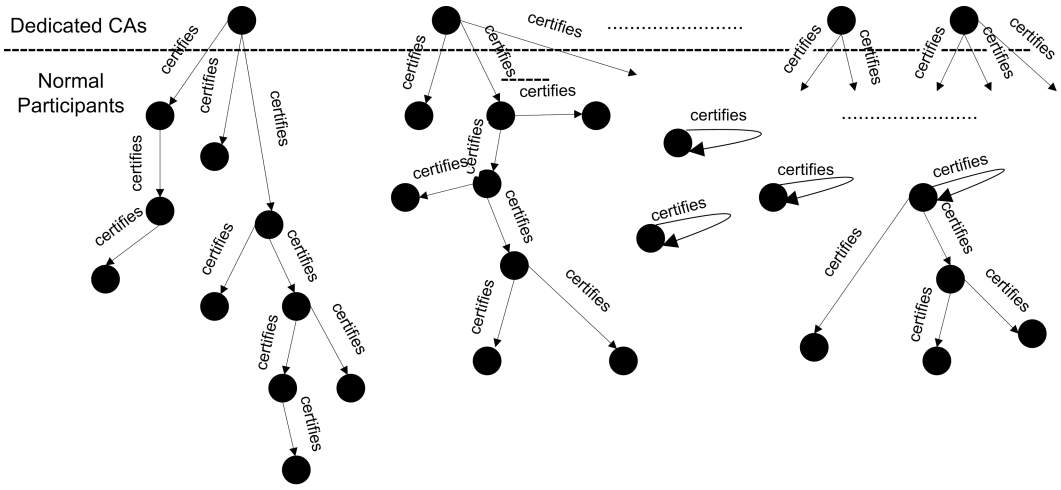


Figure 5.1: Certification Graph.

- collaboration graph (Fig. 5.2) - links the participants on the basis of previous direct collaborations (either for exchanging recommendations or execute tasks and gather responses).

Similarly to the Erdös number the following can be created:

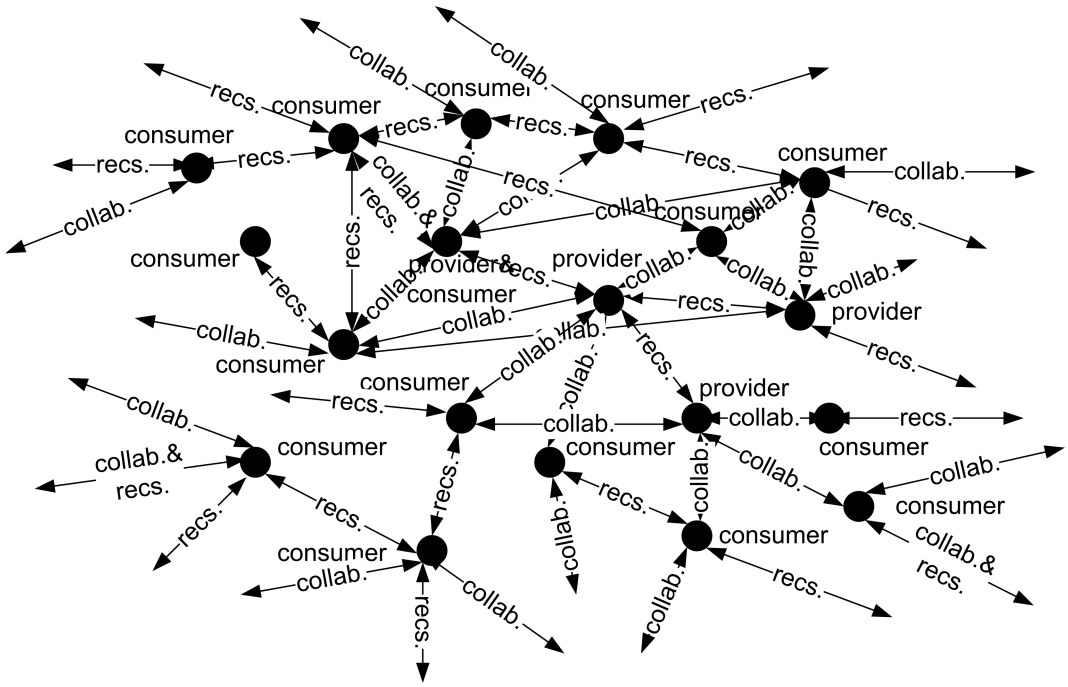


Figure 5.2: Collaboration Graph.

- the "Oracle of Certification" - every participant builds a "certification graph" centered at its own CA. "Oracle of Certification" defines the degree of separation of a target partner from this participant as the path length of the shortest certificate chain from the participant's CA to its partner. Any partner who has no path to the participant's CA is said to have an infinite relationship with the centre of this "certification graph" and thus ∞ is assigned. A relationship is considered to be of first order (i.e. 1 assigned) between trusted and dedicated CAs such as the one in [21].
- the "Oracle of Collaboration" - every participant builds a "collaboration graph" staying at the centre of the graph. "Oracle of Collaboration" defines the degree of separation of a target partner from this participant as the path length of the shortest collaboration chain from the participant to its partner.

Figure 5.3 shows the relationship of every participant to the others in the environment. "Oracle of Certification" and "Oracle of Collaboration" are going to be used by a trustor in the trust management model during the calculation of Identity Trust and Behavior Trust of its trustees.

5.2.5 Trust to Identity and Identity Trust

Each of the participants in the environment, before starting to interact with its partners needs to have some certainties about their identity, i.e. that they are who they claim to be. This subsection discusses the identification scheme used by this trust management model for

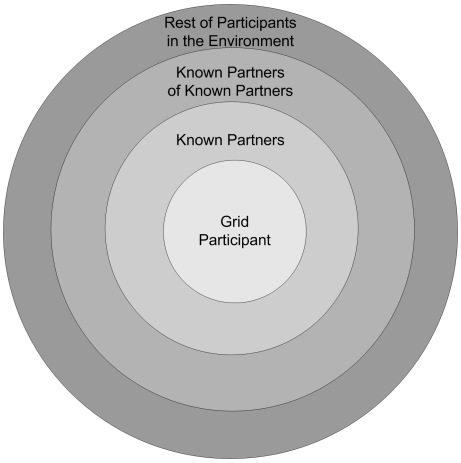


Figure 5.3: Relationship between Grid participants.

the participants in the environment together with problems related to identity and authentication of participants in Grids and proposals on how to eliminate or minimize them.

Identity. Every participant[6] in the environment is identified through a certificate. The authentication infrastructure implied here is similar to the authentication infrastructure adopted by Globus [18].

The four primary pieces of information included in a certification are:

- A subject name, which identifies the person or object that the certificate represents.
- The public key belonging to the subject.
- The identity of a Certificate Authority (CA) that has signed the certificate to certify that the public key and the identity both belong to the subject.
- The digital signature of the named CA.

Besides the dedicated third parties, every other participant in the environment is also entitled to release certificates either for itself or others that apply for it. An overview of the certification scenarios is presented in figure 5.2.

The identity of the participant in this model is considered to be much more than a simple certificate. Every participant can be viewed as a collection of physical attributes as well as a set of knowledge, behavior, possessions and history (Fig. 5.4). All these parameters can be described as *identification factors*[7].

The identification factors can be further organized under:

[6]With the term "participant", it is refered to consumers and providers. End-users deal principally with consumers which interact with providers on users' behalf. At the end of the thesis, a possible enhancement of the trust model is presented, where among others, the trust between users and consumers is considered.

[7]The term was taken from the book Network Security Illustrated [245].

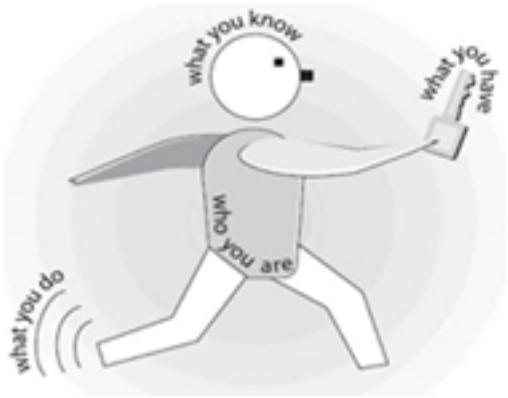


Figure 5.4: Identifying a participant in the environment (illustration from book "Network Security Illustrated" [245]).

- ***What a participant is*** - refers to *personal attributes* of every single participant. Examples of these traits include hardware and software peculiarities of the participant (i.e. operating system, hardware in use, network physical address, IP address, etc). Part of these attributes or a combination of them is extremely difficult to duplicate and very specific to a single participant. These attributes have to be included in the identification process.

- ***What a participant does*** - refers to *unique patterns of behavior* that this participant manifests during the collaboration with others. As such, quantitative QoS requirements[8] can be mentioned. However, these characteristics are far less consistent than personal physical attributes. Accurate methods of investigation are needed in this case in order to differ among normal behavior and different types of anomalies (malicious behavior included).

- ***What a participant knows*** - refers to *specific knowledge* that the participants have gathered (locally stored) during past collaborations with others. In Grid environments, knowledge of participants could be expressed in terms of personal history details (i.e. participants it collaborated with, their behavior during the collaboration, collaboration start and end time, etc).

- ***What a participant has*** - refers to *specific information* in possession of that participant. Such information is the private key assigned to the target participant from its CA. However, since this kind of "possession" could be vulnerable (can be stolen or counterfeit), additional elements need to be considered in order to increase trust to identity and when required, the reliability to the confidentiality of the communication. An example could be the generation of session key(s) that the parties use for encrypting the communication among them during the collaboration.

[8]discussed previously in chapter 3, under quantitative QoS requirements, parameters like availability, accessibility, accuracy, processing time, etc., are implied.

A fraudulent participant can fake any of the four identification factors, but faking more than one factor simultaneously is a significantly harder and expensive task.

The proposed solutions help making the identification process more secure, but could never replace the security layer. It is the first and most important layer that protects a participant from all kinds of threats and attacks in the environment. To that extent, the security layer itself can be considered as the "*fifth factor*" [245].

Discussion on Problems Related to Identity and Authentication of Participants in Grids. Having such an infrastructure for distributing digital identities comes out to be very convenient for the participants in the environment. The problems arise once someone has fraudulent (malicious)[9] intentions. In such a case the following problems could be observed:

- First, such a participant could apply and obtain different certificates from the same dedicated CA, from other dedicated CAs, from other participants in the environment or from itself. Thus a participant has the possibility to obtain more than one identity in the environment. As a result, it will be very difficult to discover and punish a fraudulent participant.
- Second, following the same reasoning, a participant could easily change its identity (apply for a new one and enter the environment with this one) in case others have discovered their intentions. Even in this case, a participant could never be punished for all wrongdoing.
- Third, every participant is (and should be) responsible only for the security of its own sensitive information (certificates, private keys, passwords, etc.). However, since the security measures differ from one participant to the other, it is to be expected that vulnerabilities exist in many of them. This allows fraudulent participants to easily impersonate (Lenstra et al. [181]) others in the environment through identity theft or duplication of sensitive personal information (identity counterfeit).

All these problems exist because of:

- ease of obtaining certificates,
- ease of duplication,
- difficulties on detecting such "counterfeit" information (Van Oorschot et al. in [175]),
- independence of new credentials. If existing credential information is used by an impersonator to obtain new credentials, the latter are in one sense "owned" by the impersonator, and usually no information flows back to the original credential owner immediately,

[9]The two primary types of adversaries in Grid environments, able to put others in the environment at risk are fraudulent and malicious participants. They are primarily distinguished by their goals in the environment. Fraudulent participants wish to have a considerable profit for their mediocre contribution, achieve a better "social" position in the environment to the detriment of the other participants or "live on others" (e.g. a consumer who is not going to pay at all or only in part for the services it got). The goal of malicious participants on the other hand, is to cause harm to either specific targeted participants in the environment or to the environment as a whole.

- ease of accessing personal information and even full credentials.

A solution to the problem could be the use of hardware equipments for storing private keys. However, this solution is not an optimal and "handy" solution for the majority of the participants in the environment. Even in this case there exists no guarantee that the hardware keys are not going to get lost.
Alternative solutions that can be applied are:

- **Fees for Certification.** Typically, a participant is authenticated through a certification authority (CA) which could be:
  - a trusted and dedicated participant (e.g. Grid Certificate Authorities discussed in [21]),
  - another participant in the environment,
  - the participant itself which could issue self-signed certificates.

  Obtaining a certificate, at least from the dedicated CAs, should imply also some *additional fees* for the participants who apply for it.

- **Hierarchical Certification Scheme.** Among certification authorities, some *hierarchy* should exist. Considering the certification graph in Fig. 5.1, dedicated CAs [21] should stay at top of it and have a relationship of first order (a supposed direct relationship between them exists; they have the same importance). Within such a certification scheme, all participants are connected through certification chains to each other. Participants that do not have such a connection (have self-signed certificates), have an *infinite relationship* with other participants.

- **Identity Trust.** In such an infrastructure for distributing certificates, *applying the notion of trust* among participants regarding their partners' identity, is the next step. *Trust on identity reflects the confidence of every participant on the declared identity of the others in the environment.*
  To give a participant X the possibility to determine the identity trust of another participant Y in the environment, a "certification graph" is centered at X's CA. The "oracle of certification" defines the degree of separation $D_{certification}$ of partner Y from partner X as the path length of the shortest certificate chain from X's CA to Y. Any partner who has no path to X's CA is said to have an infinite relationship with the centre of this graph and thus, ∞ is assigned.
  If partner X needs to gather information regarding the identity trust of another partner Y, after establishing the "oracle of certification" of partner Y with its CA, equation (5.1) can be used:

$$T_X^I(Y) = \frac{1}{D_{certification(X,Y)}} \tag{5.1}$$

  The oracle of certification is determined according to the algorithm in Fig. 5.5.

- **Identification Factors.** Including the identification factors (i.e. attributes of a participant) in the identification and authorization process can be considered as a further step against some of the threats such as usage of stolen or counterfeit certificates.

1. initialize $D_{certification} = 0$;
2. contact target participant and get its certification authority (CA);
3. compare received CA with own CA or any of first order CAs, $D_{certification} = D_{certification} + 1$;
4. if received CA equals personal CA, then stop searching any further and calculate $T_X^I(Y)$ according to formula 5.1, otherwise contact the CA of the target participant directly and ask for its CA;
5. repeat step 3 until received CA is the same with own CA or any of first order CAs.

Figure 5.5: Determining the "Oracle of Certification".

5.2.6 Behavior Shaping and Behavior Trust

Until now, the behavior of collaborating participants in Grid environment is an abstract notion. It is limited to a general abusive or abnormal notion of behavior of the participants during the collaboration (like by Azzedin et al. in [79], [81] and [80]) or to the estimation of only singular elements like accuracy (i.e. by Zhao et al. [270]).
In this thesis, behavior is referred to as consisting of any of the quantitative QoS requirements[10], a combination, or the entirety of them. Abstracting the common attributes from the variety of demands that the user, aiming at an optimal level of QoS, on a per case basis, places to the participants in the environment, the components of the behavior could be derived from the parameters of QoS such as: reliability (correct functioning of a service over a period of time), availability (readiness for use), accessibility (capability of responding to a request), cost (charges for services offered), security (security level offered), performance (high throughput and lower latency), etc.
Each of these parameters[11] can be directly measured or break up in measurable elements, in order to offer the possibility to create a history of data from past interactions among collaborating parties in Grid environments.

Behavior trust, the most important social element in Grid systems, deals with the trustworthiness of interaction partners, defining the confidence that a participant involved in an interaction will offer the desired QoS, and behave as expected.
Each of the participants gathers these results by continuously monitoring and controlling their interaction partners.
Trust management is considered to be the process leading to the decision which partners in a collaboration are to be trusted to complete particular actions.

5.2.7 Trust Relationships

Trust relationships are modeled as directed graphs where trust is a unidirectional directed edge from the trustor to the trustee.
A distinct feature of the trust relationships is their dynamicity. According to the observed behavior during the collaboration, in case of undesired or unexpected behavior of the other party, participants can decide on the future of the current collaboration (or future collabo-

[10] Presented at the chapter 3.
[11] Referred to from now on as behavior trust elements.

rations) with that partner.

The following trust relationships are considered:

- **consumer - provider (provider - consumer):** participants trust that their counterparts will behave properly during the collaboration. This *belief* is constructed thanks to the observations of past collaborations and/or experiences of others. At the same time it expresses the *expectations* of the parties; their partners will show at least the same behavior as in previous direct or indirect collaborations. Consumers expect providers to supply services on the desired level of quality and providers expect their consumers to behave accordingly.
  It is a *bilateral relationship*, meaning that both parties have to trust each other (not necessarily at the same level) for the interaction to take place.

- **consumer (provider) - recommenders:** this kind of trust relationship differs from the trust relationship established between consumers and providers. In this model, the experience the participants establish with their counterparts during single collaborations is considered to be "personal" to the participant itself. It can be *freely offered* as recommendation to the others in the environment that *ask* for it, without establishing any trust relationship in this direction. The trust relationship exists only from the side of the party that needs these recommendations. Every participant can ask the others if recommendations are needed, but not necessarily consider at all or at the same level every recommendation it gets. The reasons are:

  – since the experience the participants make in the environment is personal, its categorization as "good" or "bad" does not necessarily have the same meaning for the others, and

  – malicious participants could intentionally offer low trust values for well behaving participants and high trust values for others with mediocre behavior or no contribution at all.

In this model, recommendations depend:

- on the user/application specification of the trust requirements (if any recommendation is going to be considered, or what trust value to assign them) and

- on a history of the past recommendations and the resulting behavior of the participants recommended by them.

Furthermore, a separation between behavior trust and recommendation trust exists. The "ability/inability" of a participant to offer valuable recommendations should in no way interfere with decisions regarding its "capability" to properly behave during the collaboration (consumer-provider; provider-consumer) with another participant.

5.2.8 Direct Experience and Direct Trust

Every time a trustor collaborates with a trustee, a direct experience is established between them. The output of the collaboration determines the type of experience the trustor had

with the trustee and thus the *trust on the behavior* of the trustee. This is known as *direct trust*.
It is calculated based on the entirety of the behavior elements under observation. Considering the participant X as trustor and participant Y as trustee, direct trust is expressed according to the formula 5.2:

$$T_X^I(Y) = \prod_i B(Y)_i \tag{5.2}$$

where $B(Y)_i$ represents the behavior trust elements under observation. Each of the behavior trust elements is calculated as the ratio of the "positive" observed experiences with the total number of observations during the collaboration:

$$B_i(Y) = \frac{"positive\_observations"}{total\_observations} \tag{5.3}$$

The entirety of behavior trust elements is referred to as Absolute Behavior Trust (ABhvT) and any combination of them is referred as Relative Behavior Trust (RBhvT).

The accumulated experience can also be offered to the others in the environment in the form of recommendations.
Furthermore, if recommendations are accepted by third parties, the observed behavior of the trustee during the collaboration is going to influence the trust that the trustor establishes with regard to its recommenders. Thus, the trust a trustor assigns to a participant K it got recommendations from, is the average behavior trust all trustees (N_i), recommended by this participant, manifested during the collaboration (5.4).

$$T^B_{X,D_{collaboration}=0}(K) = \frac{\sum_{i \in N_i} T_X^B(Y_i)}{|N_i|} \tag{5.4}$$

5.2.9 Third Parties' Experience and Indirect Trust (Recommendations)

Recommendations could gain importance especially in those cases when the trustor never had any direct experience with the trustee(s), a certain amount of time has passed since the last collaboration, or when a more "objective" opinion on the trustee is required. However, some uncertainties exist about the "honesty" of the recommenders or even in indicated circumstances under which the experience was made.
Here, a more social-oriented approach is proposed.
Referring to Fig. 5.3, for every participant in the Grid environment, the others can be categorized as "known" and "unknown" participants based on the previous direct experiences they had. The same principle can be applied to the recommenders, a participant got recommendations from. There are recommendations from "known" sources (participants for which direct experiences exist) and recommendations from "unknown" sources, participants in the environment, with whom, considering the collaboration graph in Fig. 5.2 a path can be found using the known partners of known partners and so on. Considering the relationship between Grid participants (Fig. 5.3), the oracle of the collaboration (Fig. 5.2) expresses the relationship of the trustor to:

- *itself* - personal experience is considered as a "recommendation to itself". The participant stays at the root of the graph; the oracle of collaboration is 0 ($D_{collaboration} = 0$). Trust of a trustor X to the trustee Y is given by $T^B_{X,D_{collab.}=0}(Y)$ and expressed by equation 5.2.

- *known sources* - participants in the environment who have previously offered their experience to the trustor. The oracle of collaboration is 1 ($D_{collaboration} = 1$).
  Each partner in the network can now calculate the weighted recommendations coming from the set of known sources N_k according to equation (5.5).

$$R_{X,D_{collab.}=1}(Y) = \frac{\sum_{k \in N_k} (T^B_{X,D_{collab.}=0}(K) \cdot T^B_{k,D_{collab.}=0}(Y))}{|N_k|} \tag{5.5}$$

- *unknown sources* - participants in the environment who had never offered their recommendations to the trustor previously (the fact that they could have offered their recommendations to trustor's known partners, known partners of the known partners and so on is not excluded). The oracle of collaboration is considered to be simply greater than 1 ($D_{collab.} > 1$).
  There are two strategies for calculating the total weighted recommendations coming from the set of unknown sources, either considering the experience values of each participant p_i along the shortest path $P = \{p_1, \ldots, p_D\}$ between partner X and partner Y or taking only the experience of the participant in the path preceding partner Y into account. This value is weighted based on the oracle of collaboration of this participant from partner X (5.6):

$$R_{X,D_{collab.}>1}(Y) = \frac{\sum_{u \in N_u} \left(\prod_{i=1}^{|P_u|} T^B_{p_{u,i},D_{collab.}=0}(p_{u,i+1}) \right)}{|N_u|} \tag{5.6}$$

  where $P_u = \{p_{u,1}, \ldots, p_{u,D_u+1}\}$ is the shortest path from X to Y with $p_{u,D} = u$.
  Equation (5.6) requires several subjective decisions in the determination of every partner's trust along the path, which are based on experience and on several uncertainties regarding the evaluation of these experiences. Therefore, equation (5.7) represents a more *prejudiced* evaluation of the recommendations based on the idea that a more objective metric is needed to weight recommendations originating from unknown sources.

$$R^p_{X,D_{collab.}>1}(Y) = \frac{\sum_{u \in N_u} \left(T^I_X(u) \cdot T^B_{u,D_{collab.}=0}(Y) \right)}{|N_u|} \tag{5.7}$$

5.3 First Trust Problem

The first trust problem introduces itself in situations when Grid participants, completely "unknown" to the others, enter the environment. No personal experience exists with any of the participants, therefore all of the trust sources referenced in (5.5) are equal to 0. The usual strategies for selecting a partner do not apply in this situation.

In this work, two different basic strategies for "initializing" trust are distinguished:

- One is a rather open approach to assign an initial trust value slightly above the minimal trust threshold to every partner, effectively giving the partner a chance to prove itself trustworthy without prior verification. This method is called *"warm start phase"*.
- In contrast, there might be scenarios in which a partner is tested by performing validation checks and deriving initial behavior trust from these interactions. Obviously, this trust establishment phase through a *"cold start"* comes at a comparably high price. The initial trust value assigned to new participants during this initialization procedure equals 0.

The problem of establishing first trust may be seen both from a consumer as well as a provider point of view. For this reason, trust management for Grid environments is designed flexible enough to allow specification of the strategy to be used in either role and on an application basis.

5.4 Verification Strategies

For the participants, observing the behavior of their counterparts is fundamental for:

- determining the success of the ongoing collaboration between them, verifying that a particular partner stands up to the assumed or previously offered behavior and
- aiding the decision making process when it comes to choose partners in future collaborations.

The extent to which verification is performed may vary depending on application scenarios or various user requirements. Also, the need for verification of the partners' behavior may arise in both roles (i.e. consumer or provider) independently of the Grid scenario. Participants will continuously monitor the interaction process among each other and in case of discovered deviations in the behavior of their partners, scheduling or access policies will be reorganized accordingly.
The different behavior trust elements (e.g. availability, accessibility, accuracy, response time, etc), for which trustees were chosen, are the criteria for developing verification strategies for the trustor. However, the verification of these elements could turn up as very costly or even impossible for the trustor. It could be pretty easy to verify the availability, the accessibility, response time, etc., of the collaboration partner(s), even by the trustor itself, but from the very nature of the tasks running in Grid environments, it is very difficult for a single participant to verify elements like e.g. accuracy of the response a consumer participant receives from a provider. The strategy to use for the verification of such behavior trust elements may vary depending on certain constraints such as the additional acceptable cost for performing the verification operations.
Regarding verification of the accuracy of the responses, the following verification strategies can be applied:

- ***Challenge with known tasks*** - A service consumer may prepare a particular test challenge for which it knows the correct result. In this case, the consumer can directly verify if a service provider was able to calculate the correct response for this challenge. However, a malicious provider may be able to guess the correct answers to the set of challenges and thereby manipulate the behavior trust assigned to it, since the computational resources of the consumer for test preparation may be very limited.

- ***Best of n replies*** - A more feasible verification technique is similar to the one that is used by SETI@HOME [46]. The validity of the computed results is checked by letting different entities work on the same data unit. At the end, a majority vote establishes the correct response. This technique results in a significant increase of the total computational costs to solve a particular problem, since the overall workload is multiplied by the replication factor used to reassign subtasks. If there is a service charge collected by a service provider for every processed work unit, this also leads to a considerable increase of the overall cost.

- ***Applying the "frequency of the verification"*** - Here, a similar approach to SETI@HOME [46] is used for verifying the accuracy of the responses coming from collaboration partners. Instead of sending replicas of original tasks to many partners and applying a "best of n replies" verification strategy, only a replica of the original task is send to partners whose accuracy of the response is already proven. The coming results are compared and a decision is taken. Additionally, for the participants, the possibility to develop a personalized verification strategy regarding each of their collaboration partners is foreseen. For this purpose, the notion of *"frequency of the verification"* is introduced.
  In this approach, the verification of the accuracy of the responses coming from a certain partner is coupled with the last measured trust value of the associated behavior trust element (accuracy) of this participant. This relationship is expressed by (5.8):

$$f = -((1 - V_{min}) \cdot T^B_{last}) + 1 \tag{5.8}$$

  where V_{min}[12] is the minimal verification rate set by the consumer and T^B_{last} represents the trust value for the accuracy of a provider at a certain moment of time.
  From the consumer side this means that for a non-trusted provider every single response is verified and for a fully trusted provider only a minimum of the responses coming from that specific provider has to be verified.

Through this approach:

- trustors do not need to make replicas of every task they send to trustees, thus overhead in the system is significantly reduced and

- such a moderate number of tasks (replicas) to be verified makes it possible for the trustor to execute replicas even by itself without the need of a trusted third party.

The result of the verifications will directly be used to alter behavior trust regarding accuracy and indirectly the absolute behavior trust of the target participant.

[12]This value, as presented in next subsection, with be expressed on the user/application trust requirements, according to the needs for trust from the environments and the involved participants.

5.5 User/Application Specific Trust Requirements

A further consideration is the involvement of the human users through allowing them, according to the type of the Grid application, to express their personalized trust preferences towards the participants that are going to be considered as collaboration partners. These preferences include the initialization values that the user is willing to assign to each of the new partners, the selection of sources for getting trust information from (recommendations), verification strategies for all the trust elements, etc.

5.5.1 Application Requirements

In order to better illustrate how different application requirements may arise depending on the field of application, two scenarios, already introduced in the second chapter, will be analyzed:

- **Grid Services for Video Cut Detection.** The first application scenario identifies "cuts" in videos (Ewerth et al. [134]). To identify such cuts, the video file is split and all parts are sent to a manager, which assigns the jobs (split video files) to remote services that will take care of the analysis and the identification of the cuts. After all the jobs have been processed, the resulting cut lists are merged together, some boundary processing takes place in the manager, and the user is notified that the work has been finished.
  Video analysis is a collaborative task with moderate security requirements and even *moderate trust requirements* regarding the behavior of the involved parties. In this case, an open attitude accepting recommendations from strangers and requiring only occasional verification of the data returned from individual data processing services may be sufficient to satisfy the users of the application.

- **Grid Services for Medical Imaging.** The second example is to find similar cases (images) in a set of mammograms (Amendolia et al. [75]) for diagnosis support. Images resulting from mammography are passed to remote services representing image analysis algorithms which calculate similarities in image sets.
  These image analysis algorithms may be basically similar to those used in the identification of video cuts or in object recognition tasks within frames of video sequences. However, different standards are required in the medical scenario compared to the video application:

  - The radiologist may be interested in simply performing standard image processing techniques on generated data sets without placing any special trust requirements on the interaction partners. A subset of images stored on a remote image server is retrieved and viewed. Analysis of image data on the local workstation is performed. Only occasional verification of the data returned from individual data processing services may be required.
  - The analysis can be extended to the application of several different statistical analysis methods on data from multiple different studies to view the effects of a treatment method across a patient group. In this case, trusting the accuracy of the responses coming from the interaction partners is important. As a result,

high trust requirements are imposed regarding the behavior of the involved parties. The radiologist may consider only own experience or recommendations from highly trusted sources for the selection of partners. A high frequency of verification of the data returned from the individual data processing services is required.

While the basic application is similar in both scenarios (i.e. applying a number of image processing algorithms to a set of images and merging the individual results), the domain specific trust requirements lead to different application behavior.

5.5.2 "Human on the Trust"

Users, according to the application scenario, can express their personal requirements and configure the following elements of the trust model presented in this thesis:

Oracle of Certification. The user establishes the oracle of certification (separation distance from the CA of consumer he/she is collaborating with) of the participants that are going to be found as suitable for fulfilling his/her needs. The user is expected to either accept participants with the oracle of certification 1 (participants certified through the same CA as the consumer or through any of the first order CAs), or accept anyone, independently of the oracle of certification. The identity trust of the accepted participants is going to be calculated according to formula (5.1).

Behavior Trust Initialization. The decision whether to "warm start" (give to new participants the chance to prove that they behave accordingly to the expectations) or "cold start" (all new participants in the environment are considered as not trustworthy and thus the minimal trust threshold is assigned to them) can be taken only by the user, according to the application that is going to be executed.

Behavior Trust Elements of Interest. The behavior trust elements where the user is more interested in are considered. These elements could vary from single or combination of some of them (Relative Behavior Trust - RBhvT) to the entirety of them (Absolute Behavior Trust - ABhvT).
According to the specified elements, only the most trusted participants are going to be chosen for the upcoming collaboration (trust values regarding these elements need to be greater or equal to the value specified by the user or if not differently specified, to the maximum, thus 1). Furthermore, when monitoring the collaboration, only the specified behavior trust elements are going to be verified for discovering possible deviations during the collaboration. The rest of the behavior trust elements (those not specified) are going to be considered as "not important" for the user. Their current trust value will not have any impact on the decision whether a participant is eligible for collaborating with and they will not be monitored during the collaboration.

Trust Profile Vector. Users need the flexibility to weight the different sources for the behavior trust differently in different situations, contributing directly to calculate the resulting normalized trust to put into an interaction partner. For this purpose, a vector of all trust sources a trustor X may use for rating a trustee Y is defined (5.9):

$$\vec{T}_X^S(Y) := \left(T^B_{X,D_{collab.}=0}, R_{X,Dcollab.=1}, R_{X,Dcollab.>1}, R^p_{X,Dcollab.>1}\right) \tag{5.9}$$

Now, the trustor X can calculate the resulting normalized trust to put into trustee Y using a profile vector $\vec{P} \in [0,1]^4$:

$$\vec{P} = \left(w_{D_{collab.}=0}, w_{Dcollab.=1}, w_{Dcollab.>1}, w^p_{Dcollab.>1}\right) \tag{5.10}$$

where the relative weights express direct experience ($Dcollab. = 0$), indirect experience of "known" sources ($Dcollab. = 1$), experiences of "unknown" sources ($Dcollab. > 1$) without prejudice and the last weight for experiences of "unknown" sources making prejudices on how they have gathered their experience with interaction partner Y. The weights can have values in $[0, 1]$.
Having defined the relative weights for the different sources, trustor X can calculate the resulting normalized trust to put into a trustee Y:

$$T^B_X(Y) = \frac{\vec{T}^S_X \cdot \vec{P}_X}{\left\|\vec{P}_X\right\|} \tag{5.11}$$

The resulting trust value is only used for the decision to interact with a certain participant. It does not affect the experience value, because this value only depends on the outcome of the subsequent collaboration.

Verification Strategies and Sub-Strategies[13]**.** The user may choose to:

- *"Trust-no-One"* - In this case, the user may choose to verify the accuracy of every single response. This technique results in a significant increase of the total computational costs to solve a particular problem, since the overall workload is multiplied by the replication factor used to reassign every subtask. If there is a service charge collected by trustworthy participants for every processed work unit, this also leads to a considerable increase of the overall cost.

- *"Optimistic-Trust"* - In order to minimize added costs, the user may choose to verify the accuracy of only a part of the responses. In this case the user specifies the minimal verification frequency V_{min} with values $[0, 1]$. The idea behind is that for a non-trusted participant every single response may preferably be verified and for a fully trusted participant only a minimum of the responses have to be verified.

The verification strategies the user may choose to apply are:

- **Offline verification** - behavior trust elements like the accuracy of the responses coming from a certain participant are verified only at the end of the collaboration, before the user receives the results. The user needs to specify only the minimal verification frequency to apply to the verification process.

- **Online verification** - behavior trust elements like accuracy of the responses coming from a certain participant are continuously monitored during an ongoing collaboration.

[13]The verification strategies will be explained in detail in the next subsection.

The user needs to specify the minimal verification frequency, and also the *clearance number "i"*. It is used to establish the number of responses to be sequentially verified (100% verification), before applying the frequency of verification.

Stopping Rule "$nc_{BhvT\_element}$"**.** In a certain sense, this represents the tolerance that the user has against deviations in the behavior (behavior trust element of interest) of the interaction parties. Regarding the example behavior trust element, the accuracy of the responses, the stopping rule represents the number of $non-accurate$ responses that the user is willing to accept from a target participant. Once this number is exceeded, the interaction with the specific partner is interrupted and the scheduling policy is reorganized.

5.6 Keeping the Behavior of the Collaboration Parties "In Control".

Until now, the key concepts of how the participants establish and manage trust among them are presented. Considering different sources for gathering trust information from (self experience, indirect experience, user/application trust requirements), each participant sorts out the collaboration partners and starts interacting only with the "most trusted" of them. During the collaboration, behavior trust elements are constantly verified either 100% or at a certain verification frequency (5.8). According to the verification results, trust values are updated influencing directly the decision making process if the collaboration with a certain participant will continue or will be interrupted. Naturally, if the expected behavior was met, the collaboration will most probably continue. The problems start once any deviation from the expected behavior of a collaboration partner is verified.
The behavior deviations can be categorized under:

- deviation within the current collaboration and
- deviation in time.

The first has a more immediate effect on the current collaboration and the validity of the data being processed. If a 100% verification strategy is applied, it is easy to tell that until that specific moment, no other deviation was ascertained. On the contrary, if a verification frequency was applied, how could it be possible to tell that no more deviations exist except those verified? In the trust context: *"does any difference exist between the trust calculated for that specific participant, according to the observed behavior, with the real trust that should have been assigned to it according to its real behavior?"*
The second deviation affects more than the current collaboration. It deals with the trustworthiness of a participant in general and its intentions. As an example, let consider a provider which offers a high processing speed for the tasks assigned, applying also high charges for the offered services. At the beginning it behaves according to the expectations of its partners. As time passes, trying to have the maximum profit, it starts serving at the same time a greater number of consumers that it can afford without negatively affecting the processing speed, applying the same charges. How could it be possible to verify these "long term" deviations in the behavior of a participant? In the trust context: *How to verify the deviations on the trustworthiness of a participant?"*

Another question that comes out either in the case when deviations are observed within a single collaboration or in the case when deviations are observed in time, is: *"How long should a collaboration continue with a participant despite its anomalous/malicious behavior?"*

For answering the above posed questions, new functionalities need to be added to the trust model. For this reason, statistical methods of quality assurance are going to be considered. *Collaboration(s) among participants is seen as a "production process" where the behavior trust elements under observation establish the "quality" of the collaboration process.*

In the following, an overview of the statistical methods of quality assurance together with their application for keeping the behavior of the collaboration parties in Grid environments "in control" will be presented.

5.6.1 Statistical Methods of Quality Assurance

Statistical methods for monitoring and improving the quality of manufactured goods have been around since the early 1920s when W. A. Shewhart introduced the graphical control chart method for detecting possible problems in manufacturing processes (Mittag and Rinne [202]).
The term "quality" is broadly used by service industries and it embraces all the characteristics of an entity (goods or services) that determine the capacity to satisfy the expressed and implicit requirements of who uses it.
Current applications of statistical methods of quality assurance have widened to include many service industries as well as traditional manufacturing applications.
General aspects of the quality are:

- The quality of output process. Goods and services are produced with various degrees of quality.
- The conformity to already set process regulations and standards.

Every output possesses a number of measurable elements perceivable by the user, which contributes jointly to the formation in the quality of the product. These elements can be indicated as quality characteristics or quality parameters. Quality parameters can be of different types. Generally when the characteristics of quality are measures, expressed on a continuous scale (weight, resistance, length and duration) one speaks of variables. When instead discrete data are considered, for instance data that can be numbered (number of non-working light bulbs, etc.) one speaks of attributes. Quality parameters are evaluated in comparison to the specifications or the established values for any of the quality parameters of the product/service. The desired value of the quality parameters is defined as the nominal value or the target value.
The following statistical concepts for quality assurance are considered:

Sampling and Sampling Distribution. The primary goal of statistical quality assurance is to draw inference about the fulfillment of a quality standard in a population based on information about individual units. From the statistical point of view, the quality standard

of an individual unit is related to the specific realization of its quality parameter, and the quality standard of the population (the entire output) is related to a function parameter, or a functional parameter of the distribution of the quality parameter. The population consists of a finite or infinite collection of elements where the sample(s) to be verified are taken from. A *random sampling procedure* is the procedure of selecting a finite number of units from a population through a random mechanism.

Common Sample Statistics. The elements $x_1, \ldots, x_n$ of a random sample are the realizations of the corresponding random variables $X_1, \ldots, X_n$. The latter constitute the sample vector $X = (X_1, X_2, \ldots, x_n)$.
The vector notation $x = (x_1, x_2, \ldots, x_n)$ is the realization of the sample vector X. The following statistics is taken into consideration:

- Sample Sum:
$$X_n^T := \sum_{i=1}^{n} X_i \tag{5.12}$$
where T stands for "Total".

- Sample Mean:
$$\overline{X}_n := \frac{1}{n} \sum_{i=1}^{n} X_i \tag{5.13}$$

- Sample Variance:
$$S_n^2 := \frac{1}{n-1} \sum_{i=1}^{n} (X_i - \overline{X}_n)^2 = \frac{1}{n-1} \left(\sum_{i=1}^{n} X_i^2 - n\overline{X}_n^2 \right) \tag{5.14}$$

- Sample Standard Deviation:
$$S_n := +\sqrt{S_n^2} \tag{5.15}$$

Average Outgoing Quality (AOQ). AOQ is also an important characteristic of the sampling plan. Considering the above posed questions, it gains even a greater importance in this model, because it determines the portion of defective units through the inspection procedure. Thus, according to the verification results, determining the AOQ of the verification plan makes possible to identify the differences between the observed and the "real" behavior of a participant during the collaboration.

Average Total Inspection (ATI). ATI for a sampling plan is the average number of units inspected per lot in a series of lots. It depends upon the lot size, the sampling plan and on the incoming fraction defective.

Average Fraction Inspected (AFI). AFI tells what proportion of the total responses coming from a participant is verified on average.

Process Control. The underlying concept of statistical process control is based on the

comparison of the present process' output with the previous outputs. In this way, a snapshot of how the process performs in general, is taken. These data are used to calculate the control limits for the expected measurements of the output of the process. Data from the running process is collected and compared to the control limits. The majority of measurements are supposed to lie within the control limits. Data that fall outside the control limits are examined and perhaps, some will later be discarded. If so, the limits would be recomputed and the process is repeated.
There are many ways how to implement process control. An alternative is offered through control charts. Control charts (Fig. 5.6) are used to track process statistics such as a subgroup mean, individual observation, weighted statistic, or number of defects, versus sample number or time and to detect the presence of deviations. Control charts consist of:

- Centre line at the average of the statistic by default.
- Upper control limit, $3S_n$ above the center line by default.
- Lower control limit, $3S_n$ below the center line by default.

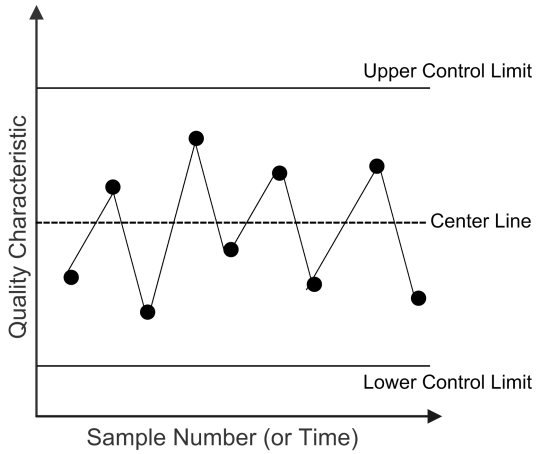


Figure 5.6: Example of a Control Chart.

A process is considered to be in control when after a verification, points fall within the bounds of the control limits, and the points do not display any nonrandom patterns. When a process is in control, it is possible to use control charts to estimate process parameters needed to determine capability.

These key concepts, together with some other additional sampling methods (presented in the following) will be considered for defining the verification strategies. The strategies for the verification of the behavior of collaboration partners are "offline verification" and "online verification". Each of them is able to make a difference, according to the user/application trust requirements, between the observed and the "real" behavior of a collaboration partner. Differences on the performance of each verification strategy will be considered in the next chapter.

5.6.2 Offline Verification

Under "offline verification", the verification of special behavior trust elements (i.e. accuracy of the results coming from a collaboration partner, etc) *at the end of the collaboration process* has to be understood.

Verification of Attributes. In the following, it will be assumed that the behavior trust element under observation can be assigned exactly one of the two possible definitions:

- "non-deviating/conforming" and
- "deviating/nonconforming".

The considered parameters are:

- N - total number of units of the behavior trust element under observation, received from a specific collaboration partner.
- n - total number of verified units of the behavior trust element under observation from the total $N(0 < n \leq N)$. It is randomly established using the formula (5.8).
- D - "real" deviations in the behavior trust element under observation. It is known only to the collaboration partner (establishes its "real" behavior during the collaboration)
- d - verified deviations in the behavior trust element under observation.
- nc - stopping rule; established by the user, determines his/her acceptance/rejection tolerance of possible deviations of the behavior trust element under observation ($0 \leq nc < n$).
- P - the "real" fraction of defective units in N considering D. It is calculated as:

$$P = \frac{D}{N} \tag{5.16}$$

Single-Sampling Plans. A single sampling plan is the one in which a random sample of units of the behavior trust element under observation is taken from the entirety resulting from a collaboration process, is inspected/verified and at the end a decision is taken on the acceptance/rejection of the entire process.

During the offline verification, the verification of the responses coming from a single provider, follows the algorithm in Fig. 5.7.

The sample n is taken out randomly from N and without replacement. The performance of the sampling plan is assed through:

- **Average Outgoing Quality.** AOQ for offline verification is calculated according to the formula (5.17).

$$AOQ(P) = P \tag{5.17}$$

  If deviations are found and they overcome the user expectations, none of the units (completed tasks) coming from that specific partner is going to be considered.

1. initialize number of defective responses found $d_f = 0$ and number of responses verified $u_f = 0$;
2. $u_f = u_f + 1$;
3. verify the u_f-th response;
4. if deviation is found, $d_f = d_f + 1$ and update respective trust value according to (5.3), otherwise go to step 5;
5. check if $d_f \leq nc$;
6. if yes, repeat from step 2 until $u_f = n$, otherwise interrupt verification and do not consider responses from that participant.

Figure 5.7: Single-Sampling Plan.

- **Average Total Inspection.** ATI for the offline verification strategy is calculated according to the formula (5.18):

$$ATI(P) = n \tag{5.18}$$

- **Average Fraction Inspected.** AFI for the offline verification strategy is simply calculated according to the formula (5.19):

$$AFI := \frac{ATI}{N} \tag{5.19}$$

5.6.3 Online Verification

Under "online verification", the verification of special behavior trust elements (i.e. accuracy of the results coming from a collaboration partner, etc) *during the collaboration process* has to be understood.

The considered parameters are:

- i - the clearance number. It establishes the number of sequential units that need to be verified (100% verification period) and is expressed by the user in his/her trust requirements.

- k - the sampling interval. It is the size of the block to pick one unit for verification and is calculated according to the formula (5.20):

$$k = \frac{1}{f} \tag{5.20}$$

  where f results from (5.8).

- P - is the fraction of defective entities discovered during the verification process with respect to the total number of entities verified.

Continuous Sampling Plans (CSP). The aim of the CSP is to control the intensity of the verification process depending on the verification results in such a way so that the maximum of the average outgoing quality does not exceed a specified limit.

1. initialize the variables for the number of entities (responses, results etc.) verified through 100% inspection $v_i = 0$, number of the entities to be verified through the frequency of verification in (5.20) $v_k = 0$ and number of defective entities found $d_f = 0$;
2. $v_i = v_i + 1$, verify the v_i-th unit;
3. if defective, then $d_f = d_f + 1$, and update the corresponding trust value according to (5.3);
4. if $d_f \leq nc$, repeat from step 2 until $v_i = i$, otherwise interrupt collaboration and do not consider what is received from that participant;
5. $v_k = v_k + k$, verify the v_k-th unit;
6. if defective, then $d_f = d_f + 1$, and update the corresponding trust value according to (5.3);
7. if $d_f \leq nc$, repeat from step 5 until no more units are left, otherwise interrupt collaboration and do not consider what is received from that participant.

Figure 5.8: Continuous Sampling Plan.

Execution of a CSP plan follows the algorithm in Fig. 5.8.

Measuring the Performance of Continuous Sampling Plans. The performance of the CSP could be assessed through the same parameters as in the measurement of performance of the single-sampling plan.

The parameters needed to quantify the effectiveness of a CSP are the average fraction inspected (AFI) and average outgoing quality (AOQ). All other relevant measures of CSPs can be computed from these two values.
In terms of the plan parameters:

- **Average Outgoing Quality.** AOQ for the online verification strategy can be measured according to the formula (5.21)

$$AOQ(P|i;k) = \frac{(k-1)P(1-P)^i}{1+(k-1)(1-P)^i} \tag{5.21}$$

  where for $P = 0$ or $P = 1$: $AOQ(0|i;k) = AOQ(1|i;k) = 0$.

- **Average Total Inspection.** ATI for the online verification strategy is calculated according to the formula (5.22):

$$ATI(P|i;k) = \frac{1}{P(1-P)^i} \tag{5.22}$$

- **Average Fraction Inspected.** AFI for the online verification strategy is calculated according to the formula (5.23):

$$AFI(P|i;k) = \frac{1}{1+(k-1)(1-P)^i} \tag{5.23}$$

  where for $P = 0$: $AFI(0|i;k) = \frac{1}{k} = f$ and $P = 1$: $AFI(1|i;k) = 1$.

1. Gather the data for the period of interest.
2. Calculate the center line.
3. Calculate the control limits (the upper and lower control limits).
4. Verify if observed data lie lie within the the control limits.
5. Classify the behavior of the participant.

Figure 5.9: Control Charts for Behavior Trust Elements.

5.6.4 Behavior Monitoring

Most of the related work considers that participants in Grid environments behave either "good" or "bad". In most of the cases "good" behavior reflects the expectations of the collaboration party to simply receive a response from its partners or sometimes considering only the accuracy of the results. If a collaboration party is going to behave other than these "normal" expectations, the behavior of the other party is going to be labeled as "bad". Participants with "good" behavior are considered as trusted ones and as a result eligible for future interactions. Others have only minor or not any possibility at all to be taken in consideration for further interactions.
To support a flexible behavior management and classification system, additional mechanisms are necessary, e.g. splitting behavior in detailed elements, making continuous observation on them and offering the possibility for the behavior classification. This will answer also the questions related to the trustworthiness of a participant in time and its consideration in case a deviating behavior is verified.

Observing the Behavior of Grid Participants. In general, a process is considered as statistically stable over time (with respect to the behavior trust element under observation) if the distribution of this element does not change over time. Stability makes the prediction of the range of variability to expect in the behavior trust element in the future possible.

Each of the Grid participants, as seen above, keeps this element, for each of their interaction partners, under continuous "observation". The primary interest relies on the discovery of the fraction of the non-conforming elements, coming from a participant, to non-standard sample size over a period of interest (e.g. as long as the collaboration takes place). Samples of measurements are periodically taken at one or more stages during or at the end of a collaboration, according to the sampling procedures of each verification strategy applied.

The steps to follow for constructing the control charts for behavior trust elements are as in Fig. 5.9.

Calculation of the Central Line and Control Limits. Proportion charts (known as p-charts) are used to calculate the central line and control limits:

- First, according to (5.13) $\overline{n}$ (mean value of all sample sizes) and $\overline{p}$ (mean value of all sample sizes proportions nonconforming) are calculated.

- Central Line:
$$CL = \overline{p} \tag{5.24}$$

- Upper Control Limit:
$$UCL = \overline{p} + 3\sqrt{\frac{\overline{p}(1-\overline{p})}{\overline{n}}} \tag{5.25}$$

- Lower Control Limit:
$$LCL = \overline{p} - 3\sqrt{\frac{\overline{p}(1-\overline{p})}{\overline{n}}} \tag{5.26}$$

"In-Control" and "Out-of-Control" Behavior. When observed behavior trust element is "in control", the fluctuations are expected to lie around the common mean (center line). *When it is "out of control", the mean usually changes and flips outside the control limits.* As "out of control", only fluctuations outside the upper control limit (UCL) are considered. Fluctuations above the lower control limit (LCL) could mean only a manifested behavior that surpasses the expectations.

Behavior Classification. In this work, it is distinguished between "normal" and "deviating" behavior. If the observed trust element lies on CL or between UCL and LCL, the behavior of the participant can be defined as normal and the participant is eligible for future interactions.
Furthermore, another fine separation regarding the types of deviations that a Grid participant manifests is considered. Despite its deviating behavior, a participant can still be considered for future interactions. In a social environment where interaction among participants is established based on interpersonal relations there do exist differences in the single expectations that each of the participants has for the behavior of its interaction partners. As a result:

- *if the observed behavior lies between UCL and CL, a participant can still be considered for future interactions but for the accomplishment of moderated expectations.*

- *if the observed behavior lies outside the UCL then the participant is banned and no more considered for future interactions.*

5.7 System Architecture

The notions discussed are summarized in the following model where the overall decision whether to trust a collaboration partner or not is under the control of the user when requesting such a decision. The underlying trust system is in charge of monitoring the collaboration process according to the strategies suggested by the user.
The presentation and the discussion will concentrate on the service consumer use of the trust management system components (every provider is also equipped with a similar trust management system bearing the same functionalities).

5.7.1 Specification of User's Trust Requirements

Every user specifies his/her trust requirements according to the schema presented in Fig. 5.10.

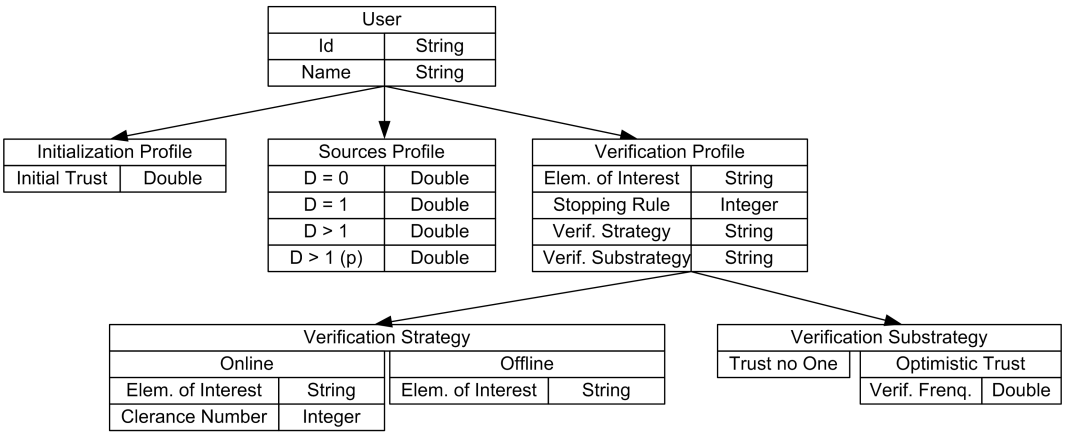


Figure 5.10: Specification of User Trust Requirements.

These trust requirements are going to be distributed on the participants' site (consumer/provider) as in Fig. 5.11.

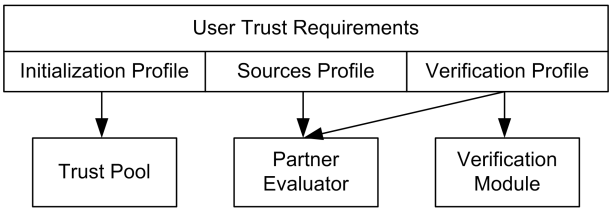


Figure 5.11: Distribution of User Trust Requirements.

5.7.2 Trust Engine

The trust engine, Fig. 5.12, is a conceptual element that manages trust values regarding the collaboration partners and offers partner discovery and rating functionality to higher level applications, such as workflow engines or job scheduling systems.

The trust engine is knowledgeable about the requirements of the user, the standards that are used and the threshold of trust that is required. It manages all trust related interactions with partners and strangers based on the information that it sources (locally stored trust values), or that is presented to it (recommendations gathered from "known" and "unknown" sources).
The trust engine assigns values to the new partners that are going to be considered or updates the trust value according to the results of the verification process. Any adverse change of the trust value leads to the re-organization of the scheduling/accessibility policies.

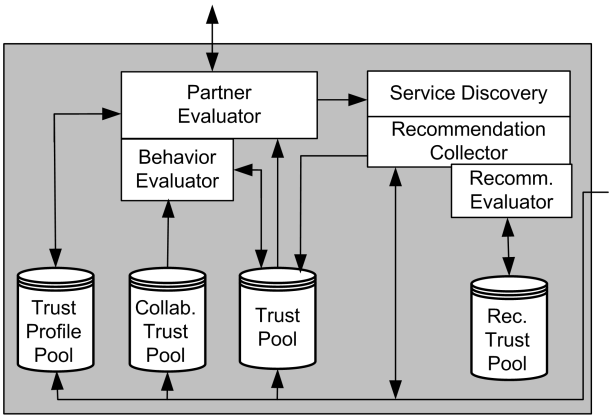


Figure 5.12: Trust Engine.

Thus, the trust engine gains a sense of the trustworthiness of a collaboration partner. Its elements are:

Service Discovery. In a Grid environment, there are several participants (several metadata services). Discovering the appropriate collaboration partner (service) is necessary. The trust engine uses its service discovery component to discover individual partner (services) (Fig. 5.13):

PartnerMap discoverParticipants (string Request)

Figure 5.13: Discovery Component.

The "request" determines the needed features (functionalities) that a participant should have.
For partner/service discovery, different solutions, already implemented can be used:

- Monitoring and discovery service (MDS) from Globus [17].
- Peer to peer node/service discovery [243].
- Brokering systems like Nimrod-G [95].

Recommendations Collection. Every participant could ask the others regarding their experience with a target participant. Recommendations are going to be requested (and weighted) according to the users' requirements (5.9, 5.10, 5.11). The recommendation scenarios are going to include either one or any combination of:

- direct experience;
- recommendations from known sources (weighted according to 5.5);

- recommendations from unknown sources and received through a "referral chain" (weighted according to 5.6) and
- recommendations from unknown sources with prejudice (weighted according to 5.7)

The recommendations collection follows the algorithm in Fig. 5.14.

1. according to the user's requirements, discover a list of possible partners to receive recommendations from (*RecommenderMap discoverRecommenders (string targetParticipant)*).
2. ask those "recommendation partners" about their trust value(s) regarding the ABhvT or RBhvT of a target participant (*requestRecommendation (string targetParticipant; string elementOfInterest)*).
3. calculate the indirect trust value for the target collaboration partner according to 5.5, 5.6, 5.7 and save to "Trust Pool" accordingly.

Figure 5.14: Recommendations Collection Algorithm..

Trust Pool. It is a storage component (Fig. 5.15) where the trust values regarding all the collaboration partners are saved for personal use during future direct collaborations or to be offered to the others in the form of recommendations.
Trust values for identity are calculated according to formula (5.1). Trust values regarding the direct experiences are updated according to 5.3 using values that come out from the verification process. Trust values regarding the indirect experiences are calculated according to 5.5, 5.6, 5.7.

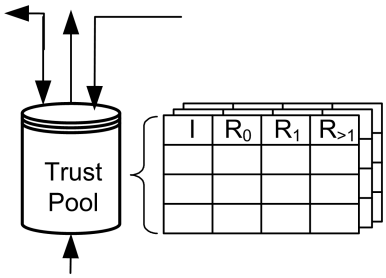


Figure 5.15: Trust Pool.

Collaboration Trust Pool. It is also a storage component where only the trust values of the partners during the current collaboration are saved. The values are updated according to 5.3 using values that come out from the verification process.

Partner Evaluator. Partner evaluator uses the information contained in the Trust Profile Pool and in the Trust Pool for determining which of the possible collaboration partners discovered suit at best to the user/application's trust requirements. The output is a list of the most suitable participants in the environment to collaborate with.

If during the collaboration the trust value(s) regarding a specific collaboration partner changes (deviation was verified) and the new trust value does not fulfill the user trust requirements, the partner evaluator changes the list of trusted collaboration partners accordingly and in this case also the scheduling policy is changed and the deviating partner is no more considered for the rest of the collaboration process.

Behavior Evaluator. The behavior evaluator module makes the classification of the behavior of a partner during the last collaboration according to the algorithm in Fig. 5.9, using the values stored in Trust Pool and Collaboration Trust Pool. If any of the observations lies outside the established limits, then the specific collaboration partner is blacklisted (added to a "black list") and no more considered for future collaborations independently of the adequacy of the trust requirements.

Recommender Evaluator. Every recommender, at the end of the collaboration with other participants, is going to be weighted according to (5.4). The recommender evaluator uses the trust values of the collaboration partners during the last collaboration (Collaboration Trust Pool) for determining the trust values that can be assigned to the recommenders, and weights the recommendations handled to the Partner Evaluator.

Recommender Trust Pool. It is also a storage component used for storing locally the trust values regarding the recommenders.

5.7.3 Verification Engine

The verification engine (Fig. 5.16) is also a conceptual element that handles the verification of behavior trust elements and generates the necessary feedback for the trust engine regarding the specific collaboration partner(s).

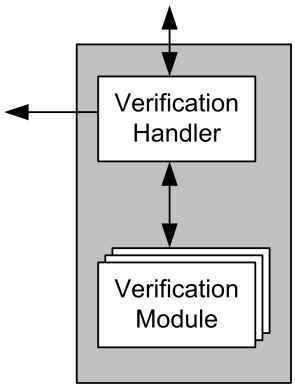


Figure 5.16: Verification Engine.

Its elements are:

Verification Module. Stores the verification strategies that the user wants to apply to the current collaboration (Fig. 5.10 and Fig. 5.11).
This information is going to be used by the verification handler.

Verification Handler. The verification handler, considering the list of selected partners for the current collaboration, together with the users' verification strategy stored at the verification module produces an XML file as in Fig. 5.17.

```
<?xml versi<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified"
attributeFormDefault="unqualified">
        <xs:element name="partnersList" type="partnersListType">
                <xs:annotation>
                        <xs:documentation>List of partners to collaborate with and the behavior trust elements of
                        interest</xs:documentation>
                </xs:annotation>
        </xs:element>
        <xs:complexType name="partnersListType">
                <xs:sequence>
                        <xs:element name="partner" type="partnerType" maxOccurs="unbounded"/>
                </xs:sequence>
        </xs:complexType>
        <xs:complexType name="partnerType">
                <xs:sequence>
                        <xs:element name="elementOfInterest" maxOccurs="unbounded">
                                <xs:complexType>
                                        <xs:sequence>
                                                <xs:element name="availability" type="xs:string"/>
                                                <xs:element name="accessibility" type="xs:string"/>
                                                <xs:any maxOccurs="unbounded"/>
                                        </xs:sequence>
                                </xs:complexType>
                        </xs:element>
                </xs:sequence>
                <xs:attribute name="partnerName" type="xs:string"/>
        </xs:complexType>
</xs:schema>
```

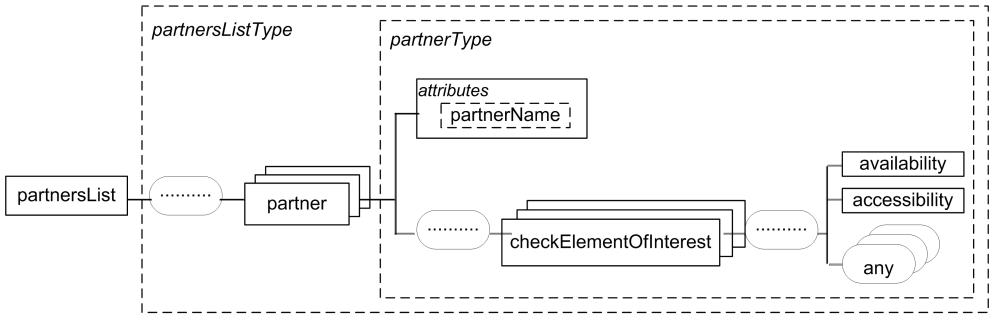


Figure 5.17: XML File Produced by the Verification Handler.

The information contained in this file is going to be used during the verification process. For every participants specified, the desired behavior trust elements will be verified.

The verification handler establishes also if a given task for a target participant needs to be replicated for verifying the accuracy of the responses of the target participant. This decision is taken according to the verification frequency applied by the user. Once the results

from the target participant and from the "trusted partner" where the replica was executed come back, a comparison is performed.
According to the verification results (example in Fig. 5.18), the respective trust values of the behavior trust elements are going to be updated at the trust pool and also stored in the collaboration trust pool.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<partnersList>
  <partner partnerName="pc12435.mathematik.uni-marburg.de">
    <checkElementOfInterest>
      <availability>The partner with name pc12435.mathematik.uni-marburg.de is Available. Partner is 100%
        available</availability>
    </checkElementOfInterest>
  </partner>
  <partner partnerName="pc12664.mathematik.uni-marburg.de">
    <checkElementOfInterest>
      <availability>The partner with name pc12664.mathematik.uni-marburg.de is Available. Partner is 100%
        available</availability>
    </checkElementOfInterest>
  </partner>
  <partner partnerName="pc12710.mathematik.uni-marburg.de">
    <checkElementOfInterest>
      <availability>The partner with name pc12710.mathematik.uni-marburg.de is Not-Available. Partner is 98%
        available</availability>
    </checkElementOfInterest>
  </partner>
  <partner partnerName="pc12545.mathematik.uni-marburg.de">
    <checkElementOfInterest>
      <availability>The partner with name pc12545.mathematik.uni-marburg.de is Available. Partner is 100%
        available.</availability>
    </checkElementOfInterest>
  </partner>
</partnersList>
```

Figure 5.18: Example Results of the Verification Process.

5.7.4 Putting it all together

The above presented components collaborate with each other as shown in the systems architecture in Fig. 5.19.

The user starts with specifying his or her trust requirements along with the input data to a trust enabled Grid application (step 1), which in turn uses the workflow engine of the local service-oriented Grid platform (step 2). To enable the selection of trusted services, the decision is made based on a rated list of potential partner services that is obtained from the trust engine (step 3). The trust engine uses its service discovery component to discover individual services (step 4) and to collect recommendations from other trust engines (step 5). These values are stored in the local trust pool to be used in subsequent interactions. The user specified trust profile is also stored in a trust pool for later reference and used by other components in the trust engine. The information gathered by the trust engine is now processed according to the user's trust profile specification and passed on to the workflow engine which then can use the partner services according to the rating generated by the trust engine.
Invocation of external services is then delegated to an invocation handler (step 6). The invocation handler consults the verification engine (step 7) to determine whether a call has to be replicated or redirected. The verification engine considers the trust profile managed by the trust engine (step 7), allowing, for example, cost-trust-ratio relations to be taken into

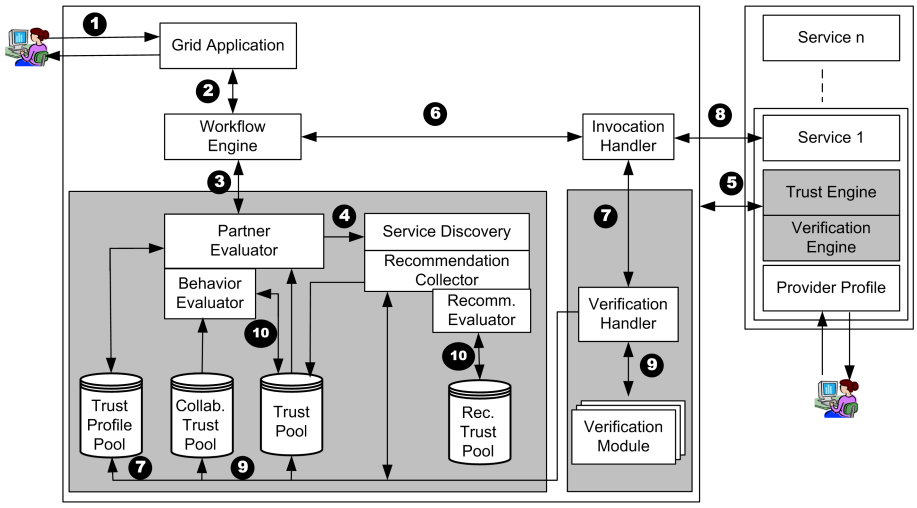


Figure 5.19: Architecture of a Grid System Supporting the Trust Model Presented in this Thesis.

account. The resulting invocation is carried out at the selected partner services and results - both synchronous and asynchronous (notification) results - are then collected by the invocation handler (step 8) and verified through the verification engine, using a strategy and verification module consistent with the user supplied trust profile (step 9). Verification values are stored in the trust pool (TP) and the collaboration trust pool (CTP).
The verification values are passed to the behavior evaluator module and to the recommenders evaluator module which make the classification of the behavior according to the algorithm in Fig. 5.9, using the values stored in TP and CTP and the weighting of the trust for the recommenders respectively (step 10).
The overall result of this process is then passed to the workflow engine that collects results for the application to present them to the end user.

The configuration of the trust engine by use of trust requirement profiles influences three phases during execution of an application workflow. These main phases are addressed by the three arrows in Fig. 5.20.
The initialization profile determines the influence and scope of factors for initializing the trust values to be used in an interaction. It allows to manually assigning trust values to certain partners, as well as specifying how trust recommendations of partners are handled and weighted. This profile specifies the behavior of the local platform in a situation that requires the establishment of first trust.

The source selection profile determines the selection of behavior trust dimensions (e.g. availability, accuracy) as well as trust sources (e.g. personal experience, recommendations from directly known partners) to determine a partner ranking according to the application

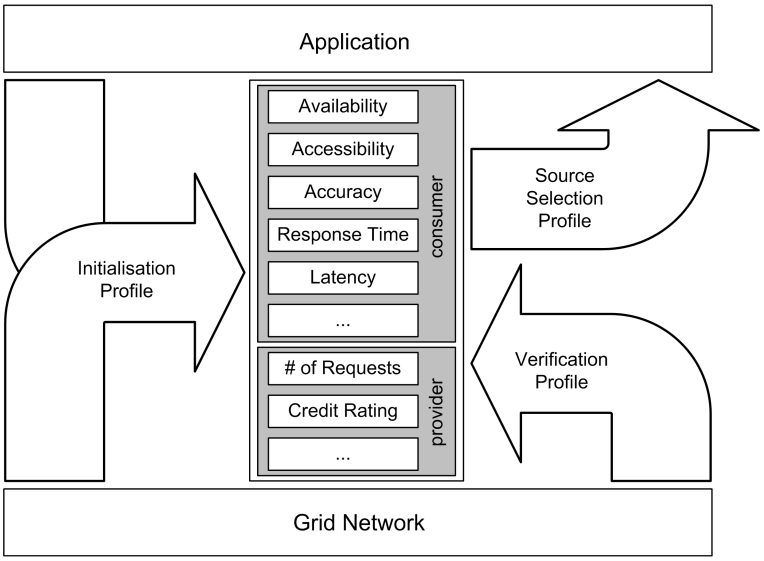


Figure 5.20: Trust profile elements influencing the stored trust values and application decisions.

needs. This allows a user to take accuracy trust recommendations from known partners into account with a higher weight than, for example, availability values (which might be caused by the different network locations) coming from the same partner.
The verification profile specifies which verification strategies are to be applied to the results of partner service invocations and the feedback parameters into the trust engine. In this profile, the user specifies how breaches of assumed service level agreements should influence the future interactions with a partner since they are fed back into the trust store for this application and partner service. This profile also dynamically determines the frequency of verification to allow a fine grained control over costs incurred by result verification.

5.8 Further Security Considerations

Until now, the problem of trust establishment and management among collaborating participants is discussed:

- the most trusted participants are going to be considered thanks to user/application trust requirements, direct and indirect experience and
- the collaboration among parties takes place as long as parties prove themselves as trustworthy.

The trust model presented in this chapter aims to gain a better view on the current behavior of the participants in the environment, helping also in the construction of beliefs and predictions regarding the future behavior of the collaboration partners. Some of the threats and risks discussed in the third chapter were considered.
Grids are designed to provide access and control over enormous remote computational resources, storage devices and scientific instruments. The information exchanged, saved or processed can be quite valuable and thus, a Grid is an attractive target for attacks to extract this information. Each Grid site is independently administered and has its own local security solutions, which are mainly based on the application of X.509 certificates for distributing digital identities to human Grid participants and a Public Key Infrastructure (PKI) for securing the communication between them. The primarily used techniques for assuring message level security are:

- public/private key cryptography - participants use the public keys of their counterparts (as defined in their certificates) for encrypting messages. In general, only the participant in possession of the corresponding private key is able to decrypt the received messages.
- shared key cryptography - participants agree on a common key for encrypting the communication between them. The key agreement protocol is based on using the target partner's certified public key.

These solutions are built on top of different operating systems. When all participants are brought together to collaborate in this heterogeneous environment, many security problems arise.
In general, Grid systems are vulnerable to all typical network and computer security threats and attacks (Negm [208] and [207], Lindstrom [185] and Bloomberg et al. [90]). Furthermore, the use of web service technology in the Grid (Foster et al. [143]) will bring a new wave of threats, in particular those inherited from XML Web Services. Thus, the application of the security solutions mentioned above offers no guarantees that the information exchanged between Grid participants is not going to be compromised or abused by a malicious third party that listens to the communication.
Furthermore, they all escape the idea *why a participant in the Grid environment was chosen among the others for completing a specified task and for how long a collaboration partner is going to be considered*. Thus, the behavior of the participants also needs to be considered in order to limit the possibility of malicious participants to actively take part in a collaboration.

An alternative solution to the problem is the establishment of a secured communication channel between collaborating participants (using a virtual private network - VPN). Thus, the transport mechanism itself is secured. Although in this case an inherently secure communication channel is opened between parties, the method itself is impractical to be used in Grid environments (Tsugawa et al. [253]) due to:

- administration overhead - new tunnels need to be configured each time a new virtual organization joins or leaves the environment.
- incompatibility between different formats used for private IP spaces in small and large networks - 16-bit private IP space is preferred for small networks, while in large networks the 24-bit private IP space is preferred. There exists the possibility that (multiple) private networks use the same private IP subnet.

There are several approaches for establishing secure communication between Grid participants. For example, the Globus Toolkit [19] uses the Grid Security Infrastructure (GSI) for enabling secure communication (and authentication) over an open network. GSI is based on public key encryption, X.509 certificates and the Secure Sockets Layer (SSL) communication protocol. Some extensions have been added for single sign-on and delegation. A Grid participant is identified by a certificate, which contains information for authenticating the participant. A third party, the Certificate Authority (CA), is used to certify the connection between the public key and the person in the certificate. To trust the certificate and its contents, the CA itself has to be trusted. Furthermore, the participants themselves can generate certificates for temporary sessions (proxy certificates). By default, GSI does not establish confidential (encrypted) communication between parties. It is up to the GSI administrator to ensure that the access control entries do not violate any site security policies.
Other approaches try to improve the security of the communication between Grid participants by making use of different encryption methods. Lim and Robshaw [182] propose an approach where Grid participants use identity-based cryptography (as proposed by Shamir in [236]) for encrypting the information they exchange. However, in traditional identity-based encryption systems, the party in charge of the private keys (private key generator - PKG) knows all the private keys of its participants, which principally is a single point of attack for malicious participants. Furthermore, the approach requires that a secure channel exists between a participant and its PGK, which in turn is not very practical in Grid environments. In a later publication Lim et al. [183], try to solve these problems by getting rid of a separate PKG and by enabling the participants to play the role of the PKG for themselves. Additionally, a third party is introduced with the purpose of giving assurances on the authenticity of the collaborating parties. Collaborating participants, based on publicly available information and using their PKG capabilities, generate session keys "on the fly", which are used between collaborating participants to exchange the initial information (job request, credentials from the third trusted party, etc.). During a collaboration, a symmetric key, on which parties have previously agreed, is used for encrypting/decrypting the information flow. This could also be a single point of attack (the attack is directed only towards a single participant) for a malicious participant willing to obtain it.
Saxena and Soh [230] propose some applications of pairing-based cryptography, using methods for trust delegation and key agreement in large distributed groups. All Grid participants that collaborate at a certain moment form a group. A subset of group members generates the public key, and the rest of the group generates the private key. A distributed trusted third party with a universal key escrow capability must always be present for the computation of the keys. These keys (public/private) are going to be used within the group for encrypting/decrypting the communication between group members.
A similar approach is followed by Shen et al. [237] where some strategies for implementing group key management in Grid environments are proposed. The main difference to the work by Saxena and Soh [230] is the re-calculation of the group key every time a participant rejoins the group.
The vulnerability of both approaches lies in the fact that all group members are aware of the public/private key. A malicious participant, already part of the group, could decrypt all messages that group members exchange between them. Even if a malicious participant is not part of the group, a single point of attack (gaining access or stealing key information from only a single group participant) could be sufficient to decrypt all the information the group

participants exchange between them.
Crampton et al. [116] present a password-enabled and certificate-free Grid security infrastructure. Initially, a user authenticates itself to an authentication server through a username and password. After a successful verification, the user obtains through a secure channel the (proxy) credentials (public and private keys) that will be used during the next collaboration with a resource. The resource in turn verifies if the user is authorized to take advantage of its services and creates its proxy credentials and a job service in order to complete the tasks assigned by the user. A single trusted authority accredits the authentication parameters for the users, resources and authentication servers.
There are several problems with this approach. First, the complexity of the environment is artificially increased. While the authentication of the resources is done directly by the trusted authority, the authentication of the users is done by a third party, the authentication server. Adding more components to the authentication chain increases the points of attack. Second, the resource has to believe that the user is authenticated through a "trusted" authentication server and not by a malicious one. Third, the resource has to believe that the user is not impersonating someone else in the environment. Finally, a single participant (the trusted authority) is in charge of the authentication parameters of all other participants in the environment. It must be trusted by the participants, and at the same time it has access to private information of the participants. Thus, the participants' private information is not protected either in the scenario where this "trusted" third party turns out to be malicious or in the scenario where another malicious participant gains access to the private information of different participants through attacking this "trusted" third party (as a single point of attack).
Additionally, some web services security standards (applied also to Grid services) are also emerging. XML Signature [60] signs messages with X509 certificates. This standard assures the integrity of messages, but it does not offer any support for threat prevention. WS-SecureConversation [63] is a relatively new protocol for establishing and using secure contexts with SOAP messages. Partners establish at the beginning a secure context between them, but all the following messages are signed using the XML-Signature standard. XML Encryption [59] is also a standard for keeping all or part of a SOAP message secret. A participant in the communication is able to encrypt different sections of an XML document with different keys making possible for its collaboration partners to access only certain parts of the document, according to the assigned keys. However, in the case when many partners want access to the same part of the document or to the entire document at the same time, they come in the possession of the same key.

In the following such a hybrid approach, belonging to the message level security solutions, is presented. It is based on combination of two asymmetric cryptographic techniques, a variant of Public Key Infrastructure (PKI) with Certificateless Public Key Cryptography (CL-PKC). The aim of the proposal is to make the malicious efforts to compromise the communication and the information exchanged between parties as difficult and as expensive as possible. It mainly concentrates on the confidentiality of the communication between Grid participants, but issues related to authorization, integrity, management and nonrepudiation will also be treated.

5.8.1 Communication Threats

In general, there exists a flow of information from a participant to another target participant as in Fig. 5.21.a.

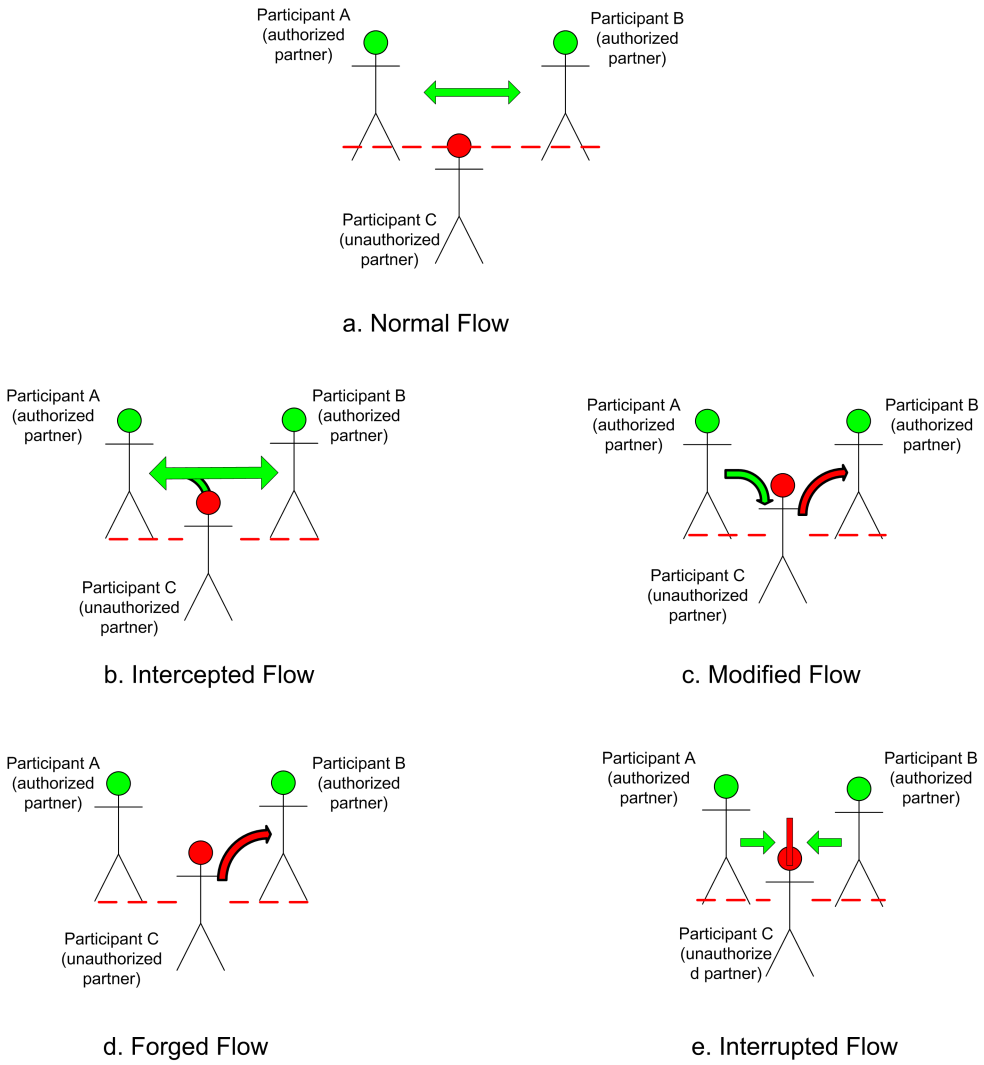


Figure 5.21: Attack scenarios to the communication between Grid participants.

This information flow can be the target of different threats. The same threats, as depicted by Stallings in [247], can also be encountered in Grid environments: passive threats and active threats.

The aim of passive threats is to simply intercept the communication and obtain the information being transmitted, as shown in Fig 5.21.b. They affect the *confidentiality* of the exchanged information, and are difficult to detect due to the lack of direct intervention possibilities on the information the parties are exchanging.

The situation changes completely when active threats are considered. Here, intervention on the information flow is always possible. The information flow can be:

- modified: the *integrity* of the exchanged information is placed at risk as a result of the modification of the data being exchanged, through the intervention of an unauthorized third party (Fig 5.21.c).
- forged: the *authenticity* of the exchanged information is placed at risk as a result of the forged stream an unauthorized participant tries to exchange with the target participant, impersonating another authorized participant in the environment (Fig 5.21.d). This is also a *non-repudiation* problem.
- interrupted: the normal communication between partners is interrupted as a result of any intervention from an unauthorized participant in the environment (Fig 5.21.e). This is a threat to *availability*.

Prevention is the key to fighting passive threats. For active threats, fast detection and recovery are crucial.

5.8.2 Basic Key Management Model

Grid systems typically make use of public key cryptography for securing a communication session between collaborating participants (Sommerville et al. [187]). Two parties use a randomly generated shared key for encrypting/decrypting the communication between them. To ensure that the data is read only by the two parties (sender and receiver), the key has to be distributed securely between them. Throughout each session, the key is transmitted along with each message and is encrypted with the recipient's public key.
A second possibility is to use asymmetric session keys. Each of the parties randomly generates a pair of session keys (a public and a private one). Their application is similar to symmetric session keys with the difference that in this case different keys are used for encrypting and decrypting messages.
Here, each Grid participant is allowed to generate its own keys such that each participant simultaneously possesses multiple public keys while all these keys correspond to a single private key. This method was first proposed by Waters et al. [258] and was later further developed by Zeng and Fujita [269].
According to their scheme, each time two participants A and B communicate with each other, the sender (participant A) decides to use either a public key from its pool of existing public keys or to generate a new one. This key is going to be sent to the receiver (participant B). Whenever B sends a message to A, the message is encrypted using A's previously sent public key. Upon receipt, A decrypts it using its private key. The entire process is described in 5.22.

The generation of the public keys is done according to the algorithm in Fig. 5.23.

The terms group and subgroup were originally defined by Menezes et al. [198]:

- A group $(G, *)$ consists of a set G with binary operation $*$ on G satisfying the following:
  - the group operation is *associative*. Thus, $a * (b * c) = (a * b) * c$ for all $a, b, c \in G$,
  - there is an element $1 \in G$, called *identity* element 0, such that $a * 1 = 1 * a = 1$ for all $a \in G$,
  - for each $a \in G$ there exist an element $a^{-1} \in G$, called the *inverse* of a, such that $a * a^{-1} = a^{-1} * a = 1$.

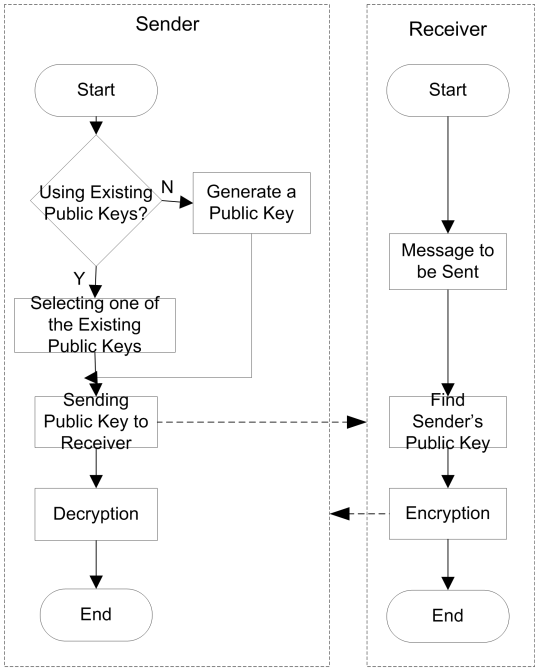


Figure 5.22: Encrypting/decrypting scheme (as used by Zeng and Fujita [269].

- A group G is *finite* if $|G|$ is finite. The number of elements in a finite group is called its *order*;
- A non-empty subset H of a group G is a subgroup of G if H is itself a group with respect to operation of G;
- A group G is *cyclic* if there is an element $a \in G$ such that for each $b \in G$ there is an integer i with $b = a^i$. Such an element a is the *generator* of G;
- If G is the group and $a \in G$ then the *order* of a is defined to be the last positive integer t such that $a^t = 1$, provided that such an integer exists. Otherwise the order of a is ∞.

To apply the above key management model to Grid environments, the following is proposed:

- First, a collaboration in Grid environments *has to take place between trusted participants*. In terms of the trust-based communication model, the collaboration takes place between the *trustors* (subjects that trust a target participant) and *trustees* (participants that are trusted). Two Grid participants involved in an interaction play both the role of a trustor and a trustee to each other.
  According to the model:

  - a participant interacts with the target participant(s) and learns their behavior over a number of interactions. In this case, the participant reasons about the outcome of the direct interactions with others. When starting an interaction with a new participant, i.e. no information about previous behavior exists, it can use its beliefs about different characteristics of these interaction partners and reason about these beliefs in order to decide how much trust should be put in each of them.
  - the participant could ask others in the environment about their experiences with the target participant(s). If sufficient information is obtained and if this information can be trusted, the participant can reliably choose its interaction partners.

- Second, *the number of public keys has to equal the number of the trusted partners (trustees) each Grid participant (trustor) selects.* In general, a normal collaboration between a trustor and its trustees, takes place as described in the following scenario. The trustor specifies the trust requirements regarding its future partners. Then, the participants which comply with the current application requirements (Grid-enabled application) are selected. The decision which one of the chosen participants should be considered further as a collaboration partner is made by comparing the trustor's trust requirements with the obtained trust information about these specific participants (personal experience, third parties' experience). Once the trustor has taken a decision regarding the "trustworthiness" of its counterparts, it generates a single private key and exactly as many public keys corresponding to this single private key as the number of its trusted partners. These keys will be used for securing the communication between the trustor and its trustees during the collaboration that is going to take place.

- Third, *directly after the generation of these public keys, the trustor has to assign a key to each of its trustees.* Thus, every trustee uses a separate public key for encrypting the messages (information) it exchanges with the trustor. The trustor itself uses a single private key for decrypting the communication flow.

- Fourth, *the generated keys should be valid only during the lifetime of the upcoming collaboration.* Since the trust values that participants establish to each other change according to the personal performance (and intentions), a trusted participant in the current collaboration is not necessarily a trusted one in future collaborations.

1. Select a cyclic group G of order n;
2. Select a subgroup of G of order m, where $m \leq n$;
3. Select and fix the private key x, where $1 < |x| < m$;
4. Select a generator g of G;
5. Select indicator r, where $0 < |r| < m$;
6. Compute $y_1 = g^r$ and $y_2 = {y_1}^x$;
7. Release public key (y_1, y_2).

Figure 5.23: Generating multiple public keys.

The entire approach is summarized in the algorithm shown in 5.24.

1. According to its needs and to the trust information gathered from different sources, the trustor establishes all the target participants (trustees) that are going to be considered in the very next collaboration (the number of trusted partners is referred with n).
2. A private key (P_B) is determined and the algorithm presented in 5.23 is repeated n times ($K_B(n)$; n public keys are generated).
3. The generated public keys are sent to the trustees; every trustee receives only one key ($K_B(i)$).
4. Each trustee, once it wants to send a message (information) to the trustor, encrypts the information flow using the respective $K_B(i)$.
5. As soon as the trustor receives the encrypted message (information), it uses P_B to decrypt it.

Figure 5.24: Multiple public keys management scheme.

The advantages of the proposed approach are:

- public keys are created by the trustor itself and are distributed directly and only to trusted participants. Not every participant in the environment is aware of them. Thus, the proposed approach mitigates also the non-repudiation problem,
- the lifetime of the private key (P_B) and the incomparable public keys ($K_B(i)$) does not span over the lifetime of the collaboration itself.

However, since the public keys are going to be distributed through a *public* and *non-secure* communication channel, the key distribution scheme is vulnerable to a *man-in-the-middle* attack. Thus, a third *unauthorized* participant could either obtain the key(s) by intercepting the information flow as shown in Fig 5.21.b or by impersonating some other trusted participant in the environment (Lenstra et al. [181]).
For this reason, the presented approach is extended by applying a double encryption scheme. A second pair of keys, generated via a certificateless key generation scheme, and information tightly related to the participant itself, is used, as described in the following.

5.8.3 A Double Encryption Scheme

Certificateless Public Key Cryptography in Grid Computing.

Certificateless public key cryptography (CL-PKC) was first proposed by Al-Riyami and Paterson in [72]. It combines elements of identity-based public key cryptography and traditional public key cryptography.
The generation of the keys is done in two stages. In the first stage, a participant in the environment receives from a key generation center (KGC), over a confidential and authentic channel, a partial private key. This partial key is computed using an identifier of the participant.
In the second stage, the participant produces its private key by combining the partial private key with some secret known only to the participant. Thus, no one else, other than the

participant itself, knows the generated private key. A public key, which matches the private key, is then published.
A distinct feature of the model is that it completely eliminates the need to obtain a certificate from the trusted authority in order to establish the authenticity of a public key.
According to Foster et al. [141], the Grid is aimed at enabling virtual communities to share geographically distributed resources as they pursue common goals, assuming the absence of central location, central control, omniscience, and existing trust relationships. Thus, having a central KGC is quite impossible. In order to overcome this problem, a hierarchical model for KGCs can be used. The idea is presented in 5.25:

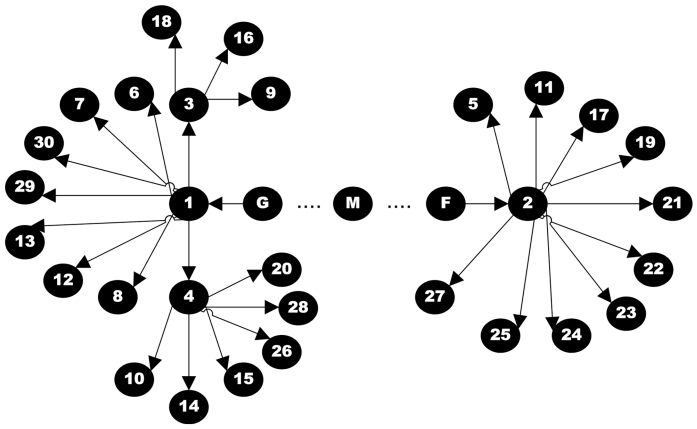


Figure 5.25: Establishing a hierarchical model for KGCs.

Every Grid participant, other than being a possible partner for the other Grid participants in the environment, could also be a KGC for another participant or even for more than one of them (i.e. in 5.25, participants G and F are KGC for participants 1 and 2 respectively, participant 1 is KGC for participants $3, 4, 6, 7, 8, 12, 13, 29$ and 30; participant 3 is KGC for participants $9, 16$ and 18; participant 4 is KGC for participants $10, 14, 15, 20, 26$ and 28; participant 2 is KGC for participants $5, 11, 17, 19, 21, 22, 23, 24, 25$, and 27). Considering the graph in 5.25, dedicated $KGCs$, like G, M and F (in charge only of partial key distribution; e.g. international or national centers, universities, etc.), have a relationship of first order, i.e. a supposed direct relationship between them exists and they have the same importance. Within such a scheme, all participants are connected through chains to each other. Participants that do not have such a connection (i.e. do not possess a KGC or serve as a KGC for themselves), have an *infinite relationship* with other participants (not present in 5.25).
The following information, already introduced in subsection 5.2.5, could be used from $KGCs$ for computing the partial private keys:

- ***What a participant is*** - refers to *personal attributes* of every single participant. Examples of these traits include hardware and software peculiarities of the participant (i.e. operating system, hardware in use, network physical address, IP address, etc). Part of these attributes or a combination of them is difficult to duplicate and very specific to a single participant.

- ***What a participant does*** - refers to *unique patterns of behavior* that this participant

manifests during the collaboration with others in the environment. Trust values can be gathered from different (ex) partners of the target participant, whose partial private key a KGC is currently computing, and be used during the computation process.

Having received this partial private key, the Grid participant could generate the full private key.

A Protocol for Encrypting/Decrypting the Information Flow between Grid Participants. The proposed protocol is a combination of the approaches described above and works in the following manner (assuming that each KGC has a master key M_{KGC} and a public key K_{KGC}):

- Every participant i contacts its KGC ($KGC(i)$) for receiving the partial private key;
- The $KGC(i)$ computes the partial private key $P_{PK}(i)$ using its master key $M_{KGC}(i)$, its public key $K_{KGC}(i)$ and an identifier $ID(i)$ (personal attributes or specific patterns of behavior) of the participant;
- The participant, in an intermediary step, computes a secret value $S(i)$ making use of $K_{KGC}(i)$ and $ID(i)$. This secret value $S(i)$ is then combined with the partial private key obtained $P_{PK}(i)$ and the $KGC's$ public key $K_{KGC}(i)$ for generating the actual private key $P_{CL}(i)$; Similarly, the public key $K_{CL}(i)$ is generated from the combination of the user's secret value $S(i)$ with the public key $K_{KGC}(i)$ of its KGC. This public key ($K_{CL}(i)$) is made available to the others through placing it in a public directory;
- The participant, according to the application requirements and to the trust information gathered from considered trust sources, establishes all the partners (trustees) that are going to be considered during the very next collaboration (the number of trusted partners is referred to with n); two partners that decide to collaborate with each other are both trustor and trustee to each other; a participant in the environment with an *infinite relationship* to the trustor is not considered at all as a trustee;
- The participant i (in this case the trustor), determines a private session key $P_B(i)$ and n different public session keys $K_B(n)$ (a different public key for each of the n established trustees);
- Before sending each public session key $K_B(j)$ to the target trustee j, the trustor encrypts it with the corresponding $K_{CL}(j)$;
- The trustee j, once receiving the encrypted message, decrypts it using its $P_{CL}(j)$, obtaining the $K_B(j)$ that is going to be used to encrypt the information flow with its partner.
- Once a collaboration has to take place, the trustor first encrypts the information using the public session key $K_B(i)$ assigned by its partner and then re-encrypts the already encrypted information using the $K_{CL}(j)$ key made public by its partner;
- The double encrypted information is initially decrypted by the trustee using its $P_{CL}(j)$ key. The obtained information is further decrypted using the personal private session key $P_B(j)$.

5.8.4 Discussion

In this subsection, a discussion on how the approach deals with the different threat scenarios presented in subsection 5.8.1 of the thesis is done:

Intercepted Flow. Here, the following scenarios can be distinguished:

- An unauthorized participant does not have any clue about the existence of the encryption of the information flow or does not possess any of the decryption keys. This is an ideal scenario, because the encryption itself brings the advantage that the unauthorized participant cannot gather any information. A brute force attack will result in significant costs and time to break the encryption.

- An unauthorized participant is aware of the encrypted flow and is able to forge or obtain the P_B and P_{CL} keys. Forging both keys of a Grid participant is an extremely difficult task, because P_B is valid only during the ongoing session, and P_{CL} is generated using specific information of this participant and is in possession of only the participant itself. Even the KGC has no complete knowledge of P_{CL}. The only possibility for an unauthorized participant is to take control of the authorized participant for obtaining the original keys. However, in order to have a fully decrypted information flow, the unauthorized participant needs to obtain all the private keys of all the authorized participants involved in the current collaboration.

Modified Flow. Following the same reasoning as above, modifying multiple encrypted information flows is a very difficult task for an unauthorized participant. Enormous efforts, monetary means and time are needed in order to succeed.

Forged Flow. In this approach, two participants establish a collaboration between them only if they are considered as trusted partners for each other. The only possibility for an unauthorized participant to forge an information flow is to impersonate another participant in the environment. However,

- impersonating a participant C in the environment does not mean that it is a trusted partner for participant A, although C might have been considered as trusted for participant B (non-transitivity of trust). An additional attack to the trust information of participant A is needed. Even though, since trust changes with time (increases, decreases), a trusted partner for participant A during a current collaboration is not necessarily a trusted one in future collaborations.

- impersonation is not enough. An unauthorized participant also needs the information owned by the authorized participant it is impersonating (i.e. the public key(s) delivered from its trusted partners).

Interrupted Flow. This attack prevents or inhibits the normal collaboration between trusted participants; the approach presented here does not offer any direct possibility to prevent such attacks. However, let us consider the scenario presented in 5.26.

In Grid environments, the trustor generally collaborates with more than one trustee. The entire process is monitored and trust information is collected with respect to every single trustee the trustor collaborated with. The components to be monitored could be derived

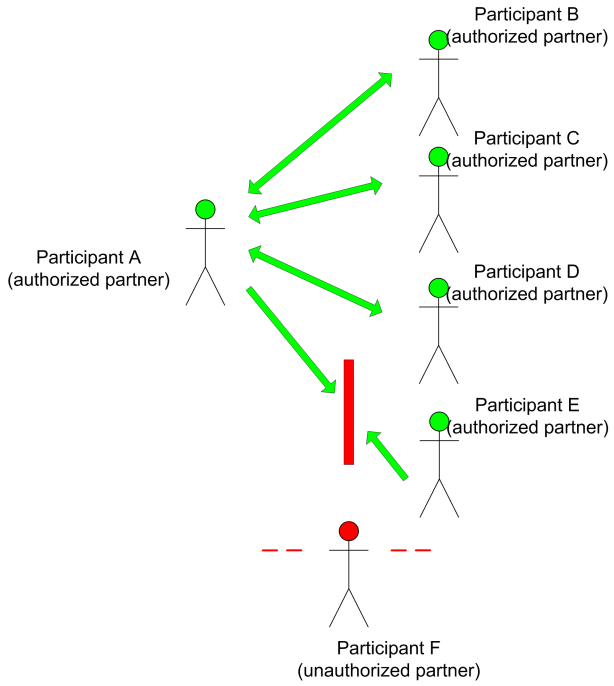


Figure 5.26: Interrupted flow between trusted Grid partners (scenario no. 1).

from the parameters of QoS like: reliability (correct functioning of a service over a period of time), availability (readiness for use), accessibility (capability of responding to a request), cost (charges for services offered), security (security level offered), performance (high throughput and lower latency), etc. In terms of 5.26, the attacked QoS element is the availability of one of the trustees. The indirect solution offered by the approach presented here is that after some unsuccessful efforts to contact the attacked partner, the flow is directed towards the other available and trusted partners and the rest of the collaboration is going to take place only with them.
However, for the attack scenario presented in 5.27, the considered approach does not offer any prevention possibilities:
In this case, (distributed) denial-of-service prevention mechanisms need to be considered.

5.9 Summary

In this chapter, a flexible trust model for collecting and managing multidimensional trust values in Grid environments has been presented. The trust model can be classified at the individual level as learning based, reputation based, and till a certain point socio-cognitive based. Both identity and behavior trust of the interaction partners were considered and different sources were used to determine the overall trust value of a collaboration partner.

The trust to the identity of the participants is established through an approach which considers the distance (oracle of certification) between collaboration partners in a "certification graph". To gain confidentiality on the identification of the collaboration partners, different

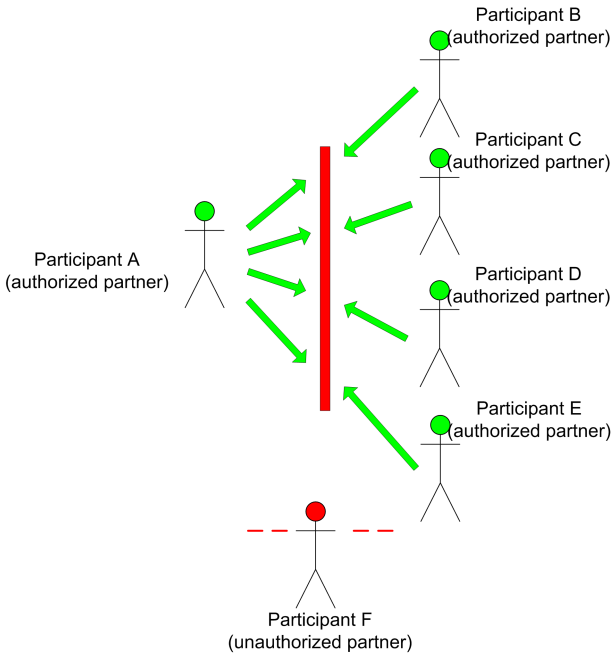


Figure 5.27: Interrupted flow between trusted Grid partners (scenario no. 2).

"identification factors" for each of the partners, difficult to counterfeit, were considered.

To behavior is referred to as consisting of any of the quantitative QoS requirements, a combination, or the entirety of them. Behavior trust deals with the trustworthiness of interaction partners, thus defines the confidence that a participant involved in an interaction will offer the desired QoS, behaving as expected. This trust relationship is bidirectional, meaning that both parties have to trust each other (not necessarily at the same level) in order for the collaboration to take place.
The beliefs and expectations are built based on past direct collaborations with that partner or on the experience of other participants in the environment. The relationship of a Grid participant to its "recommenders" is unidirectional, meaning that every participant is free to "share" its "personal experience" with everyone else in the environment. It remains to the participant who receives the recommendation(s) to decide whether to consider it at all and at what degree. A recommender is weighted according to the resulting behavior of all participants it recommended.

The trust system can be configured to the domain specific trust requirements by the use of several separate trust profiles covering the entire lifecycle of trust establishment and management. An important role on the definition of these profiles is played also by the human users. They choose between different first trust scenarios, experiences to be considered when sorting out collaboration partners and monitoring strategies. For this purpose, in the system architecture flexible and easy to use components that can be configured to the specific needs of the users on per case basis are offered.

Through the use of statistical methods for quality assurance, an alternative to monitor the behavior (or only a part of the behavior trust elements) of the participants involved in an interaction and differ among types of anomalies observed in their behavior, is offered.
This process is oriented on the discovery of differences between the "real" behavior of a participant and its "observed" behavior either during the current collaboration or in time.

The chapter was closed with an approach which tends to increase the confidence on the collaboration process between trusted partners with offering an extensible encryption scheme.
The participants first generate a single private key and as many different public keys for this private key as the number of its trusted partners. Each of the partners receives a public key which is going to be used for encrypting the information they will exchange with the participant in question. This first encryption is followed by a second one, using keys generated through a technique based on certificateless public key cryptography.
The approach tends to prevent many of the threats and attacks to the communication between Grid participants through making it even more difficult and expensive to malicious participants to trace and decrypt the information flow they exchange between them.

The performance of the trust model, together with the general effects of trust in the performance of the Grid systems, will be analyzed and presented in the next chapter.

Chapter 6

Implementation and Evaluation

"There is no such thing as a failed experiment,
only experiments with unexpected outcomes."
Richard Buckminster Fuller

6.1 Introduction

In this chapter, the simulations and measurements performed for obtaining a quantitative assertion of the effects of trust in the performance of Grid environments and the performance of the verification strategies and sub-strategies presented in the previous chapter will be presented.

6.1.1 Design

The experimental infrastructure is based on the GridSim Toolkit [27]. The basic functionalities and the architecture of this simulation toolkit, together with the undertaken modifications, are presented in Appendix B.
Figure 6.1 shows the flowchart of the simulation process as used in the experiments.

In the GridSim toolkit, a desired number of GridSim users (consumers) create a number of Gridlets (which represent the GridSim jobs), each with average Million Instructions (MI) (which represent the Gridlet length), allowed deviation percentage of the MI, granularity time of the simulation, overhead processing time per Gridlet (Gridlet overhead processing time). Additionally, any desired number of resources (providers) are created. Considering the fact that each "Grid entity" has the potential to play both roles (consumer and provider), for the experiments each of the users configures also a resource to which other consumers in the environment could send their Gridlets for processing.
Each of the consumers and providers specifies their initialization strategies together with the verification strategies and sub-strategies. In a second step, the GridSim entities, and parameters are initialized.
Details on Grid resources are obtained from a file containing a list of resources with their characteristics (total PEs - processing elements and MIPS - million instructions per second for each PE).

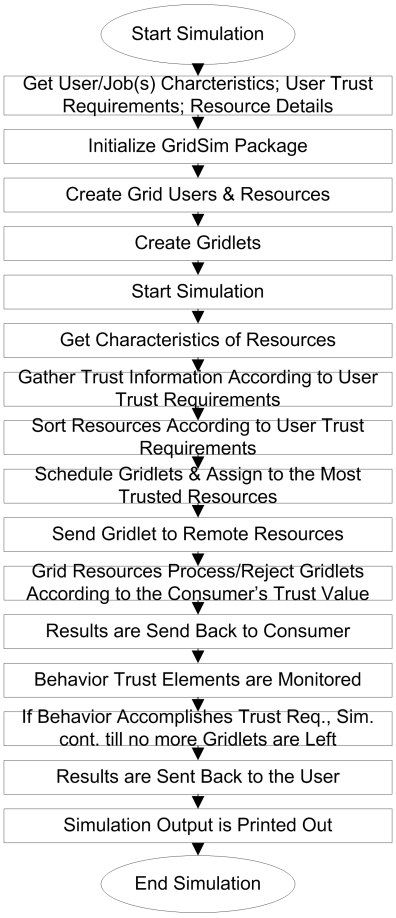


Figure 6.1: Flow of the Simulation Process.

Then, the GridSim simulation is started. Each of the experiments is specified within a class object. Here, the Grid user and the Gridlets are created for the simulation. The user specific trust requirements and the Gridlets are created based on user specified parameters (total number of Gridlets, average MI of each Gridlet and the deviation percentage of the MI).
First, the characteristics of the available Grid resources created in the resource creation section in the system are gathered. The Grid user, according to this information, sorts out the most suitable resources that could be used to fulfill its needs. Based on the user trust requirements, personal experience gathered during previous collaborations and experience of the others in the environment, a second sorting of the resources takes place. Only the most trusted of the suitable resources is going to be considered in the next collaboration. The scheduler, with help from the "partner evaluator" (Fig. 5.12 and Fig. 5.19), submits the Gridlets to the chosen resources. The Grid resources, following the same logic as in the consumers' case, decide if the user (consumer), according to the trust value it built on previous collaborations, could be served or not. If allowed, Grid resources process the received Gridlets and send the results back to the Grid user (consumer). Again, according to the verification strategy and sub-strategies, the collaboration is monitored from both partners. In the case of an "online" verification strategy, the monitoring takes place during the collaboration, and

in the case of an "offline" verification strategy, the monitoring takes place at the end of the collaboration process. The second strategy concerns only a limited number of behavior trust elements, e.g. on the consumers' side, the accuracy of the results coming from a certain provider and on the providers' side, the behavior trust elements such as "on-time payment" or "due payment", as showed in the chapter 3.
The processed Gridlets are gathered from simulated network through the I/O port or queue. Finally, the details (text and graphical) of the simulation, concerning the execution statistics (Gridlets execution time, status, etc.) and trust evolution are displayed.

Figure 6.2 gives a detailed view of the integration of the trust model presented in this

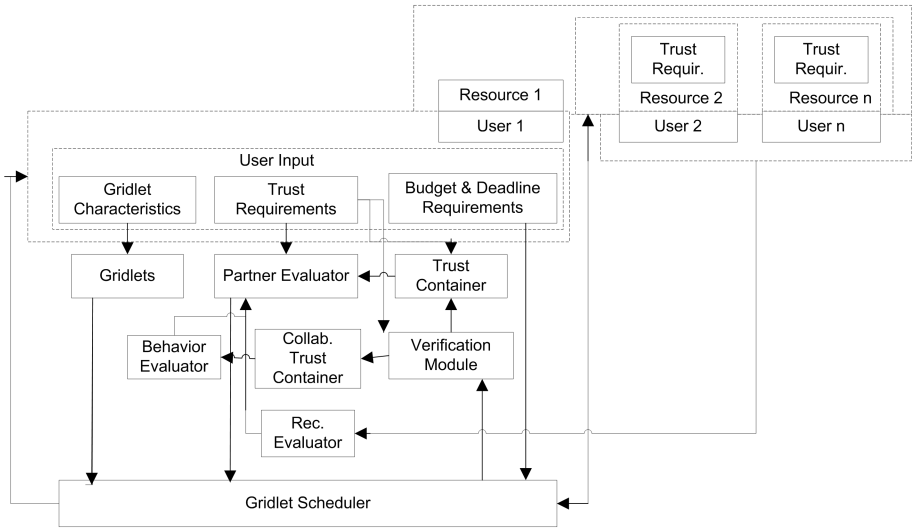


Figure 6.2: Integration of the Trust Model into the GridSim Toolkit.

thesis into the environment offered by the GridSim toolkit. The user creates the jobs with different characteristics and defines the trust requirements for the very next collaboration. The Partner Evaluator according to the trust requirements and the experience of the previous collaborations (either personal or that of third parties) sorts out the most trusted providers from the list of available and suitable resources. The scheduler in turn prepares the scheduling strategy and sends the jobs to the resources "recommended" by the partner evaluator. According to the verification strategies and sub-strategies, the collaboration is monitored and the trust values of the respective resources used are updated. The user receives the results back at the end.

6.1.2 Implementation

Figure 6.3 shows the Graphical User Interface (GUI) used for managing the experiments. More details on the implementation are given in Appendix B

The main classes involved in the simulations are presented in Fig. 6.4 (more details can be found in Appendix B).

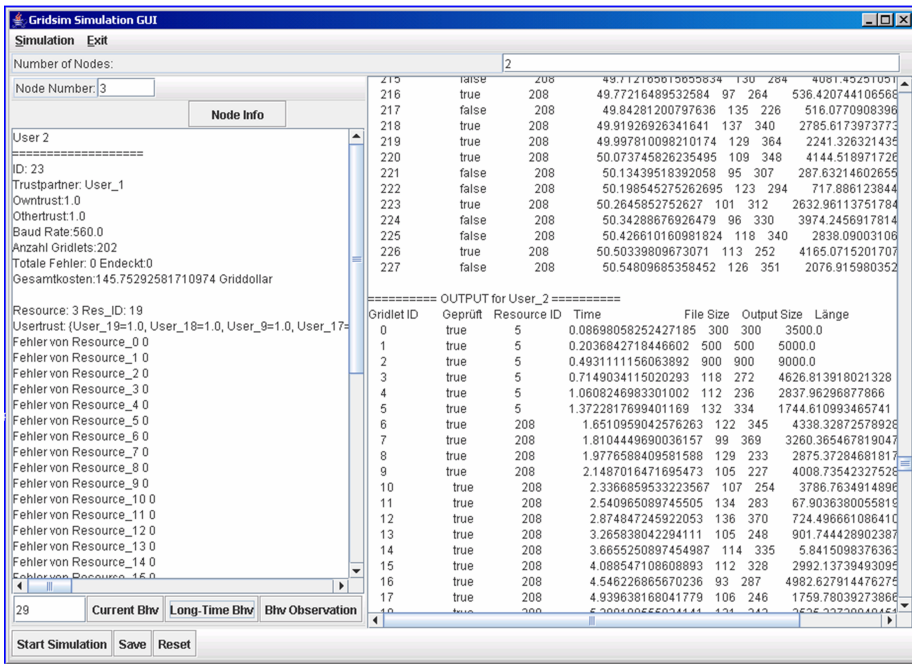


Figure 6.3: Graphical User Interface for the Simulation Scenarios.

Furthermore, the user can choose to perform the simulation with or without involving trust in the system. This facilitates the comparison of the output between conventional simulation scenarios and simulation scenarios where partners involved consider the notion of trust[1] while collaborating with each other. A coarse-grained view of the input and output of the simulation system is given in Fig. 6.5.

6.1.3 Simulation Scenarios

As previously mentioned, this chapter will concentrate on:

- evaluating the performance of the trust model during establishment of trusted collaborations between parties, monitoring of the collaboration and discovery of short-term or long-term deviations on the behavior of the collaboration partners,
- evaluating the effects of trust in the performance of the Grid systems and
- helping the Grid participants to tune their trust requirements to their needs and capabilities.

[1]Here, the personal definition for trust given at the chapter 3 is implied.

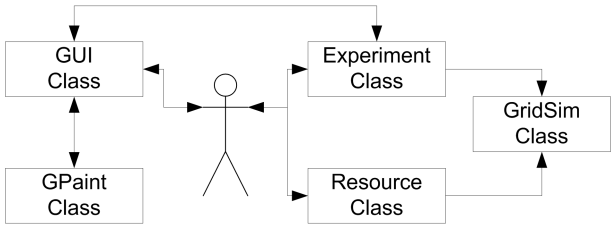


Figure 6.4: Main Classes Involved During Simulations.

Considering the analysis done on the application/user requirements in subsection 5.5.1, the main situations simulated deal with moderated and high trust requirements. The aim of the experiments is to observe:

- trust establishment and development;
- detection of deviating behavior;
- behavior categorization;
- effects of trust processing time, cost and network load.

A set of resources and users using is modeled using GridSim where:

- each user creates/owns a resource. The idea is to show that in real Grid environments, participants could play both roles; consumers and providers of services.
- each user sends his/her tasks to every resource in the environment (considered to be suitable), except for its own resource.
- the behavior trust elements considered are the accuracy of the responses coming from the different resources, their availability, accessibility, and speed of processing.

The simulation scenarios can be summarized as follows:

Trust Establishment and Development. Here, the normal course of trust establishment and development among Grid participants will be observed. Participants (consumers or providers) either new to the environment or with a certain personal experience are going to sort their collaboration partners on the basis of identity and behavior trust considerations; personal and/or third parties' experience; moderated or high trust requirements. Furthermore, the influence of trust on the duration of the completion of users' requests, on the budget foreseen by the user and on the network load is going to be investigated.

Trust Development under the Presence of Cheating. In every Grid environment (and any other collaborative environment), many participants, either intentionally or under

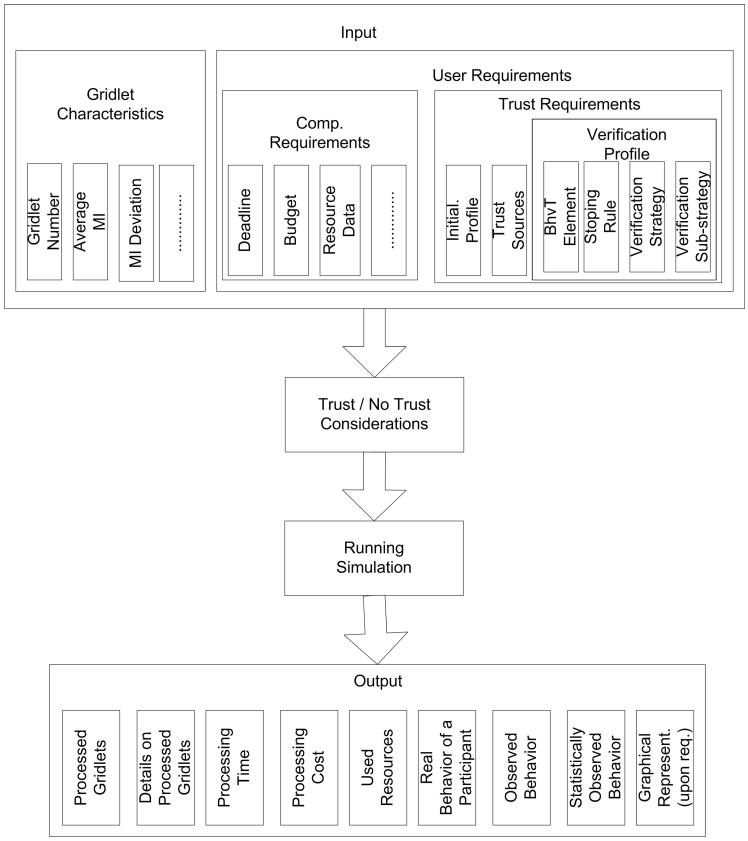


Figure 6.5: Input and Output of the Simulation System.

the influence of outside factors, introduce deviations in their "normal" behavior. These deviations affect single behavior trust elements, any combination or the entirety of them at different frequencies or completely at random. Here, the performance of the trust model in general and more specifically, the performance of the verification strategies and sub-strategies in detecting the "real" behavior of a participant within an active collaboration will be observed.

"Malicious" Recommenders. In the presented model, either in the absence of information gathered directly on a specific participant, or upon the user's request, it is possible to make use of the recommendations the other participants in the environment are willing to share. According to the weight assigned to recommendations, every participant establishes their influence on its personal decision on the collaboration with a certain participant. As such, it is possible that "malicious" recommenders intentionally offer low trust values for "good" participants and high trust values for other "malicious/mediocre" participants.
The negative effects of such "malicious" recommendations, together with the efficiency of the presented trust model in minimizing them will be observed.

From "High" to "Low". As discussed in subsection 3.8.2, it has to be expected that many selfish or malicious participants behave properly once introduced in the environment

or at the beginning of the collaborations. Thus, they fulfill the expectations of their interaction parties and offer services of high quality, the reason for which they were chosen among the others in the environment. As soon as they have reached a "high social position" in the environment, they start displaying their real purposes, lowering the quality of the offered services. Here, again, the performance of the trust model in general and more specifically the performance of the verification strategies and sub-strategies in detecting short-term and long-term deviations in the behavior of participants that keep a "high social position" and preventing such behavior for future collaborations will be observed.

From "Low" to "High". A "low social position" could be attributed to a participant for different reasons:

- the participant entered the environment with limited capabilities,
- outside factors prevented the participant to behave properly,
- the collaboration party established a "prejudiced" collaboration at the very beginning (initial trust value 0 or very small) although the participant was able to fulfill its "high" trust requirements.

It is interesting to observe the development of trust of such participants to their collaboration partners, if they improve the quality of the services they offer (upgrade personal capabilities or minimize outside inference) or continue offering services of high quality (in the case when their counterpart "prejudiced" its capabilities).

Trust Effects. The presence of trust in Grid environments will definitively improve the quality of the collaboration among participants. However, it is impossible to develop trust for nothing. The disadvantages of trust involvement and of different verification strategies and sub-strategies will be presented in terms of runtime, computational costs and overhead.

6.2 Establishing and Managing Trust Among Grid Participants

Trust establishment and development are two issues strongly related to each other. A Grid participant will establish a collaboration with the most suitable partners which fulfill its trust requirements. At the same time, trust information (developing trust through collaboration monitoring) is going to be accumulated with respect to only active partners in the current/last collaboration.

6.2.1 Trust Establishment

Trust establishment is influenced by the following factors:

Initial Trust Value. The initial trust value expresses:

- the disposition of a trustor to trust its trustee(s) in the environment in the absence of the personal trust information regarding them,

- the scale to which the trust value resulting from the combination of trust values coming from different sources will be compared. If the combined trust value equals or is greater than the initial trust expressed by the trustor, than the target participant can be considered further to establish a collaboration. On the contrary, if the combined trust value is smaller than the initial trust expressed by the trustor, it is going to be discharged.

Personal Experience. It is based on the direct trust (as introduced in 5.2.8) that a trustor has previously established with its trustees.

Third Parties' Experience. It is based on the indirect trust (as introduced in 5.2.9) that a trustor gathers from different trust sources other than its own personal experience.

Weights for Personal and Third Parties' Experience. While deciding on its trustees, each trustor, according to the formulas (5.9), (5.10) and (5.11), decides on the percentage each of the trust sources (personal experience and third parties' experience) will influence this decision. The weights for the experiences during the experiments are:

- personal experience with 0.0 and third parties' experience with 1.0. Thus, the entire decision on establishing a collaboration with a target participant is based only on the recommendations a Grid participant obtains. Here, the case when the user/consumer does not have any personal experience at all, is also implied,
- personal experience with 0.2 and third parties' experience with 0.8,
- personal experience with 0.5 and third parties' experience with 0.5 (the presented graphics in this sub-section are referred to this combination of weights),
- personal experience with 0.8 and third parties' experience with 0.2,
- personal experience with 1.0 and third parties' experience with 0.0. The participant makes use only of its personal experience, discharging any offered recommendation.

Identity Trust of the Target Participant. Making use of the oracle of certification (as introduced in 5.2.5), each trustor has the possibility to express a trust value for the identity of the trustees it communicates with.

Simulation Scenarios and Environment. During the experiments, 30 users and 30 resources were created. A user obtains at the same time also a resource and considers as possible partners, every resource in the simulation environment other than its own (practically 29 resources in total). Another feature of the experimental environment is the presence of concurrent users. Thus, during a simulation all the users try to establish a collaboration with one (or more) resources present in the environment that they consider as trusted according their current trust requirements. The purpose of such configuration is to come near to the real Grid environments, where the existence of concurrent consumers (multiple consumers competing for resources) is the most common phenomenon to be found. However, the graphics are built with the results gathered from a single GridSim user/consumer regarding the most suitable participants, those that fulfill its trust requirements.
The bars represent:

- the user's personal experience, obtained either through past collaborations or through the initial trust value,
- experience obtained from other active participants in the environment (third parties' experience),
- combined and weighted experience according to the expressed preferences and
- the percentage each of the suitable (capable and trusted) resources was used during the very last collaboration.

Combined and weighted experience bars show how personal and third parties' experience influence trust establishment between trustor and a trustee.
The bars on the percentage each of the trustees was used show the load distribution among collaboration partners.

Three scenarios were considered:
In the first scenario, a Grid user/consumer tries to establish a communication with everybody in the environment that fulfills its moderated trust requirements. The user does not express any preference regarding the identity trust of its future communication partners and assigns an initial trust value equal 1.0 to those participants with whom it has previously not communicated with. The three sub-scenarios describe the situations where:

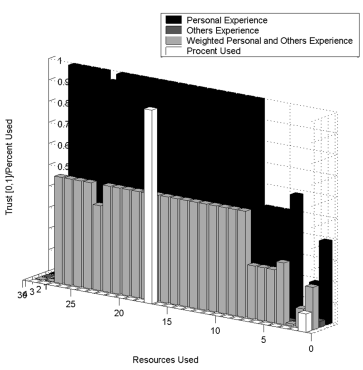
1. recommendations are accepted from everybody else in the environment that has previously had any experience with the target participant;
2. recommendations are accepted no more from everybody in the environment but only from the direct partners of a participants' partners, and
3. recommendations are accepted only from the direct partners.

The results are presented in Fig. 6.6. The different modelled resources are listed along the X axis and the Y axis presents the level of trust the participant gathered through the direct and indirect experiences, together with the percentage of tasks distribution among the different resources.

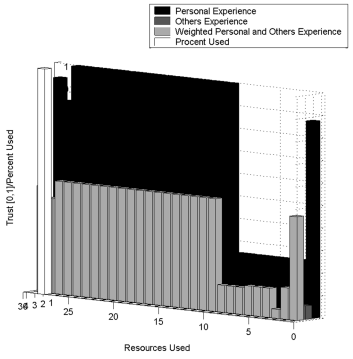
The tendency of the trustor during the simulated sub-scenarios is to not make use of the most trusted partner, but to collaborate with the most capable one(s). Thus, from all available and accessible participants in the environment, the fastest of them was considered:

- in a. and c., no previous personal experience exists for the partners considered, and
- in b. other partners were most trusted.

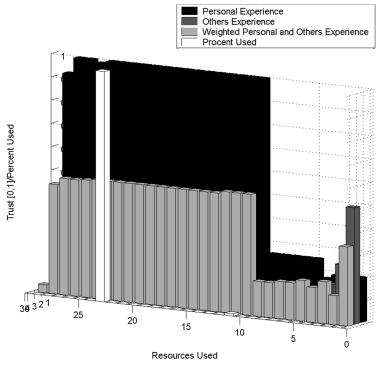
The results reflect the open attitude of the trustor toward every possible collaboration partner in the environment that fulfills its minimal trust requirements. As a result of the high concurrency in the environment, only a minimal number of resources (1 to 5) were used in total from a single trustor during the entire collaboration. However, most of the tasks were completed only by a single resource.



a. recommendations from everyone



b. recommendations from direct partners and their direct partners



c. recommendations from direct partners only

Figure 6.6: Trust establishment while considering everyone as a communication partner and assigning an initial trust value 1.0.

In the second scenario, the user/consumer keeps quite the same moderated trust requirements. The difference to the first scenario is that it chooses to collaborate only with those partners whose identity trust is at least 0.5, meaning that the trustor chooses to collaborate with those partners in the environment which has obtained a certificate either directly from the first order CAs or at least from a participant who has.
Once again, the initial trust assigned to unknown partners is 1.0 and the trustor accepts recommendations from:

- everyone,
- known partners of its direct partners and its direct partners only and
- its direct partners only.

Just like during the simulation of the first scenario, the same open attitude of the trustor toward every possible collaboration partner in the environment that fulfills its minimal trust

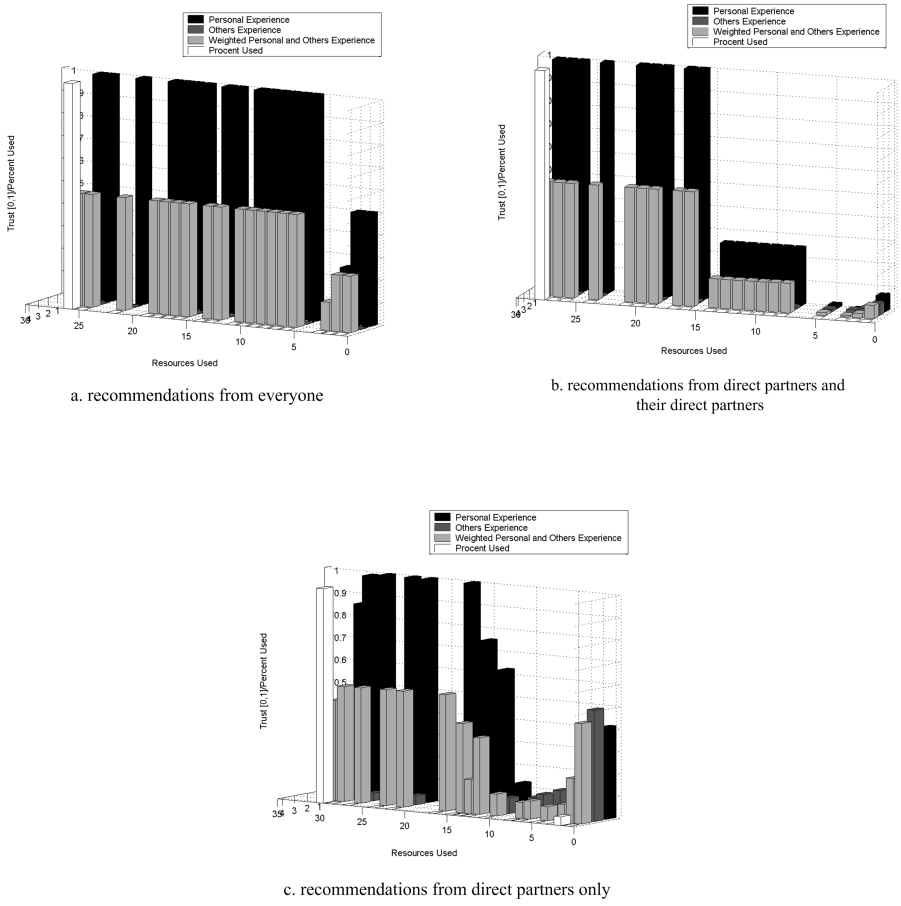


Figure 6.7: Trust establishment while considering those participants who have an identity trust at least 0.5 and an initial trust value 1.0 is assigned.

requirements is reflected.
The only distinct feature is the decrease of the number of possible partners to establish a collaboration with as a result of applying an identity trust of at least 0.5 during the simulations.
The behavior trust elements under observation were availability, accessibility and speed of processing. In the simulation environment for the first three scenarios, none of the resources injected errors on the responses they sent back. The resources were available during the entire collaboration and their accessibility was conditioned from the concurrent requests each of the users made during a simulation. The speed of processing was not artificially influenced. The simulator itself created resources based on a singular principle, the later the resource was created, the higher their speed of processing. Observing the graphics in Fig. 6.7, it can be derived that more capable resources exist in the environment other that those chosen. Thus, sorting the trustees based on the trust that the trustor places on its identity has some side effects, more precisely the number of the available partners in the environment that can be

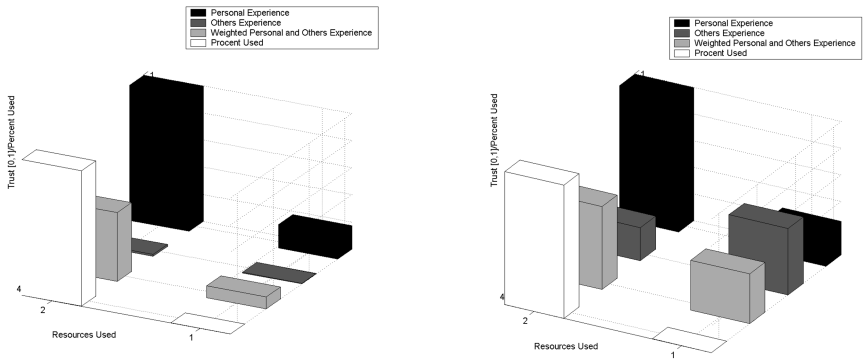
considered during the upcoming collaboration (wrongly) reduces. The same effect, but in a more extended scale was observed during the simulation of the third scenario.

In the third scenario, the user/consumer tries to establish a collaboration only with those participants who:

- have an identity trust 1.0, thus, the trustor chooses to collaborate only with those partners in the environment which have obtained a certificate directly from the first order CAs and
- total trust, weighted personal and recommendations, is at least 0.5 (initial trust value is also 0.5).

In this scenario, recommendations are accepted from:

- known partners of its direct partners and its direct partners only and
- its direct partners only.



a. recommendations from direct partners and their direct partners

b. recommendations from direct partners only

Figure 6.8: Trust establishment while considering those participants who have an identity trust of 1.0 and a total trust of at least 0.5.

In these simulations, the common characteristic observed is the tendency of the trustor to establish a collaboration not only with the most capable of the active participants in the environment but they should also be the most trusted ones. However, the number of the partners to collaborate with reduces drastically since the majority of the participants in the environment do not comply with the trust requirements of the trustor in this scenario.

6.2.2 Trust Development in the Absence of Malicious/Mediocre Behavior

The development of trust between participants is strongly related to trust establishment. Once a resource is considered as suitable and trusted for establishing a collaboration with, the user sends the tasks there and monitors the entire collaboration process.
An example of the output of such a monitoring process is presented in Fig. 6.9.

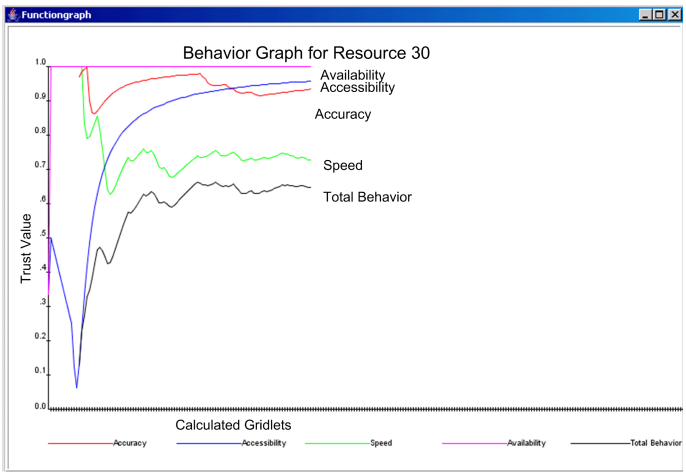


Figure 6.9: Measuring Behavior Trust Elements (no identity trust involved while calculating the Absolute Behavior Trust (ABhvT)).

Absolute Behavior Trust is calculated as the production of single behavior trust elements under observation. It is only a demonstration of the observed behavior trust elements. However, for the target trustee can be said that:

- it was available during the entire collaboration,
- initially there were some problems for the trustor to access it (most probably as a result of the concurrent requests from other users/consumers in the environment),
- some malicious behavior was observed during the collaboration (erroneous responses were sent back to the trustor; does not reflect the "real" behavior of the trustee), and
- speed of processing does not comply with the trustee's "promise" at the beginning of the collaboration.

The scenario is pretty simple. A GridSim user with trust requirements as those presented in Fig. 6.6 (moderated trust requirements, weights for personal and third parties' experience are 0.5) interacts between different experiments with many of the active resources in the environment. Thus, tasks are assigned to some of the participants considered as most trusted, according to the expressed trust values, and responses are collected once the tasks are completed. All the resources are considered to give back correct responses (the assigned tasks are completed correctly; there does not exist any malicious behavior in the environment) and that the resources are always "online" (available). Behavior trust is determined based on only two behavior trust elements: speed and accessibility. In each experiment, the user sends an average of 100 jobs. The measurement of speed and accessibility take place every time a job is sent/processed to/from a resource. The same frequency is also applied for monitoring

the availability, while accuracy, as it will be seen in the next subsections, will be observed according to the verification strategies and sub-strategies specified by the user and presented in 5.6.2 and 5.6.3.

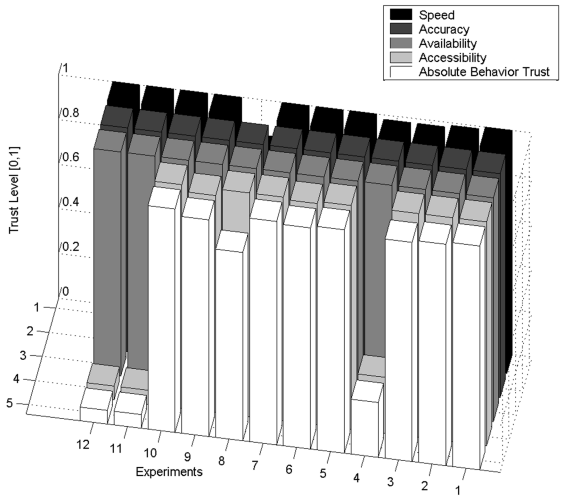


Figure 6.10: Personal experience of User 4 regarding the behavior observed for Resource 2 over 12 experiments (no identity trust involved).

The results of the observations, presented in Fig. 6.10 and Fig. 6.11, regard the perceived behavior trust for Resource 2 and Resource 30 from User 4 and User 1, respectively. The identity trust of the two considered resources is not involved in the calculation of the total trust. The X axes lists the different behavior trust elements under observation, through a certain number of experiments. The trust values of these elements, together with the Absolute Behavior Trust (ABhvT) are presented at the Y axis.

Both resources (Resource 2 and Resource 30) were available during the entire collaboration and introduced no errors (at least this conclusion was derived from the observations of respective trustors, User 1 and User 4). The oscillations on the ABhvT come from:

- the oscillations on the accessibility of the two considered resources. It can be explained with the fact that no resource is strictly dedicated only to a single user. They can serve other users as well, either simultaneously (unfortunately not possible in the GridSim simulator) or in between (while the user continues to send its tasks).
- the oscillations in their speed of processing. Although not desired and not configured to be part of the resources' behavior, they are the best example of the indirect influence that other factors could have in the behavior of a Grid participant (network problems, malicious attempts from third parties to prevent the normal functioning of a target participant, etc.).

The graphics in Fig. 6.12 and Fig. 6.13 represents the same observations as in Fig. 6.10 and Fig. 6.11 with the only difference that the Absolute Behavior Trust (ABhvT) is no more

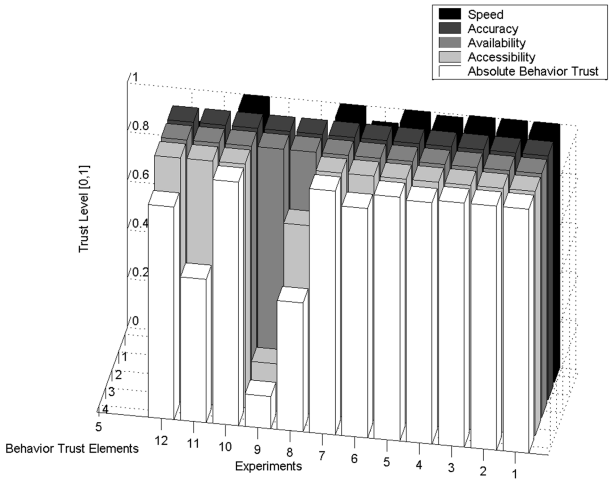


Figure 6.11: Personal experience of User 1 regarding the behavior observed for Resource 30 over 12 experiments (no identity trust involved).

shown in these graphics. Instead, the total trust calculated according to the formula (3.1) is presented.

Resources show a similar behavior as in Fig. 6.10 and Fig. 6.11. The difference with that scenario is the consideration by User 1 and User 4 of the respective identity trust for Resource 30 and Resource 2. Using a similar certification chain as the one presented in Fig. 5.1, the identity trust that User 4 calculates for Resource 2 is $T_4^I(2) = 1.0$ and the identity trust that User 1 calculates for Resource 30 is $T_1^I(30) = 1.0$. As a result, the total behavior observed for Resource 30 is less appraised by User 1.

A better view for this collaboration scenario (User 1 - Resource 30; some malicious activity (lack of 100% accuracy) was introduced in the behavior of the Resource 30) is presented in Fig. 6.14 and Fig. 6.15.

Further simulations were conducted in order to observe:

- the development of the personal experience of a single user, regarding all its possible collaboration partners over a number of experiments, and
- the development of the third parties' experience that a single user obtains in regard to all its collaboration partners over a number of experiments.

The obtained results for the development of the personal experience of a GridSim user are presented in Fig. 6.16. As in the above simulation scenarios in this subsection, ABhvT of a resource is primarily determined from their speed of processing and their accessibility.
The user starts with an initial trust value of 1.0 and tends to communicate with the most capable of the resources. If for a certain user these resources result as available but not accessible, the corresponding trust value is updated and the user tries to send the tasks to

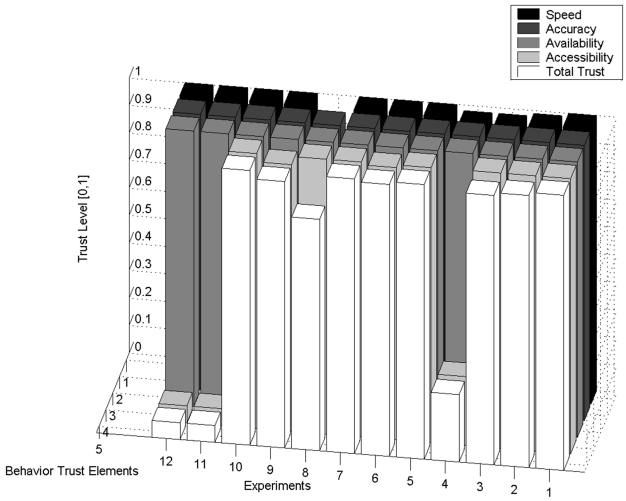


Figure 6.12: Personal experience of User 4 regarding the behavior observed for Resource 2 over 12 experiments (identity trust involved).

other resources in its list of capable and trusted resources.
During the simulations, the user makes a rich experience in the collaboration with the active resources in the environment. In total, 29 resources were created and 15 simulations were sequentially run. The axes in Fig. 6.16 show the trust (Z axis) that a single trustor built for all the other participants in the environment (Y axis) throughout the simulations (X axis). From the observations it can be concluded that:

- even with the very best intent of a participant to offer the maximum of its capabilities, it is quite impossible that the trust value remains (if "warm start" with initial trust value 1.0) at, or reaches (if "cold start" with initial trust value 0.0) its maximal value 1.0, and
- a higher constant trust value for a target participant does not always reflect the observed behavior regarding that participant. The trustor could have simply not established a collaboration with that partner at all or in a long time.

The development of the third parties' experience that a single trustor obtains in regard to all the other participants in the environment, over a number of sequential simulations is presented in Fig. 6.17, Fig. 6.18 and Fig. 6.19. The development of the recommendations follows the same logic as the development of the personal direct experience of a user. Once again, X axis and Y axis show the resources/trustees in the environment and the sequential simulations respectively. The trust regarding a target trustee that the trustor receives as a recommendation from all the others in the environment (calculated according formulas 5.5, 5.6 or 5.7) is represented in Z axis.
The continuous decrease of the obtained values in the three simulated scenarios (Fig. 6.17, Fig. 6.18 and Fig. 6.19) are principally because of the different experiences each of the participants in the Grid environment makes, according also to the very personal trust requirements, configurations, degree of concurrency (negative influence in the accessibility) and the influence that other outside factors could have in some of the behavior trust elements (i.e. speed

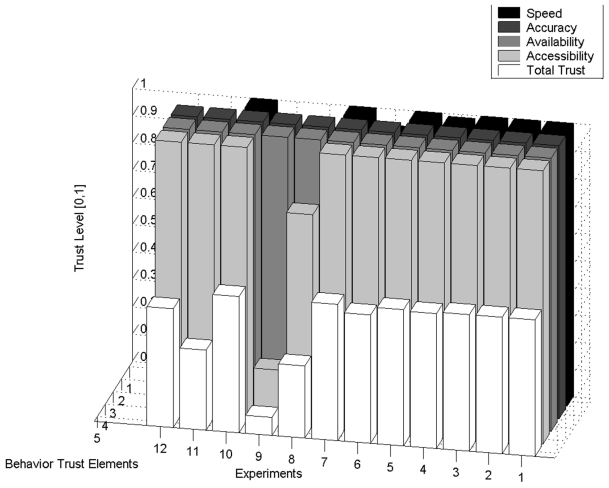


Figure 6.13: Personal experience of User 1 regarding the behavior observed for Resource 30 over 12 experiments (identity trust involved).

of processing).
The smaller the number of recommenders, the higher the trust values received from the trustor. However, no warranties can be offered on their objectivity. This problem will be treated in subsection 6.3.4.

6.2.3 Partial Results on Trust Establishment and Development

Here, the most important conclusions derived from the simulations run for observing trust establishment and development will be briefly summarized:

- Recommendations could many times offer a non-correct judgment on the real capabilities of a participant. According to the personal trust values that everyone specifies on a per case basis, the resulting personal experience with a target participant is built differently between different Grid participants. In the case when recommendations are accepted from everyone, the greater is the risk of obtaining non-decent information on the collaboration partners. Recommendations could be considered as important only at the beginning of a collaboration between partners. Later on, the only trust information a user has to make reference to is its personal trust developed through monitoring the collaboration process with the target participant.

- Involving the identity trust adds some prejudice in the capabilities of a target participant, but will definitively help in placing some order in the Grid environments in general.

- Logically, the more closed the attitude of a Grid participant while choosing its interaction partners is:

  – the less it has to make use of the third parties' experience,
  – the more will be the importance of the personal experience.

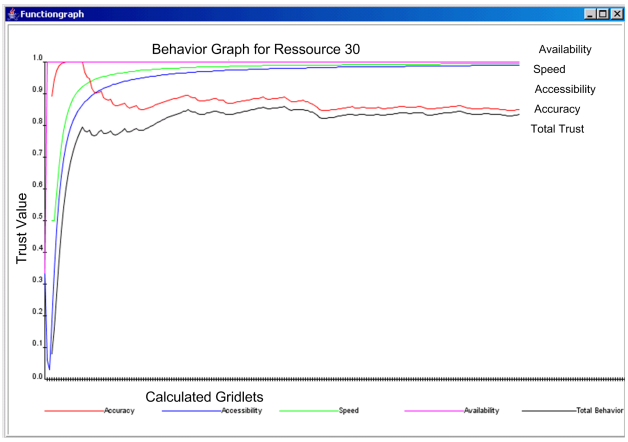


Figure 6.14: Behavior Trust of a Collaboration Partner when Identity Trust is not Involved.

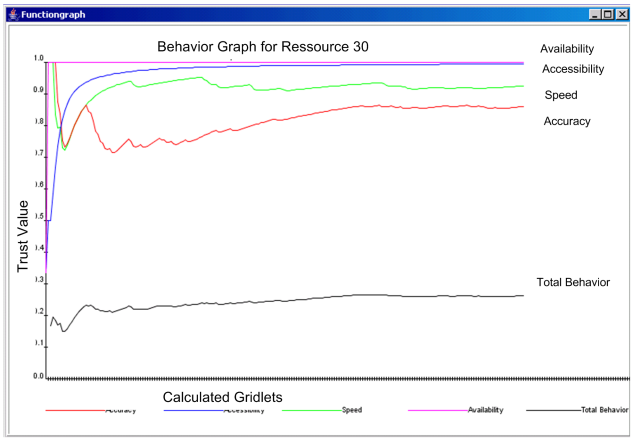


Figure 6.15: Behavior Trust of a Collaboration Partner when Identity Trust is Involved.

- Trust reaches and stays never at the exactly maximal possible value 1.0.
- A constant trust value, either low or high, does not necessarily imply that the target participant continuously behaves "bad" or "good". The gathered results showed that the partner was most probably not used at all over a series of experiments.
- The claim done by Azzedin et al. in [82], [79], [81], and [80] and supported later on by von Alunkal et al. [74] that *as time passes, if no more collaborations have found place among participants, trust decays*, has not any "real" background and must in no way be converted into a rule. The experiments demonstrated that although a GridSim resource was not used over a number of simulations by a certain GridSim user, its trust to the other participants in the environment (behavior showed during the collaborations with them) did not necessarily diminish; it developed normally either to higher or even lower values as well.

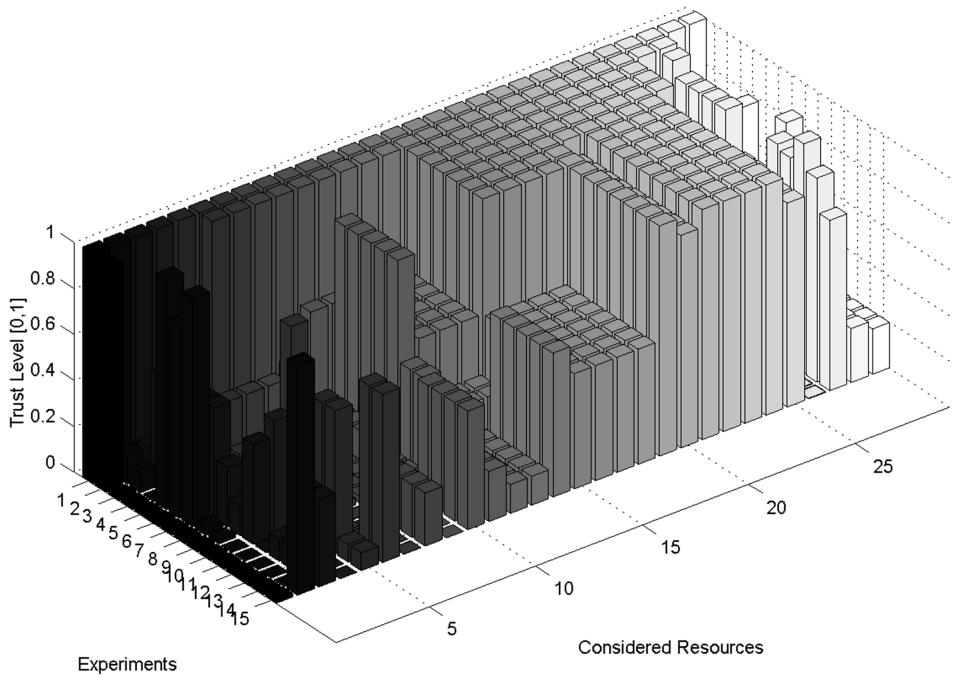


Figure 6.16: Development of the personal user experience regarding its collaboration partners over different simulations.

6.3 Measuring the Performance of the Trust Model

The performance of the trust model is evaluated as below:

- First, the mean absolute error regarding the real behavior shown by a participant to the observed behavior (either through applying the verification model only or making a double check through applying the statistical model) will be measured. For this purpose, the following formula is considered:

$$MAE = \frac{\sum_{i=1}^{n} |Bhv_{observed}(i) - Bhv_{real}(i)|}{n} \tag{6.1}$$

  where n is the number of sequential experiments.

- Second, the behavior of a participant through different sequential experiments will be monitored in order to show possible fluctuations in the current behavior of a participant compared to the behavior previously shown.

- Third, the effects of inaccurate third parties' recommendations (intentionally high or intentionally low) to the general decision of a trustor to collaborate with its trustees will be assessed.

The simulation scenario is similar to the trust development: according to the trust requirements, a number of trusted participants is sorted out. Tasks are assigned to trusted partners and responses are collected once the tasks were completed. The entire collaboration is

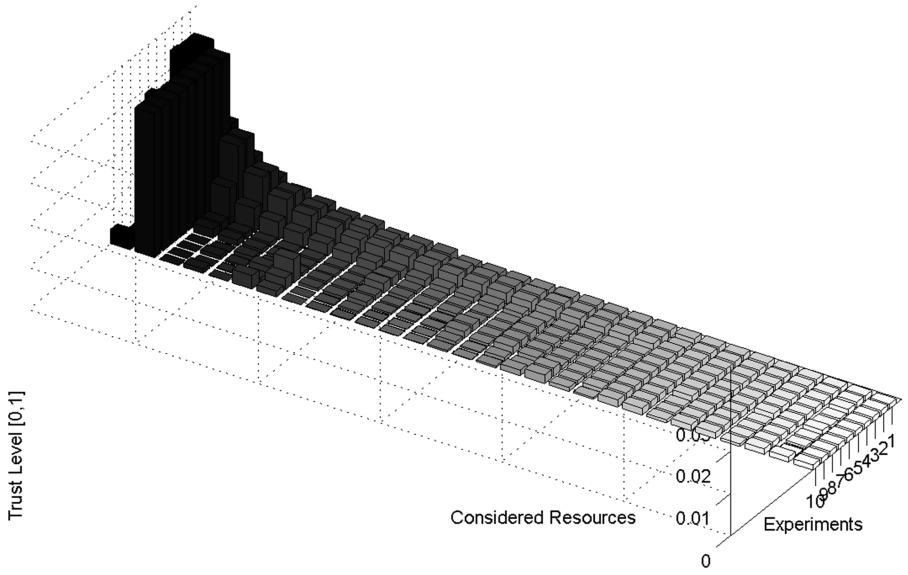


Figure 6.17: Development of the third parties' experience a participant obtains over different simulations; Recommendations are accepted from anyone in the environment.

monitored using different verification strategies and sub-strategies. This time, errors were introduced in the responses a trustee sends back to the trustor.

6.3.1 Mean Absolute Error

Measuring the Performance of the Verification Model. Here, the performance of both "Offline" and "Online" verification strategies will be assessed.

- **"Offline" Verification Strategy:** the verification frequency is calculated according to formula 5.6, after the user establishes the minimal desired verification frequency. As minimal verification frequency, the values $V_{min} = 5\%$, $V_{min} = 15\%$, $V_{min} = 30\%$ and $V_{min} = 60\%$ are considered. A "warm start" initialization strategy (initial trust value 1.0) is applied. The trustor sends to its trustee between 50 and 200 tasks. The trustee randomly introduces errors in the responses it sends back to the trustor. The error frequencies vary between 5%, 10%, 30%, 60% and 100% of the completed tasks sent back to the trustor. The verification takes place after the tasks are processed and ready to be sent to the trustor. The performance of the verification process, for this verification strategy and for the considered error rates (5%, 10%, 30%, 60% and 100%) is shown in Figure 6.20 (a, b, c, d), where the mean absolute error (MAE) calculated according to formula 6.1 is presented in Y axis for different number of tasks assigned to the trustee (X axis). Initially, for all of the verification frequencies, the mean absolute

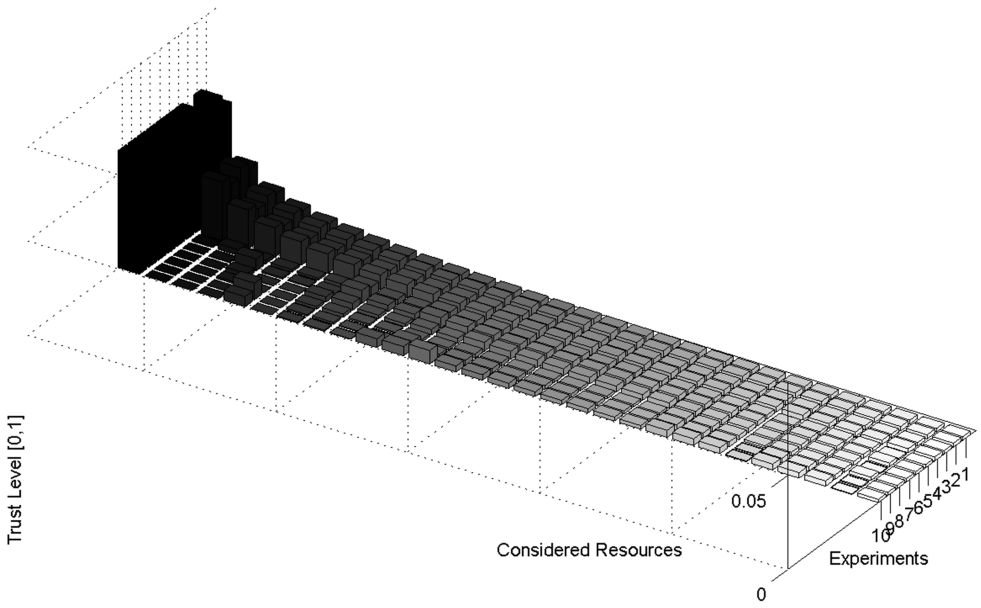


Figure 6.18: Development of the third parties' experience a participant obtains over different simulations; Recommendations are accepted from known partners of its direct partners and its direct partners only.

error increases until a certain point and then develops quite constantly. The increase can be easily explained by fact that the trustor needs some time until it realises that its trustee is sending back erroneous responses. The constant development of the MAE has to do with the capability, or better say incapability, of the trustor to discover these erroneous responses. From the graphics it can be seen that the greater the number of tasks the trustor and the trustee exchange, the more possibility has the trustor to observe errors in the behavior of a trustee and as a result the smaller the mean absolute error is. Logically, the mean absolute error diminishes also as the minimal verification frequency increases.

- **"Online" Verification Strategy:** the verification frequency is calculated according to formula 5.8, after the user establishes the minimal desired verification frequency. Once again, as minimal verification frequency, the values $V_{min} = 5\%$, $V_{min} = 15\%$, $V_{min} = 30\%$ and $V_{min} = 60\%$ are considered and a "warm start" initialization strategy (initial trust value 1.0) was applied. The number of tasks varies between 50 and 200 tasks. The trustee randomly introduces errors in the responses it sends back to the trustor. The error frequencies vary between 5%, 10%, 30%, 60% and 100% of the tasks. The trustor establishes also the "clearance number" (number of tasks to be verified sequentially). The simulations were run for clearance numbers 0, 25 and 75. The verification takes place as the collaboration between parties continues. The performance

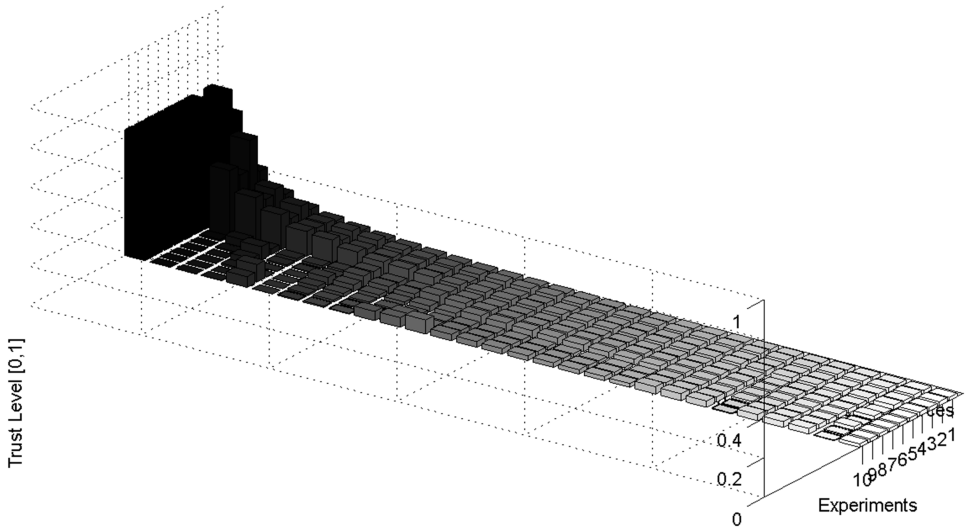
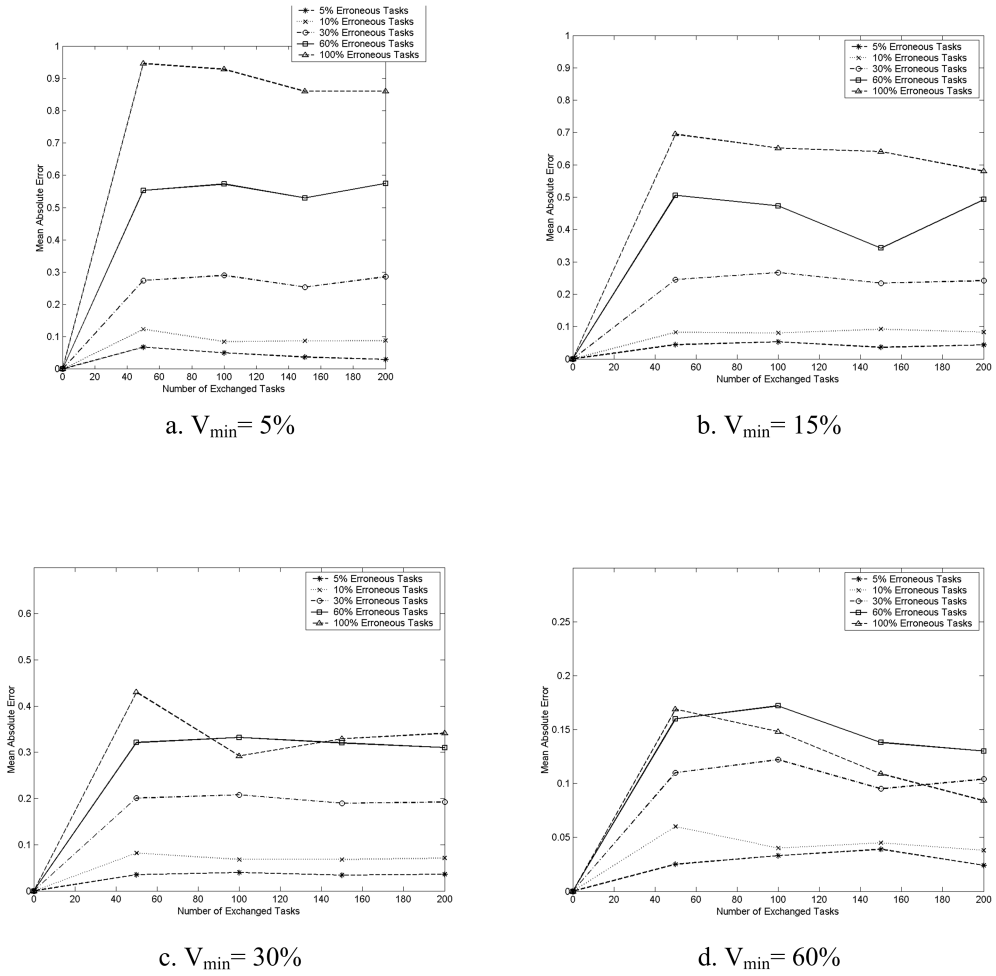


Figure 6.19: Development of the third parties' experience a participant obtains over different simulations; Recommendations are accepted from known direct partners only.

of the verification process for this verification strategy is shown in the following:

- **clearance number equal 0:** In this case, the performance of the "Online" verification strategy equals the performance of the "Offline" verification strategy presented in Figure 6.20.
- **clearance number equal 25:** The performance of the verification process for this verification strategy is shown in the Figure 6.21 (a, b, c, d). Introducing a sequential verification of the responses coming from the trustee at the beginning of the collaboration eases the identification of any "malicious/erroneous" activity in its behavior at the very first stages. The best example is the development of MAE for a trustee that introduces 100% erroneous responses. The more tasks the trustor exchanges with this trustee, the greater the experience the trustor gathers. Since the verification frequency (5.8) is tightly coupled to the trust values a trustor develops regarding its collaboration partners, the frequency of the verification increases in this scenario, because the trust value diminishes. It is normal to expect such fluctuations of the MAE values for high clearance numbers and high percentages of errors in the incoming responses. From the graphics it can be seen that as the clearance number increases, the mean absolute error reduces (once again MAE is represented in the Y axis and number of tasks "exchanged" between parties in X axis).



a. V_{min}= 5%

b. V_{min}= 15%

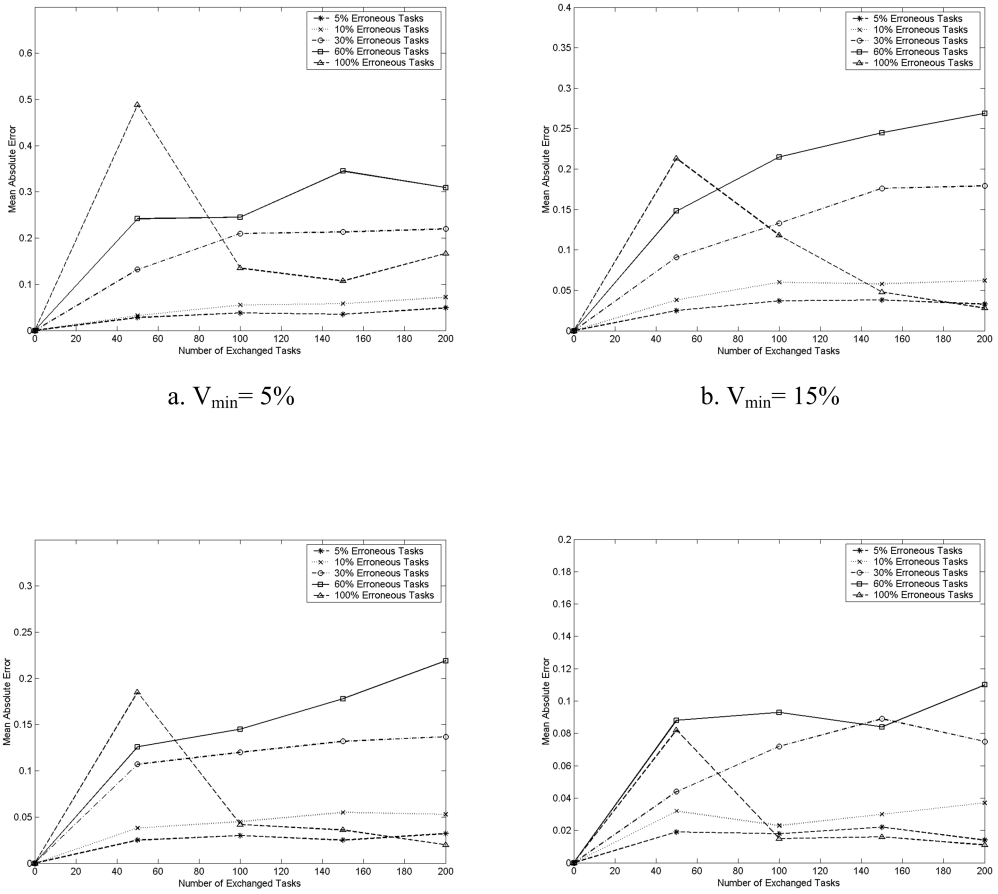
c. V_{min}= 30%

d. V_{min}= 60%

Figure 6.20: Mean Absolute Errors for the "Offline" Verification Strategy.

– **clearance number equal 75:** The performance of the verification process for this verification strategy is shown in Figure 6.22 (a, b, c, d). The number of tasks exchanged varies between 100 and 200.
From the graphics can be seen that the performance of the verification model is quite similar for all the minimal verification frequencies established from the trustor. Thus, the behavior a trustee showed while sequentially verifying its responses (clearance number equal or greater than 75) corresponded to the behavior it showed during the entire collaboration.

Measuring the Performance of the Statistical Model. Here, the performance of the statistical verification model will be assessed. The average quality level (AOQ) is also tightly coupled to the results that come out from the verification process. It is practicaly a double verification of the received results. It was calculated according to formula 5.25, after the



a. V_{min}= 5%

b. V_{min}= 15%

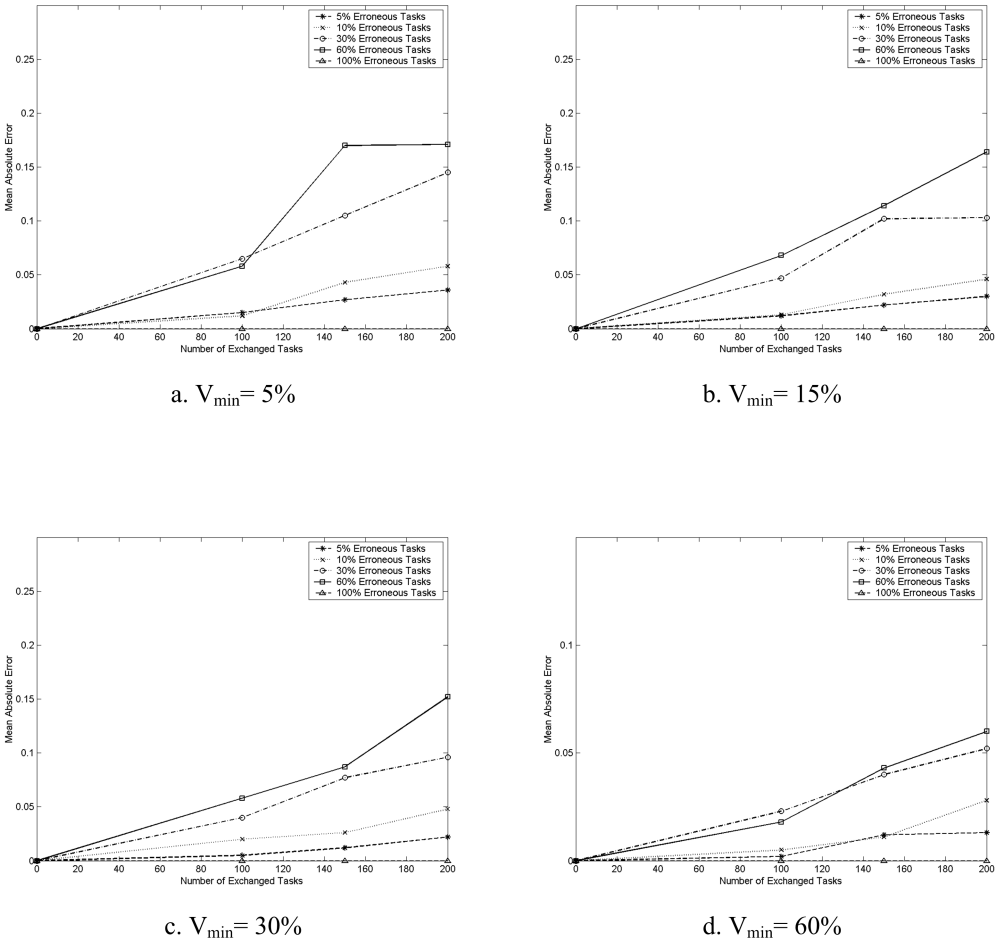
c. V_{min}= 30%

d. V_{min}= 60%

Figure 6.21: Mean Absolute Errors for "Online" Verification Strategy; clearance number 25.

verification took place. Since the sampling interval k (formula 5.20) varied according to the last updated trust value (formula 5.8), a mean value for the sampling interval is used. The "errors" found with this method were added to the previously found "errors" during the verification process. The fraction of the "total" number of erroneous tasks with respect to the total number of tasks coming from the single trustees was calculated in order to determine their behavior. In the graphics, fluctuations of the development of MAE, similar to those observed during the evaluation of the performance of verification model, are to be expected as well.

- clearance number equal 0: The performance of the verification process for this verification strategy is shown in Figure 6.23 (a, b, c, d). The decrease of the mean absolute error, as result of the double verification, is easily identifiable (comparing it also to the mean absolute error presented in Fig. 6.20 a, b, c and d respectively).



a. V_{min}= 5%

b. V_{min}= 15%

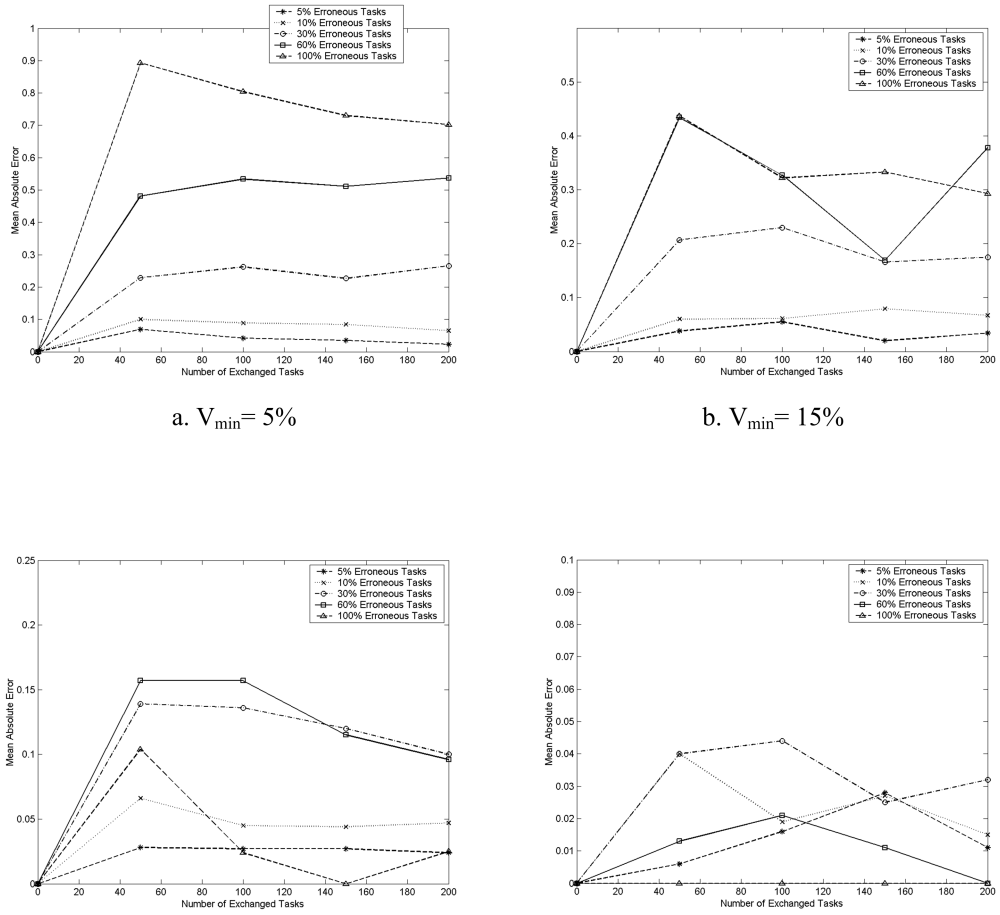
c. V_{min}= 30%

d. V_{min}= 60%

Figure 6.22: Mean Absolute Errors for "Online" Verification Strategy; clearance number 75.

- **clearance number equal 25:** The performance of the verification process for this verification strategy is shown in Figure 6.24 (a, b, c, d). The quality of the outgoing tasks (tasks that come back to the trustor) improves even further especially for the minimal verification frequencies and error rates under 100%.

- **clearance number equal 75:** The performance of the verification process for this verification strategy is shown in Figure 6.25 (a, b, c, d). In the graphs it can be easily identified that *the higher the clearance number (as in this case 75 or higher), the more similar is the performance showed from the statistical model with the performance of the verification model only* (Fig. 6.25.a, 6.25.b and 6.24.c similar with the respective graphics presented in Fig. 6.22.a, 6.22.b and 6.22.c for 30%, 60% and 100% error rates; Fig.6.25.d similar to Fig. 6.22.d with regard to all considered error rates).



a. V_{min}= 5%

b. V_{min}= 15%

c. V_{min}= 30%

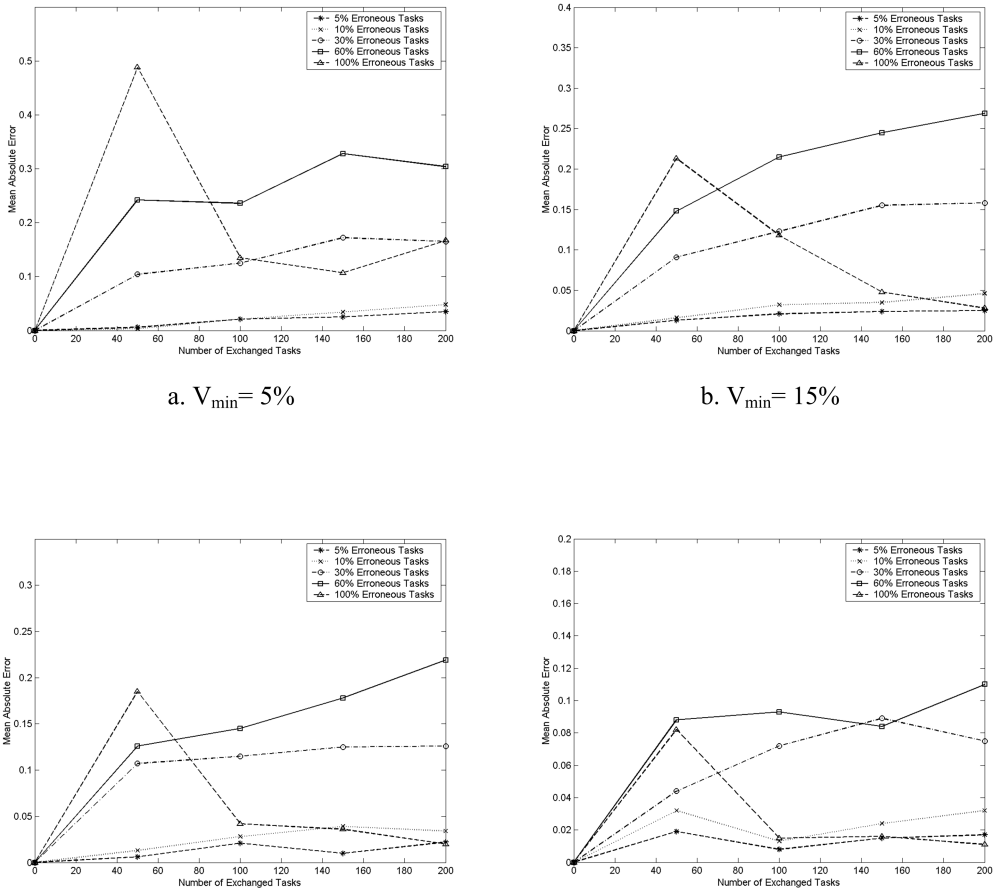
d. V_{min}= 60%

Figure 6.23: Mean Absolute Errors for the Statistical Verification Strategy; clearance number 0.

6.3.2 Partial Results

- The simulation results showed that although errors were found when applying the verification strategy, a gap between the observed behavior and the real behavior a trustee exhibited still exists. Applying the statistical model helps a lot on having a better view on the errors that could have skipped the verification process. The re-evaluated behavior for a provider is definitively near to the real behavior that the provider exhibited.

- The higher the clearance number, the more similar the performance shown by the statistical model with the performance of the verification model only will be.



a. V_{min}= 5%

b. V_{min}= 15%

c. V_{min}= 30%

d. V_{min}= 60%

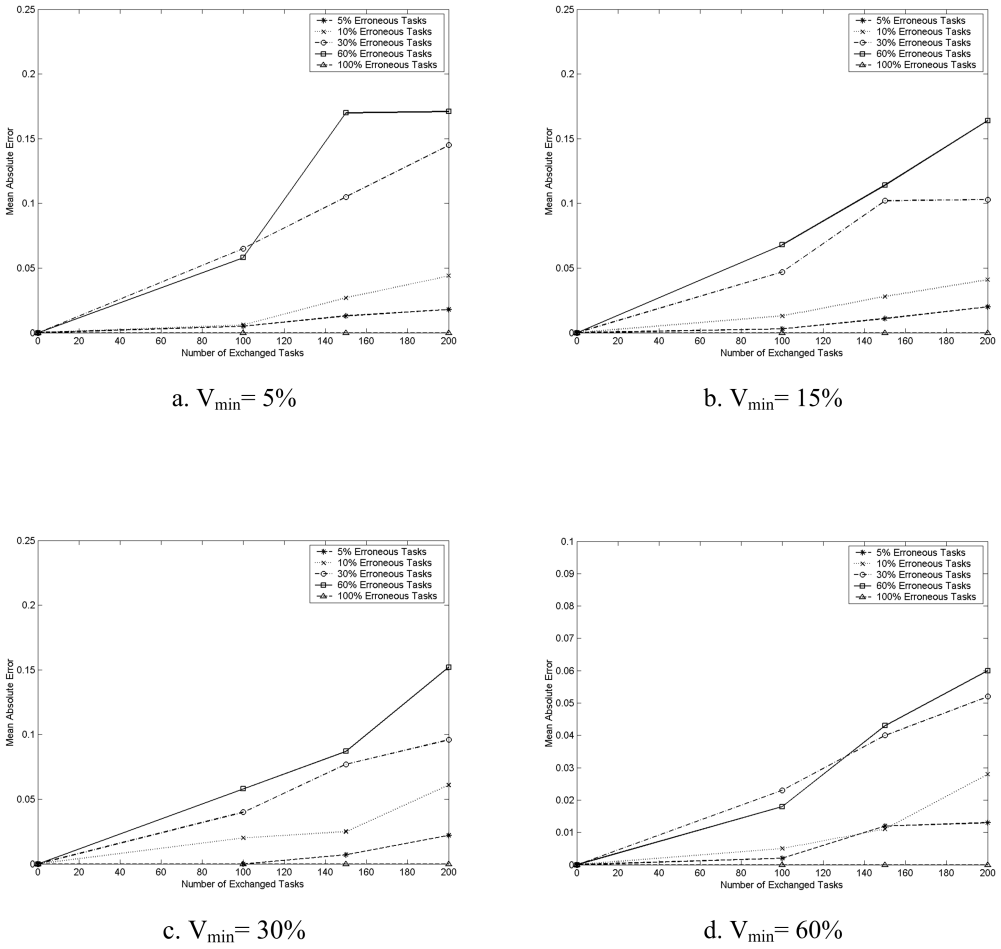
Figure 6.24: Mean Absolute Errors for the Statistical Verification Strategy; clearance number 25.

6.4 Keeping the Behavior of Collaboration Parties "In Control"

Additional simulations were conducted in order to observe the behavior of the trustees between experiments. As previously mentioned at the beginning of the section 5.6, the aim is to:

- verify the deviations on the trustworthiness of a participant between the simulations,
- establish till when the collaboration with a participant, despite its anomalous/malicious behavior may continue.

The *accuracy* of the responses coming from a provider, together with its *availability*, *accessibility* and *speed of processing* were considered. Behavior trust was calculated as the product



a. V_{min}= 5%

b. V_{min}= 15%

c. V_{min}= 30%

d. V_{min}= 60%

Figure 6.25: Mean Absolute Errors for the Statistical Verification Strategy; clearance number 75.

of all these single Behavior Trust elements (5.2). The past behavior of the trustee(s) over a number of simulations was recorded and the current behavior was compared to the recorded values. For this purpose, making use of formulas 5.25 and 5.26, UCL and LCL were calculated.
In this type of graph are placed the results of the behavior observations during the current collaboration. According to the positioning of the observed trust values (outside LCL, between LCL and UCL, or outside UCL) a trustor could derive conclusions on the behavior of its trustees. Thus:

- if the observed behavior lies between UCL and LCL, a participant can still be considered for future interactions but for the accomplishment of moderated expectations, since it did not show an error-free behavior, and

- if the observed behavior lies outside the UCL then the participant is banned and no more considered for future interactions. In this case the collaboration with that target trustee is interrupted and the remaining tasks can be assigned to other (trusted) partners present in the environment.

During the simulations, the clearance numbers used were 0, 25 and 75. The trustor established minimal verification frequencies with values 10%, 30% and 60% and a "warm start" initialization strategy with initial trust value 0.9. The trustee is supposed to initially start the collaboration with an error-free behavior and later on doubles the error rate (after every two collaborations; starting from 0%, 5%, 10%, 20%, 40%, 80% and 100% error rates). In each simulation, a total number of 200 tasks were exchanged between the trustor and the trustee.
In order to fully observe the development of the behavior of a trustee through a sequence of simulations, the stopping rule is configured higher than the number of tasks trustor and trustee exchange. Thus, the collaboration will not be interrupted even in the case when the tasks coming back to the trustor are 100% erroneous.

- **Clearance number 0,** $V_{min} = 10\%$**:** The results are shown in Figure 6.26 (a, b, c, d, e, f, g and h).

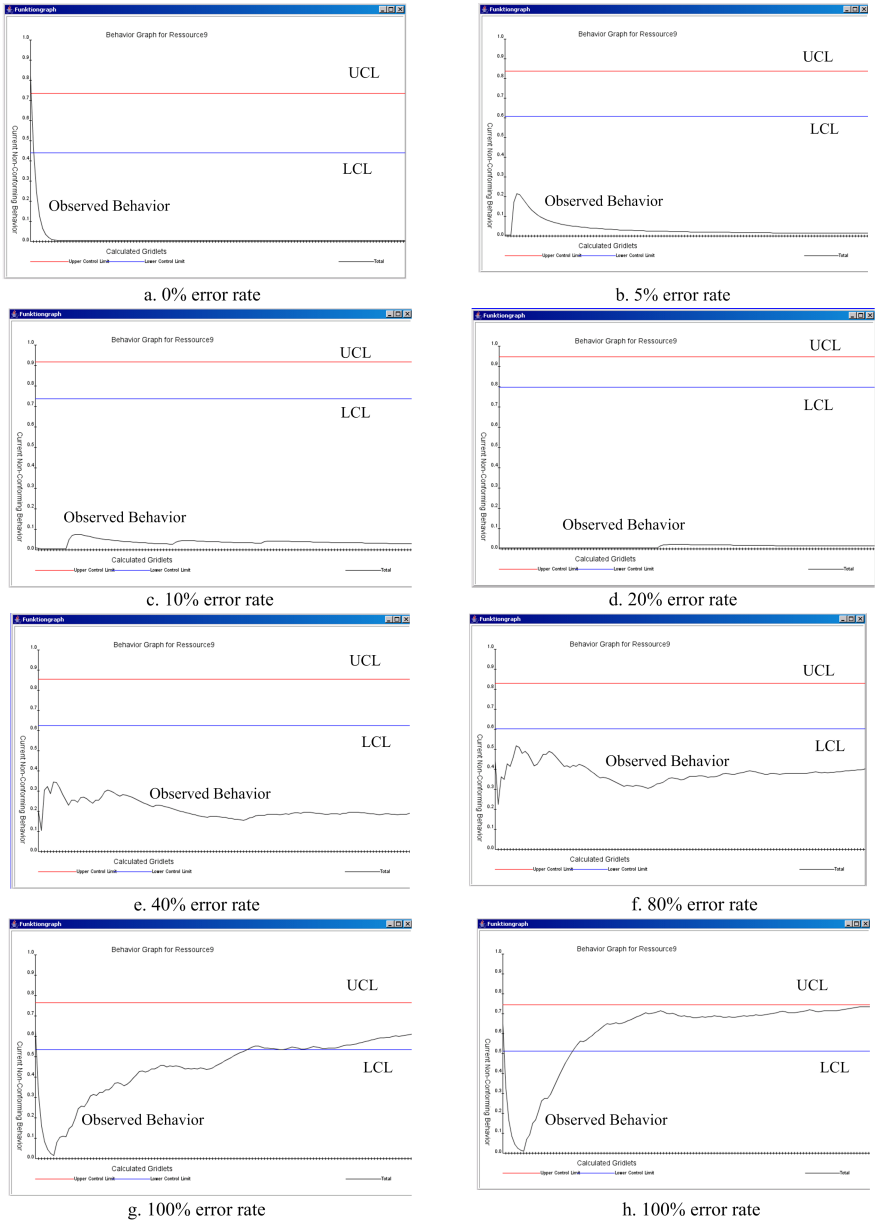
  At the beginning, from the first simulation until the error injection rate reaches 20% (Fig. 6.26 a, b, c and d), it seems that the trustee is behaving perfectly. While the error rate increases, the more the behavior graph approaches the upper control line (UCL) (Fig. 6.26 e and f). However, due to the low verification frequency, this happens very slowly. Only after the error rate reaches 100% (Fig. 6.26 g, h), the trustor, thanks also to the experience gathered in the previous collaborations, once the observed behavior exceeds the UCL, classifies the trustee as *no more suitable* for future collaborations and *blacklists* it.

- **Clearance number 0,** $V_{min} = 30\%$**:** during these simulations, one could observe that thanks to the higher verification frequency, the trustor is able to discover more erroneous tasks coming from the trustee. However, the behavior graph approached the upper control line (UCL) not particularly faster if compared to the scenario presented in Fig. 6.26. The trustor blacklists the trustee only after the error rate reaches 100%.

- **Clearance number 0,** $V_{min} = 60\%$**:** While the minimal verification frequency increases, the trustors discovers earlier the erroneous behavior of the trustee and the number of errors is more elevate. In Figure 6.27 (a, b, c, d and e) can be seen that an erroneous trustee is blacklisted at the frontier of 80% erroneous responses.

Similar simulations were run for a clearance number different from 0 (clearance number equal 25 and 75). The observed results are summarized in the following.

- Clearance number 25:
  - $V_{min} = 10\%$ - the collaboration was interrupted once the trustee started to send back only erroneous responses to the trustor (100% error rate);
  - $V_{min} = 30\%$ - collaboration interrupted once the percentage of the erroneous responses sent back to the trustor reached 80%;



a. 0% error rate

b. 5% error rate

c. 10% error rate

d. 20% error rate

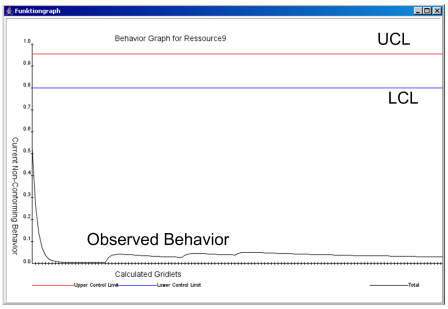
e. 40% error rate

f. 80% error rate

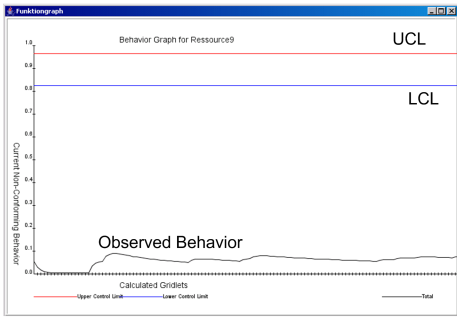
g. 100% error rate

h. 100% error rate

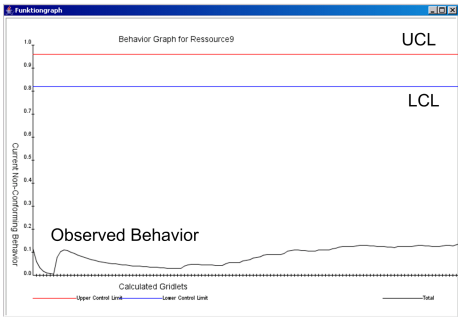
Figure 6.26: Clearance number equal 0; $V_{min} = 10\%$.



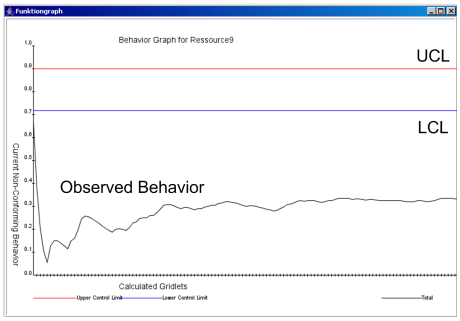
a. 5% error rate



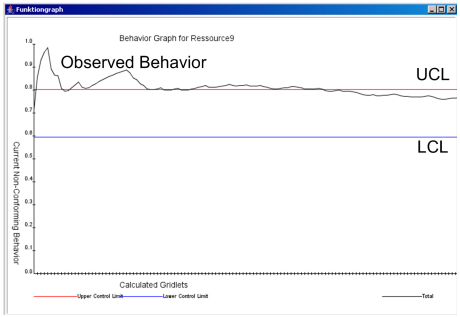
b. 10% error rate



c. 20% error rate



d. 40% error rate



e. 80% error rate

Figure 6.27: Clearance number equal 0; $V_{min} = 60\%$.

  - $V_{min} = 60\%$ - once again, the collaboration was interrupted once the percentage of the erroneous responses sent back to the trustor reached 80%.

- Clearance number 75 - from this moment, independently from the minimal verification frequency used ($V_{min} = 10\%$, $V_{min} = 30\%$ and $V_{min} = 60\%$), the collaboration was always interrupted once the percentage of the erroneous responses sent back to the trustor reached 80%.

The above simulations are run considering a stopping rule greater than the number of tasks exchanged between the trustor and a specific trustee. Once this stopping rule is exceeded, the collaboration with this trustee is interrupted (either the UCL is exceeded or not) and the rest of the tasks are distributed to other "until then" trusted partners. An example of an interrupted collaboration is shown in Figure 6.28.

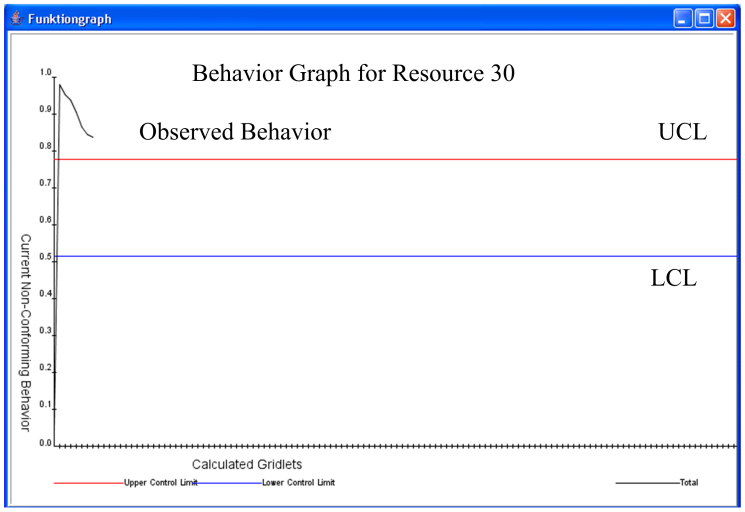


Figure 6.28: Interrupting the Collaboration Once UCL and Stopping Rule are Exceeded.

Partial Results. The simulations showed that:

- as the clearance number and the verification frequency increases, the capability of a trustor to detect any non-conformity (i.e. erroneous responses) in the behavior of any specific trustee also increases.

- for a verification frequency under 100%, the common rate of the injected errors by a trustee reached approximately in every simulation the value of 80%. After that they were blacklisted and no more considered by this trustor even for more moderated trust requirements.

6.5 Handling Inaccurate Recommendations

A further feature of the trust model presented in this thess is the use of recommendations obtained from third parties while choosing the most trusted partners to collaborate with. The trustor has the possibility to establish whose recommendations it is going to accept and what the importance (weight) of these recommendations in relation to the personal experience will be.
The aim of the experiments in this section is to evaluate:

- how the trust of the recommender(s) develops between different experiments, in the presence or absence of erroneous behavior of the trustees they recommend and
- what the effect of different presences of malicious recommendations in the decision making process will be.

Recommenders' Trust Development. According to formula 5.4, the trust each trustor assigns to a recommender is calculated as the average behavior trust, manifested by the trustees recommended by the recommender in question and used by the trustor during the very last collaboration.
During the experiments, three environmental conditions were configured:

1. normal behavior of the trustees: no erroneous activity was injected at all (accuracy 1.0) and the trustees were always available. The only behavior trust elements of importance were accessibility (many participants compete with each other) and speed of processing.
2. trustees inject errors in the responses they send back to the trustor; the fault tolerance of the trustor exceeds multiple times the number of errors contained in the received responses.
3. trustees inject errors in the responses they send back to the trustor; the fault tolerance of the trustor is very small; after four non-correct responses, another trustee is considered for processing the remaining tasks.

The experimental results are represented in Fig. 6.29. Since the trust a trustor develops regarding its recommenders is tightly coupled to the behavior trust of the trustees recommended and used by the trustor, this value stays never to the maximum[2] (value 1.0). It varies according to the behavior of the recommended trustees during the current collaboration.
It is obvious that once the trustees inject errors in their responses, the trust value of recommender diminishes. The differences observed in the trust value of the recommender between the two scenarios, with high and low fault tolerance, are explained by the fact that in the case with a higher fault tolerance the collaboration with an erroneous trustee continues for a longer time than when a lower fault tolerance is specified. The longer a collaboration between the trustor and a trustee lasts, the more the observed behavior is going to be near to the real behavior shown.

Handling Malicious Recommenders. The second group of experiments serves to examine the efficiency of the model on handling with malicious recommenders.

[2]Refer to subsection 6.2.2.

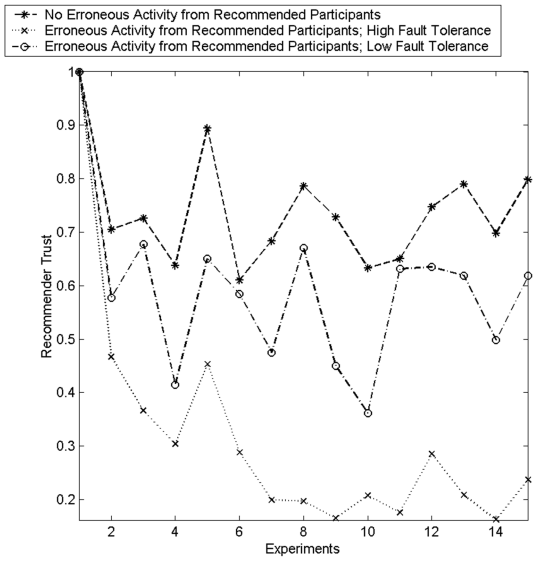


Figure 6.29: Recommenders' trust development.

All recommenders which intentionally offer subjective (very high or very low) trust values as recommendations, even in the case when they personally do not have had a direct collaboration with the trustee in question, are defined as *malicious*.

During the experiments, 20 participants in total were considered, divided as follows:

- 1 trustor,
- 1 trustee,
- 18 others were configured all as recommenders.

The recommenders offer either their correct experience accumulated with the trustee in question or a subjective value (1.0 was considered during the experiments). The trustee is configured to send back 60% erroneous responses. The percentage of malicious recommenders varies from 0% to 100%. The weights, the trustor assigns to the recommenders experience are 0.2, 0.5, 0.8, 1.0 (respective personal experience weighted with 0.8, 0.5, 0.2, 0.0). The evaluation is done through computing the mean absolute error (MAE) according to the formulas 6.2 and 6.3:

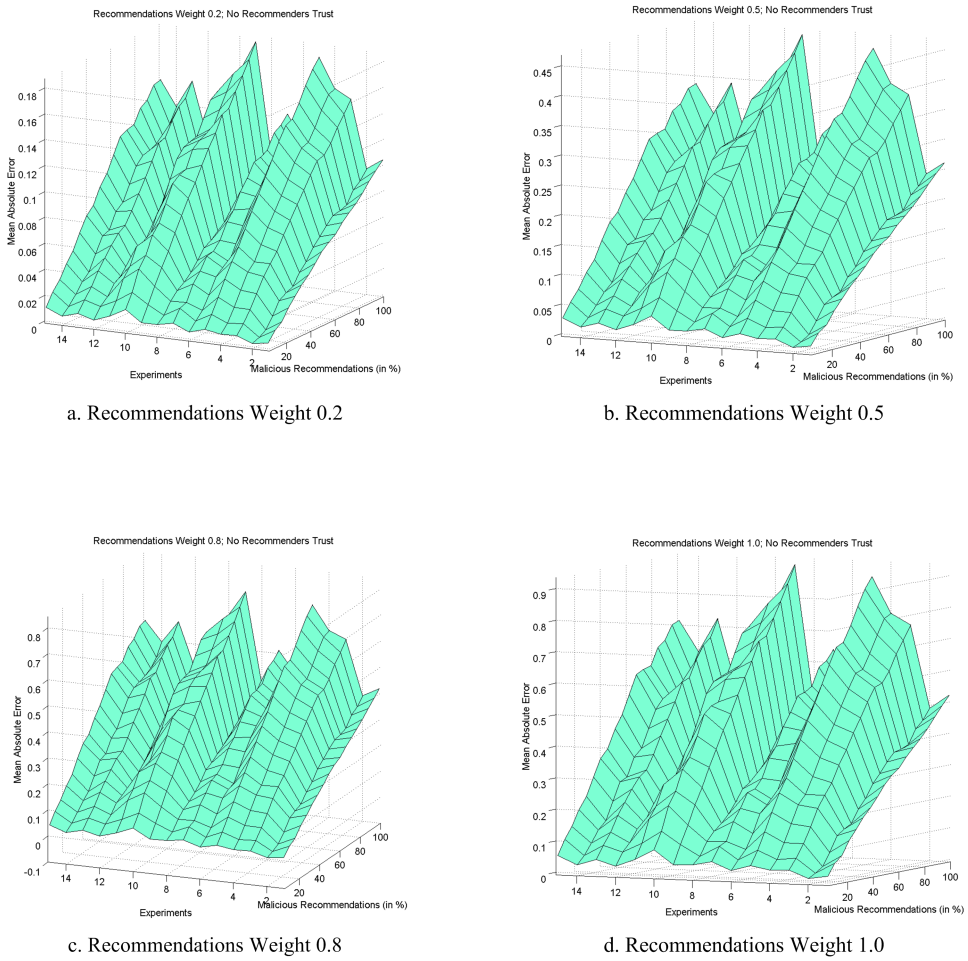
$$MAE = Weight * \frac{\sum_{i=1}^{n} |R_{subjective}(i) - R_{real}(i)|}{n} \tag{6.2}$$

$$MAE = T_R(i) * Weight * \frac{\sum_{i=1}^{n} |R_{subjective}(i) - R_{real}(i)|}{n} \tag{6.3}$$

where n is the total number of malicious recommenders.

The results are presented in Fig. 6.30 (a, b, c and d) and Fig. 6.31 (a, b, c and d). In Fig. 6.30, the mean error is calculated according to the formula 6.2. The trust values of the

recommenders are not involved in the calculations. The mean absolute error increases, first, as the percentage of the malicious recommenders grows and second, as more importance is assigned to recommendations.



a. Recommendations Weight 0.2

b. Recommendations Weight 0.5

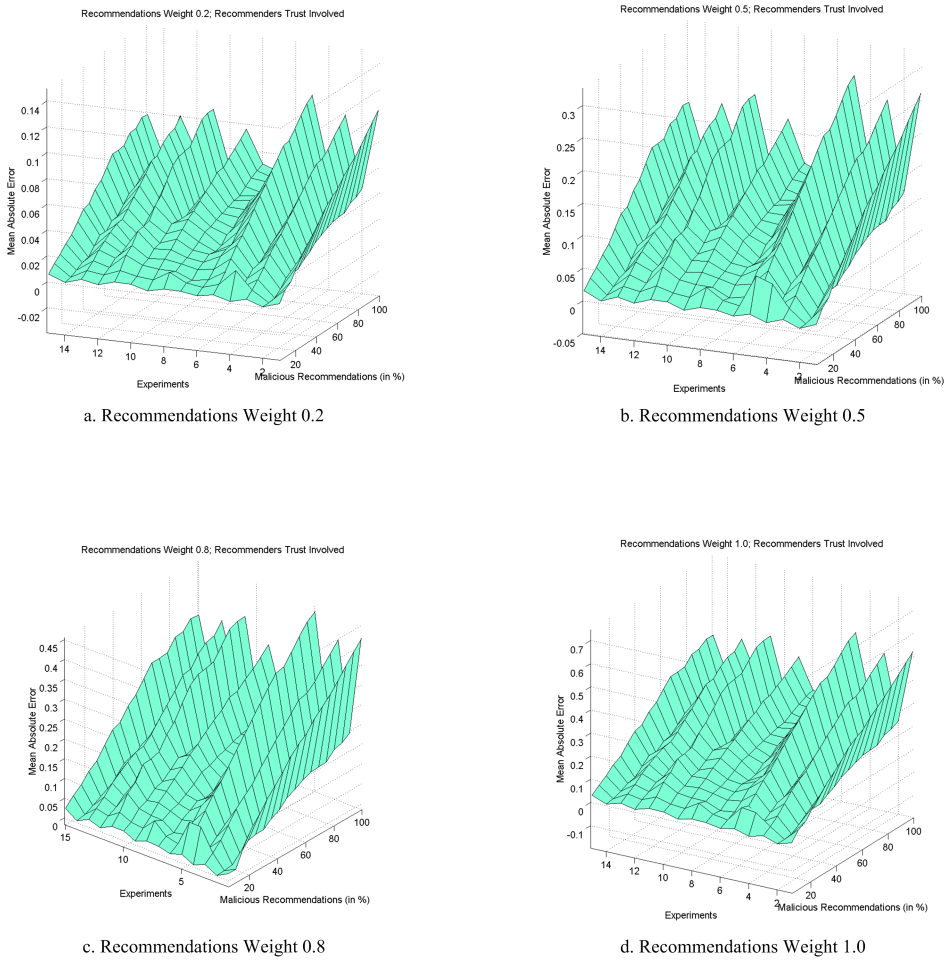
c. Recommendations Weight 0.8

d. Recommendations Weight 1.0

Figure 6.30: Mean Absolute Error caused by malicious recommenders (without recommenders' weight).

In Fig. 6.31, the mean error is calculated according to formula 6.3. The trustor does involve the trust values of the recommenders. The decrease of the mean absolute error is visible in all four graphs especially in Fig. 6.31 d for 100% malicious recommenders and weight 1.0.

In the model, there exists a strong correlation between the behavior the trustees exhibit during the collaboration with a trustor and the trust values of the participants that recommended them. A weak or good performance of a trustee will also directly affect the trust value of its recommender(s), diminishing or increasing it.



a. Recommendations Weight 0.2

b. Recommendations Weight 0.5

c. Recommendations Weight 0.8

d. Recommendations Weight 1.0

Figure 6.31: Mean Absolute Error caused by malicious recommenders (with recommenders' weight).

Discussion. Compared to other proposals for handling inaccurate reputation sources like Travos [251], Beta Reputation System [163] and BLADE [225], the presented approach has the following advantages:

- Simplicity and speed on calculating recommenders' trust. The trustor evaluates each recommender according to the final trust value of the trustees it recommended. This value is updated at the end of each collaboration process and stored for future use. On the contrary, the three other approaches, before establishing trust on a trustee, initiate a recommender estimation process gathering also information from other (known) participants in the environment. This approach can turn up to be impractical as for the additional time it consumes, the possible inaccuracy of the gathered results (although

coming from the so called "known" partners) and also for the "artificial" trust assigned to these "known" partners.

- Continuous observation of the behavior of the recommended participants. Behavior of the trustees is not considered as constant during the collaboration as in Travos [251]. Once the trust is established, it needs to be monitored. The better the trustor knows the behavior of the trustee it collaborates with, the smaller the influence of the "inaccurate" recommenders.

- Consideration of the experience of all participants in the environment for as long as they fulfill the trustor's requirements. Trust is build upon personal experiences. It makes no sense to discriminate opinions that differ from the average, as in Beta Reputation System [163], for as long as the participants that offer them have proven themselves as valuable recommenders.

- Possibility, through the first trust approach, to a trustor to express its personal attitude toward a trustee, even when it completely lacks of previous collaborations with the trustee.

- Realistic results. Although the experimented scenarios were run in a simulated environment, the results leave little space for surprises, since they were obtained from a very "real life"-like trust establishment and management process.

6.6 Measuring the Effects of Trust

In the following, experiments for evaluating the effects of trust for the Grid participants and for the environment itself are presented. The parameters measured are:

- Processing cost;

- Processing time;

- Network load.

For measuring processing time and costs, two types of experiments were run. In the first type, a comparison between systems with and without trust was done. In the second type, different verification strategies were compared.

6.6.1 Evaluation of Processing Costs and Time

During the experiments, 30 participants were created in total. One was assigned the role of the trustor and the others were trustees. A fault tolerance of 4 erroneous tasks per trustee was the condition for the trustor to interrupt the collaboration (fault tolerance) and check for some other, more trusted, trustees. In other words, if the trustor finds 116 erroneous tasks, after establishing a collaboration with all the present trustees, than no more tasks were sent out. The environment is considered as *unusable*.
The trustor creates 250 tasks, where:

- by a 10% errors frequency there are approximately 25 erroneous tasks are injected,

- by a 60% errors frequency there are approximately 150 erroneous tasks are injected,
- by a 100% errors frequency there are 250 erroneous tasks are injected,

The number of the discovered erroneous tasks and indirectly the processing costs and time depend on the verification strategy applied.
The results of the measurements are represented in Fig. 6.32 and Fig. 6.33:

Processing Time (Trust vs. No Trust).

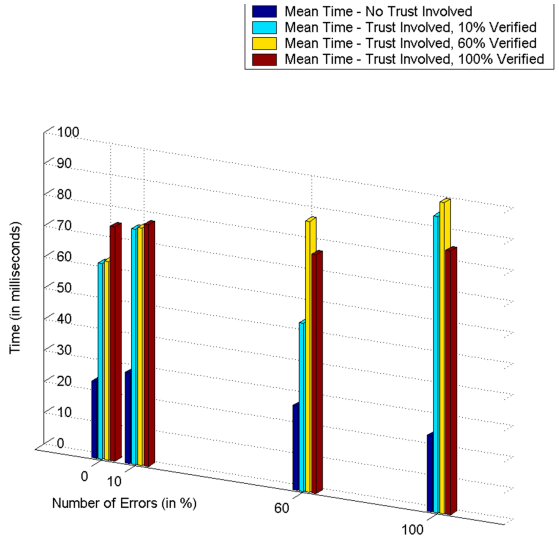


Figure 6.32: Measured processing time (trust vs. no trust).

Processing Cost (Trust vs. No Trust).
It can be seen that in the first sight, the experiments run when the trust is not involved are faster and cheaper than the experiments run when trust is involved in the system. Furthermore, the higher the verification frequency (including also the clearance number), the longer the experiments last and the more expensive they are. However, the results were obtained through measuring directly the costs and time of processing the tasks a trustor sent to its trustees. While during the scenario where trust was not involved, although a trustee (the trustees) injected errors in their responses, the collaboration continued normally until all tasks were processed, introducing trust considerations and behavior monitoring brought the interruption of the collaboration when the trustor's trust consideration were not fulfilled by a certain trustee. The rest of the tasks was (re)distributed among other participants that showed themselves as trusted ones (trustees). Thus, the real advantage of including trust in the system relies in the total number of erroneous/malicious participants discovered. As the number of erroneous/malicious participants in the environment increases (especially when an unusable status for Grid environment is achieved), the greater is the advantage brought by this trust model. The confirmation comes from the statistical reevaluation of the data recorded during the simulation of the above scenarios (where the no trust scenario resulted faster and cheaper than with trust scenario). Statistically speaking:

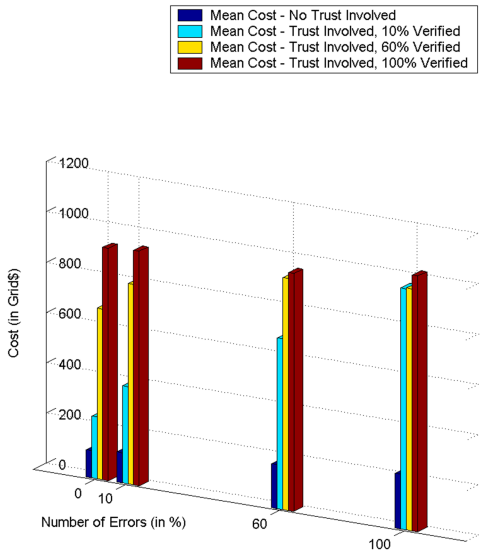


Figure 6.33: Measured processing cost (trust vs. no trust).

- for 10% verification frequency
  - 10% injected erroneous tasks, approximately 2.5 are discovered
    - for clearance number 10, approx. 3.5 erroneous tasks are discovered,
    - for clearance number 50, approx. 7.5 erroneous tasks are discovered,
    - for clearance number 75, approx. 10 erroneous tasks are discovered,
  - 60% injected erroneous tasks, approximately 15 are discovered
    - for clearance number 10, approx. 21 erroneous tasks are discovered,
    - for clearance number 50, approx. 45 erroneous tasks are discovered,
    - for clearance number 75, approx. 60 erroneous tasks are discovered,
  - 100% injected erroneous tasks, approximately 25 are discovered
    - for clearance number 10, approx. 35 erroneous tasks are discovered,
    - for clearance number 50, approx. 75 erroneous tasks are discovered,
    - for clearance number 75, approx. 100 erroneous tasks are discovered,
- for 60% verification frequency
  - 10% injected erroneous tasks, approximately 12.5 are discovered
    - for clearance number 10, approx. 13.5 erroneous tasks are discovered,
    - for clearance number 50, approx. 17.5 erroneous tasks are discovered,
    - for clearance number 75, approx. 20 erroneous tasks are discovered,
  - 60% injected erroneous tasks, approximately 75 are discovered
    - for clearance number 10, approx. 81 erroneous tasks are discovered,

    - for clearance number 50, approx. 105 erroneous tasks are discovered,
    - for clearance number 75, approx. 120 erroneous tasks are discovered,
  - 100% injected erroneous tasks, approximately 125 are discovered
    - for clearance number 10, approx. 135 erroneous tasks are discovered,
    - for clearance number 50, approx. 175 erroneous tasks are discovered,
    - for clearance number 75, approx. 200 erroneous tasks are discovered,

By a 10% verification frequency it comes only rarely to an interruption of all collaborations, however, some erroneous tasks are discovered. By a 60% verification frequency, the interruption happens for the first time when the trustees inject 60% erroneous tasks and the clearance number 75. When the trustees inject 100% erroneous tasks, than the interruption happens every time. In other words, when an unusable status is detected by the trustor, the interruption of the collaboration with all erroneous trustees could bring 53.4% savings in time and costs compared to the case when no verification took place at all.
Configuring a lower fault tolerance will bring even better results.

6.6.2 Network Load

The network load observed during the experiments is proportional to the number of active participants present in the environment, their replicated tasks, erroneous tasks found and resent for processing and to their replicas. This relationship can be described through the following formula 6.4:

$$NetLoad = N_{particip}(i) * N_{rep_tasks}(i) + \sum (N_{particip}(j) * N_{resend_tasks}(j) + N_{particip}(j) * N_{rep_tasks}(j)) \tag{6.4}$$

where $N_{particip}(i)$ is the number of those participants whose tasks are remotely being processed, $N_{rep_tasks}(i)$ is the total number of replicated tasks, $N_{particip}(j)$ is the number of participants that have already found errors on the tasks they receive back, $N_{resend_tasks}(j)$ is the number of tasks found erroneous and resent to be processed and $N_{rep_tasks}(j)$ is the number of replicas for the resent tasks.

6.7 Summary

The aim of this chapter was to assess the performance of the trust model presented in this thesis.
Initially, the basic functionalities of the simulation toolkit [http://www.gridbus.org/gridsim] were extended. The changes were undertaken, in order to integrate the different elements of the model, as introduced in the fifth chapter and to comply as much as possible with a real Grid environment (e.g. the possibility given to a participant to play the role of a consumer and a provider at the same time). The simulations were managed through a graphical interface. Instructions to the underlying configurable experiment class(es) are given and at the same time information on the very last simulation was accessed. For each of the trustors, the behavior shown by its trustees between different simulations was separately saved. Another implemented class, the GPaint class, made use of the saved results for delivering graphical representations of the behavior shown for each of the trustees during the very last simulation

and among different simulations (long time behavior observation).

The second part of the chapter concentrated on the presentation of the experimental work done for:

- evaluating the performance of the trust model during establishment of trusted collaborations between parties, monitoring of the collaboration and discovery of short-term or long-term deviations on the behavior of the collaboration partners,
- evaluating the effects of trust on the performance of the Grid systems and
- helping in some way the Grid participants to tune their trust requirements to their needs and capabilities.

During the simulations, a set of resources and users was modeled:

- each user created/owned a resource.
- each user sent his/her tasks to every resource in the environment (considered as suitable in respect of its current personal trust requirements), except for its own resource.
- the behavior trust elements considered were: the accuracy of the responses coming from the different resources, their availability, accessibility, and speed of processing.

To observe the trust establishment, different scenarios, with different trust requirements, were considered. The opened or closed attitude of the trustor, expressed through its moderated or high trust requirements respectively, was also reflected by the experimental results.

During the experiments, the way how trust develops in the simulated environment among Grid participants was tracked down. The two main scenarios involve the collaboration between trustors and trustees in the absence and presence of malicious behavior from the trustees. While in the presence of malicious behavior it was somehow expected that the trust value for the behavior trust was subject of oscillations (due to the discovered deviations), during the experiments, where the malicious behavior lacked completely, it was observed that:

- even with the very best intent of a participant to offer its best, it is quite impossible that the trust value remains (if "warm start" with initial trust value 1.0) at, or reaches (if "cold start" with initial trust value 0.0) its maximal value 1.0, and
- a higher constant trust value for a target participant does not always reflect the observed behavior regarding that participant. The trustor could have simply not established a collaboration with that partner at all or in a long time.

Simulations were also conducted in order to measure the performance of the trust model on detecting malicious behavior of the participants (trustees) during a single collaboration[3] or over a number of them.
Through measuring the differences between the real behavior shown by a trustee during the

[3]A single collaboration a trustor establishes with a trustee is considered the entire process from the moment the first task is sent from the trustor to the trustee till the trustor receives the last processed task from the trustee.

collaboration and the behavior, a trustor was able to monitor, according to the verification strategies and sub-strategies, or to statistically evaluate, the mean absolute error for the entire monitoring process was determined. Different grades of malicious behavior of the partners were simulated and different verification strategies and sub-strategies were applied. The results showed that the model, based only on the verification frequency proposed in 5.8 was able to detect some deviations in the behavior of a trustee. This performance was further improved through the use of the statistical methods of quality assurance. Experimental results showed that the higher the number of sequential tasks verified at the beginning of the collaboration (clearance number), the more similar the performance of the monitoring process to the statistical re-evaluation process was.
Additional simulations were conducted in order to observe the behavior of the trustees between experiments. This time, not only the accuracy of the responses coming from a provider but also its availability, accessibility and speed of processing were considered. The aim was to observe the conformity of the behavior a trustee showed during a collaboration, with the expectations a trustor built based on the results gathered from previous simulations.
The CL, UCL and LCL were measured, according to 5.24, 5.25, and 5.26, making use of the information gathered during the past direct collaborations. For different grades of malicious behavior injected and different verification strategies and sub-strategies applied, different performances of the model were observed.

A further feature of the model that was tested is its capability of handling malicious recommenders.
The trustor has the possibility to establish whose recommendations it is going to accept and what the importance (weight) of these recommendations in relation to the personal experience will be. Initially, the way how the trust of the recommender(s) develops between different experiments in the presence or absence of erroneous behavior of the trustees they recommend was observed. The results confirmed the expected decreases in the trust values of the recommenders as the malicious behavior of the partners they recommended increased. As a second step, the effects of different presences of malicious recommendations in the decision making process were determined. As malicious were defined all those recommenders which intentionally offer subjective (very high or very low) trust values as recommendations, even in the case when they personally do not have had a direct collaboration with the trustee in question. Once again, the parameter under observation was the mean absolute error calculated as the difference between the resulting values for the gathered recommendations, including the subjective ones(where no affinity to the real experience exits), with the values for the gathered recommendations, when the recommenders reported their real experience (or even the lack of experience as well) with the target trustee. The results showed that the mean absolute error increases, first, while the percentage of the malicious recommenders grows and second, while more importance is assigned to recommendations. The higher value is reached for 100% malicious recommenders and weight 1.0.

The last part of the experiments deals with the investigation of the effects that the involvement of trust brings in the environment in general and for the Grid participants. The parameters measured are the processing cost, processing time and the network load.
For measuring processing time and costs, two types of experiments were run. In the first type, a comparison between systems with and without trust was done. In the second type, differ-

ent verification strategies applied were compared. The results showed that, the experiments run when the trust is not involved were visibly faster and cheaper than the experiments run when the trust is involved in the system. Furthermore, the higher the verification frequency (including also the clearance number), the longer the experiments lasted and the more expensive they were. However, the real advantage of including trust in the system lies in the total number of erroneous tasks discovered, especially in the case when an unusable status is achieved in the environment. Thus, unacceptable deviations are contained in the behavior of all trustees a trustor, according to its trust requirements, considers as suitable for establishing a collaboration with. All this, translated in processing costs and time terms means that considerable savings could be achieved when the trust is involved in the system. The lower the fault tolerance will be (expressed by the trustor at the personal trust requirements) the greater the advantages will be.

Regarding the network load, the experiments showed that it is proportional to the number of tasks that each of the participants, according to the verification frequency applied, sends to be executed twice (replicas) and to the number of tasks found as erroneous and thus re-sent for execution.

Chapter 7

Conclusions

"A conclusion is the place where
you got tired thinking"
Dr. Martin Henry Fischer

7.1 Summary

This thesis presented an analysis of trust and a trust establishment and management model for Grid environments.

Trust is a complex subject relating to belief in the honesty, truthfulness, competence, reliability, etc., of the considered participants. In a trust management system, there are participants who assign some degree of trust to other participants based on a combination of identity considerations, behavior observations, recommendations from other participants and references to other trust sources. The degree of trust that the participants assign to each other governs the decisions that they make when collaborating with each other. Definition of a trust model is the basis for interoperability between Grid participants in a heterogeneous, dynamic, uncertain and vulnerable environment.

The main contributions of the thesis are as follows:

- **Consideration of Social Aspects of Trust in Grid Environments.** A wide variety of needs and requirements for trust by the users of applications running on Grids, regarding possible collaboration partners, were considered. Trust is established and managed based on experience (personal or that of the others), beliefs and prejudice. Each of the participants is able to manage its own trust values regarding the others in the environment and also offer this experience when it is requested.
- **Trust Threats Analysis.** Threats to trust in Grids were analyzed for getting a better understanding of possible fraudulent behavior of the participants in the environment.
- **A Flexible and Generic Trust Establishment and Management Model.** The offered approach considered both identity and behavior trust of the interaction partners

together with different sources to calculate the overall or partial trust values for an interaction partner.

- **User/Application and Trust.** Trust establishment and management were adapted to user and application requirements in a service-oriented Grid environment. The trust system was configured to the domain specific trust requirements by the use of several separate trust profiles covering the entire lifecycle of trust establishment and management.

- **Relationship between Trust and Control.** Each of the collaboration parties continuously controlled the behavior of the other party based not only on the user/application trust requirements but also on the observed behavior of the other party until that very moment.

- **Detection and Prevention of Anomalous Behavior.** Statistical methods of quality assurance were used for monitoring the behavior of other participants during long and short term collaborations as a tool for detecting possible deceitful behavior and trust betrayal.

- **Enhancing the Security of the Communication.** Securing the communication between Grid participants was identified as an important task. The proposed approach made use of a double encryption scheme in which the transmitted information was initially encrypted using incomparable public session keys (a technique where a participant generates itself several public keys corresponding to a single private key; the number of public keys equal the number of trusted partners a participant identifies). In a second stage, this already encrypted information was encrypted again using keys generated through a technique based on certificateless public key cryptography.

- **Analysis of System Performance while Involving Trust.** The effects that trust can have on important features like mean processing time, and processing costs were analyzed through different simulations.

- **Effects of Control Levels on Trust.** Many situations of normal and anomalous behavior were simulated and analyzed. Different control levels (from strong control to moderated control) exerted from participants and different types of interactions between them were configured. This analysis is supposed to serve to users of the model as a reference while expressing their trust preferences.

A comparative view of the characteristics of the offered approach to the related work in the field of trust for Grid environments is presented in Table 7.1.
The considered features are summarized in the following:

- **Trust sources** - Trust is still a *knowledge gaining process* where individual participants learn to trust others in the environment. In order to have a complete/better view on the trust of a collaboration partner, *many dimensions*, like personal experience, personal predisposition, third parties experience, etc., need to be considered. Different weights can be assigned to each of the sources conform participant's attitude in using/accepting them.

- **Centralization level** - Trust is inherently a *personal opinion*. Each of the participants, according to the current (application-related) trust requirements, establishes trust only with a certain number of partners from the entirety present in the environment. Trust on a trustee *increases* or *decreases* with time. It ranges from complete distrust to complete trust based only on:
  - the *intentions* and the *behavior* showed from the collaboration partners, and
  - the *influence* of other factors (e.g. concurrence in the environment).
- **Generalization (Granularity)** - different activities in which a participant is involved have to be *separated*. This is the only possibility to increase the quality of the offered services in the environment and assist participants to express which aspect of others' behavior they are interested in and at which level.
- **Trust level** - considering both *identity* and *behavior* trust, makes Grid participants more transparent to each other.
- **Mutual verification** - Trust could always be a *measurable* property in Grid environments. The trust level is a measure of participants' honesty, competence, security and dependability of this specific participant. In the environment, some participants may be trusted more than others with respect to their behavior. To have an accurate view on the level of trust regarding a specific participant, this has to be a *continuous value*.
- **Security considerations** - Trust is always *accompanied by risk*. During the collaboration, the presence of some uncertainty on the outcome of the collaboration process with a specific participant still exists, but at the same time some degree of trust in it is also needed. Some "malicious/deviating" behavior could be tolerated by the participants, but when a certain threshold is reached, *trust flips to distrust*.
  Trust alone is not enough. Further security considerations have to be involved in order to increase the confidence and reliability on the collaboration process with a participant and with the entire Grid environment.

7.2 Future Work

The work presented in this thesis is an attempt at modeling trust establishment and management for Grid computing environments, based on social properties of trust. There are several open research areas that need further investigation.

Extension of the Trust Establishment and Management Model. In this work, all Grid participants were organized for simplicity as:

- Grid Users;
- Grid Consumers;
- Grid Providers.

In reality, Grid environments comprise a wider variety of participants. These are:

Trust Models	Trust Sourc.	Centr. Level	General. (Granul.)	Trust Level	Mutual Verif.	Sec. Consid.
Azzedin et al. [79]	Direct, Indirect	Decentr.	Context	Id. Behav.	No	No
NetShield GridSec [161]	Direct	Centr.	Global	Id	Id. at begin.	Yes
G-Pass [190]	Direct	Centr.	NA	Id	No	Yes
GridEigen Trust [74]	Direct Indirect Indirect	Centr.	Global	Behav.	No	No
Lin et al. [184]	Direct Indirect	Decentr.	Global	Behav.	No	Partial
TRAVOS [251]	Direct Indirect Prejud.	Decentr.	Global	Behav.	No	No
Two-Level Trust [252]	Direct	Centr. Decentr.	Global	NA	Id. at begin.	No
Appr. present. in this work	Direct Indirect Sociol. Prejud.	Decentr.	Global Context	Id., Behav.	Id. at begin., Behav. all the time	Yes

Table 7.1: Grid Trust Models Comparison Table.

- Application Developers - construct Grid enabled applications and components. They provide programming models appropriate for Grid environments and a range of services that can be called during application development;
- Tool Developers - develop tools, compilers, libraries and so on that implement the programming models and services used by application developers;
- Grid Developers - implement the basic services required to construct the Grid itself;
- Grid Nodes - either individual systems (computers, storage systems, sensors, etc.), characterized by a relatively small scale and a high degree of homogeneity and integration, or clusters (network of workstations, any collection of computers) characterized by an increased physical scale, reduced integration and a high level of homogeneity. Such systems could be consumer nodes, provider nodes or both consumer and provider nodes;
- Service Providers/Owners - a further classification implying those intermediary participants between Grid nodes and application developers. This category is considered to deal directly with the provider nodes offering Grid-enabled services and applications developed by application developers.

As a result, the following trust relationships are supposed to exist in the environment:

- Grid Users - Consumer Nodes;
- Consumer Nodes - Provider Nodes;
- Provider Nodes - Service Providers/Owners;
- Consumer Nodes - Application Developers;
- Provider Nodes - Application Developers;
- Service Providers - Application Developers;
- Application Developers - Tool Developers;
- Tool Developers - Grid Developers.

These trust relationships could be either bidirectional or unidirectional. A general view on the existing trust relationships is shown in Fig. 7.1.

Adapting the new trust relationships in the trust model will considerably enhance its capabilities and will be a further step towards more trusted and secure Grid environments. Thus, part of the future research should be focused on:

- "shaping" the needs of every possible participant for trust;
- expressing the trust requirements;
- establishing the trust elements of interest for both identity trust and behavior trust.

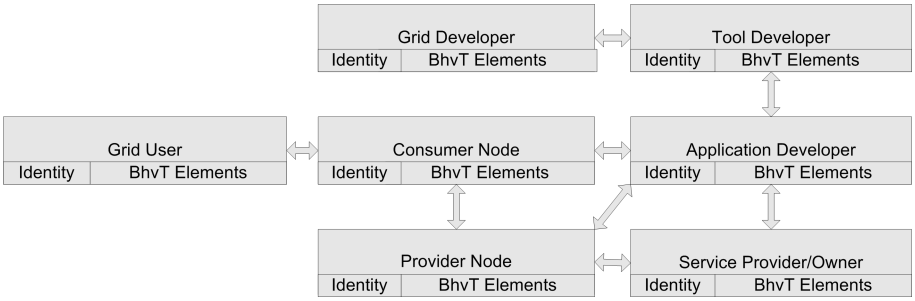


Figure 7.1: Trust Relationships in Grid Environments.

The presented trust model is generic enough to be applied for all possible trust relationships among all categories of participants in Grid environments.

Extension of the Behavior Trust Elements Under Observation and Relationship to Other Technologies. As seen in chapters 3 and 5, the behavior trust elements of interest were limited to:

- Availability;
- Accessibility;
- Accuracy;
- Response Time;
- Latency;
- Throughput;
- Packet Dropping;
- Packet Dropping Duration;
- Packet Size;
- Bandwidth;
- Concurrent Requests;
- Due Payment;
- On-time Payment.

Future research could be oriented towards the identification of further behavior trust elements. Their integration in the presented trust model should be a relatively easy task, since the model itself is very flexible and configurable at every level.

Further research should be devoted to consider the relationship of this trust model to other technologies that help make Grid environments more secure.
Several vulnerabilities and uncertainties in Grid environments were presented and analyzed. Involving the notion of trust will help the participants in the environment to have fewer uncertainties regarding their collaboration partners and at the same time diminishes some of the vulnerabilities that exist. However, the behavior of a Grid participant is not limited only to the fulfillment of its role as a consumer or provider to the expectations of the other party. Every participant should fulfill its role as a "Grid participant" towards everybody else in the environment not only during the collaboration (here, the involvement of the participant in different threats and attacks like DOS and DDOS attacks, attacks on XML parsing systems, etc., are implied).
There are technologies and mechanisms that deal with responses to incidents, intrusion detection, etc., which make the identification of different malicious activities in the environment possible (i.e. Schridde et al. [234]).
Future research should concern the relationship of the presented trust model to such mechanisms, in order to detect malicious participants before they could be chosen as collaboration partners.
In Fig. 7.2, the relationship of the trust to other security mechanisms is shown.

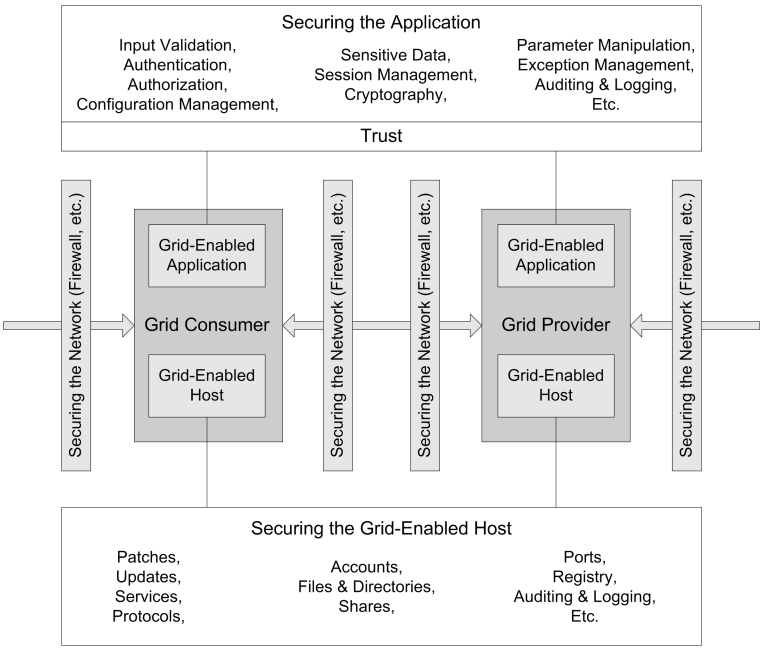


Figure 7.2: Improving Grid Security: A Summary of Threats and Countermeasures.

The entirety of these protective mechanisms (trust management included), will definitively improve the security of Grid environments in general. Only in such a way, Grid technology will be widely accepted outside of the academic domain (i.e. like in the application scenario presented by Dörnemann et al. [131].

Appendix A

Grid Simulators vs. Real Grid Platforms

There are many benefits of using a Grid computing infrastructure:

- harnessing of the collective capacity of the resources involved,
- simplified collaboration between geographically distributed and/or independent organizations (or singular participants),
- sharing of hardware and software resources and of scientific instruments.

There are a lot of initiatives regarding the construction of frameworks and toolkits for enabling Grid environments. Single projects that have been started from big companies or universities are a reality that gains even new understandings as the concept of Grid computing evolves (Jennings et al. [121]).
These approaches offer a collection of protocols and services to ease the implementation of Grid systems.
These initiatives for enabling Grid activities can be categorized into[1]:

- Real-life toolkits and frameworks - Real life toolkits and frameworks run real software on realistic machines that are connected by real networks. Such platforms are DAS2 [51], NAREGI [36], TeraGrid [50] or Grid'5000 [25]. Running experiments in such platforms has the advantage that it is possible to near the implementation in practice and that the obtained experimental results are realistic. However, considering the type of applications running in Grid environments (time-intensive; labour-intensive; uncontrolled and unrepeatable) and the dynamic nature of Grid environments, real life platforms are still challenging for Grid researchers. Changes in the configuration of the environment could negatively influence the running experiments and introduce failures. Furthermore, their scalability is limited and the possibility to build a real-life Grid testbed each time one wants to test different scenarios with different scheduling management strategies is time and effort consuming.
- Emulators - Emulators are another category of the platforms to conduct Grid experiments. Such platforms are MicroGrid [35] and Grid eXplorer [23]. Emulators address

[1] A throughout presentation of experimental platforms for Grid computing can be also found at [222].

the "simulation" of complex behaviors and interactions of distributed Grid participants. The software, in its whole complexity, is run on real machines and this feature limits the scalability and the flexibility needed for the experiments. Emulators are pretty complex, since many of the details in real Grid environment are provided. Due to their complexity, they are also difficult to implement.

- Simulators - Simulators are the third class of experimental platforms for Grid computing. They belong to the high level observation tools for Grid computing. An interesting feature of simulators is the abstraction they provide, focusing on specific behaviors or mechanisms of a Grid system. Furthermore, they are independent of the execution platform because only a model of the system runs in the simulator. Furthermore, simulators solve many problems that come up when using real life platforms or emulators. There is no need to build a real system and there are no limitations to the scenarios being experimented. It is pretty easy to control and to repeat experiments.
  Some of the existing simulators are Bricks [4], SimGrid [48], GridSim [27], ChicSim [5] and OptorSim [38].
  A problem that could come up through working with experimental results obtained through simulations is their validity and the comparability with results obtained from experiments executed on real-life platforms.

However, the aim of the experiments is to observe the establishment and development of trust, the performance of the trust model in general (performance of verification strategies applied, number of deviations not discovered from the verification process) and the effects of trust itself in the system (overhead introduced in the network, costs and time spent when trust involved or not, etc). For these purposes, the following features are required:

- the ability to model an arbitrary number of participants,
- the ability to model a variable performance of the participants, either for single/combination behavior trust elements or for the entirety of them.

These requirements can be currently met only by simulation packages. However, considering the different features and the aims different simulators were conceived to fulfill, the most suitable of them for running the experiments is going to be chosen.
The Bricks simulator focuses on client/server interaction in global high performance computing systems. It allows for a single centralized scheduling strategy, which does not scale well to large Grid systems. SimGrid is designed to simulate task scheduling (centralized or distributed) on Grids. Independent of its functionalities that can be used for running experiments, it can be considered as a low-level toolkit which interfaces to the C programming language and thus bears some difficulties to splice with previous work done for the trust model. The Chicago Simulator (ChicSim) is a simulation framework built on top of Parsec [41] for studying scheduling and replication strategies in Grids. A Grid is modeled as a collection of interconnected Grid sites with network connectivity of each Grid site modeled as a single parameter (describing the bandwidth of the gateway connecting this Grid site to the other Grid sites). OptorSim is a Java-based Grid simulator oriented on the evaluation of the performance of data access optimization algorithms. It offers limited capabilities and as such can not be considered for experimental purposes in this work. Last, GridSim is a

discrete-event Grid simulator based on JavaSim (JSim) [30]. This simulator allows to simulate distributed schedulers, and although it is specifically aimed at simulating market-driven economic resource models it fits well to the requirements presented above. Its computational resource models are highly configurable and it is flexible enough to experiment in different scenarios. A detailed description of GridSim and of changes done to suit the model requirements is presented in Appendix B.

Appendix B

Building a Simulation Infrastructure with GridSim

The GridSim toolkit [27] supports modeling and simulation of a Grid environment through a wide range of heterogeneous participants. It was proposed and designed by Rajkumar Buyya [95] as part of his Ph.D. work at Monash University (Melbourne, Australia), for investigating interactions and interferences between scheduling decisions taken by distributed brokers. It is mainly used to study cost-time optimization algorithms for scheduling task farming applications on heterogeneous Grids considering economy-based resource management and dealing with deadline and budget constraints. The scheduling involves notions like providers (resource owners), consumers (end-users) and brokers discovering and allocating resources to consumers.
GridSim, for its features and flexibility to introduce changes is considered to be the most suitable tool for running the experimental evaluations of our work. In the following, a detailed presentation of GridSim, together with the changes made to its structure for adapting it to the experimental needs, will be presented[1].

B.1 Grid Modeling and Simulation with GridSim

GridSim has the following features:

- It allows modeling of heterogeneous types of resources.
- Resources can be modeled operating under space- or time-shared mode.
- Resource capability can be defined (in the form of MIPS as per SPEC benchmark).
- Resources can be located in any time zone.
- Weekends and holidays can be mapped depending on resource's local time to model non-Grid (local) workload.
- Resources can be booked for advance reservation.
- Applications with different parallel application models can be simulated.

[1] Buyya's work and other related publications on GridSim found at [27] were the major source for this section.

- Application tasks can be heterogeneous and they can be CPU or I/O intensive.
- There is no limit on the number of application jobs that can be submitted to a resource.
- Multiple user entities can submit tasks for execution simultaneously in the same resource, which may be time-shared or space-shared. This feature helps in building schedulers that can use different market-driven economic models for selecting services competitively.
- Network speed between resources can be specified.
- It supports simulation of both static and dynamic schedulers.
- Statistics of all or selected operations can be recorded and they can be analyzed using GridSim statistics analysis methods.

B.1.1 GridSim Architecture

GridSim is based on a layered and modular architecture was employed for GridSim:

- The first layer is concerned with the Java interface and the runtime machinery (JVM).
- The second layer is concerned with a basic discrete-event infrastructure built using the interfaces provided by the first layer. For the implementation of this discrete-event infrastructure SimJava [49], [9] was used.
- The third layer models and simulates the GridSim components.
- The fourth layer simulates the different schedulers used in GridSim.
- The final layer focuses on application and resource modeling with different scenarios using the services provided by the two lower-level layers for evaluating scheduling and resource management policies, heuristics, and algorithms.

The GridSim architecture is shown in Fig. B.1.

B.1.2 GridSim Components

GridSim Users. A Grid user is an instance of the *User entity*. Each user may differ from the rest of the users with respect to the following characteristics:

- Job characteristics e.g. job execution time, number of parametric replications, etc.;
- Scheduling optimization strategy: minimization of cost, minimization of time, or both;
- Activity rate e.g., how often it creates new job;
- Time zone where the user is supposed to be;
- Absolute deadline;

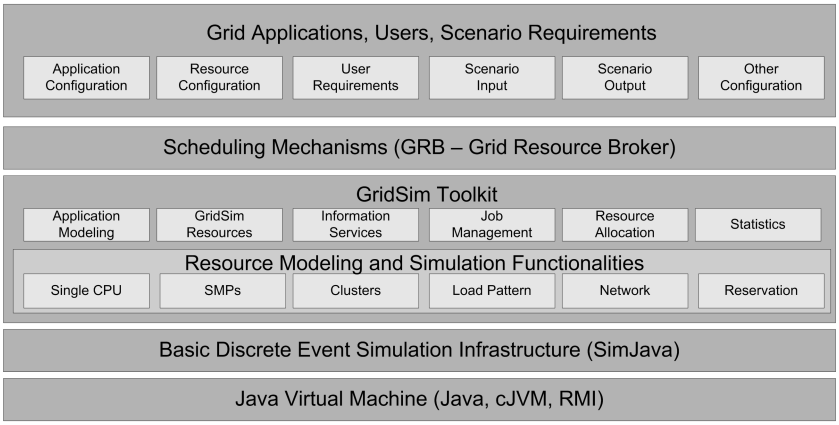


Figure B.1: GridSim Architecture.

- Absolute budget;
- Relaxation parameters (D and B factors) for the deadline and budget. They express the user's tolerance to the deviations from the established deadline and budget according to the processing requirements and available resources.

GridSim Resources. A Grid resource is an instance of the *Resource entity*. Each resource may differ from the rest of resources with respect to the following characteristics:

- Number of processors;
- Cost of processing;
- Speed of processing;
- Internal process scheduling policy (time shared or space shared);
- Local load factor;
- Time zone.

The resource speed and the job execution time is defined in terms of the ratings of standard benchmarks such as MIPS and SPEC.
Brokers query resources directly for their static and dynamic properties once the resource contact details are obtained.
GridSim Brokers. Each user is connected to an instance of the *Broker entity*. User's jobs are initially submitted to the broker who takes care about the scheduling of tasks according also to the user's scheduling policy. The broker dynamically gets a list of available resources from the global directory entity.
GridSim GIS (Grid Information Service). It offers resource registration services and keeps track of a list of resources available in the Grid. The information about resources is then used by the broker.
GridSim Inputs and Outputs. Input and output entities simulate the flow of information between GridSim entities. The use of separate entities for input and output enables a

networked entity to model full duplex and multi-user parallel communications.
GridSim Tasks. Tasks in GridSim are represented through *Gridlet objects*. A Gridlet contains all the information related to the job and its execution management details such as job length expressed in MIPS, disk I/O operations, the size of input and output files, and the job originator. These basic parameters are used for establishing the execution time, the time required to transport input and output files between users and remote resources and returning the processed Gridlets back to the originator along with the results.
The environment architecture is shown in Fig. B.2:

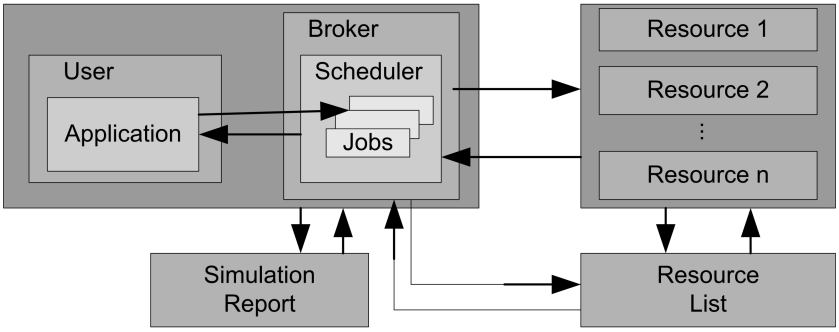


Figure B.2: Environmental Architecture.

B.1.3 GridSim Java Package Design

The GridSim package implements the following classes:

- *GridSim* - This is the main class of Gridsim package that must be extended by GridSim entities. The GridSim class adds networking and event delivery features, which allows synchronous or asynchronous communication for service access or delivery. The GridSim class supports methods for simulation initialization, management, and flow control. Its environment must be initialized to setup the simulation environment before creating any other GridSim entities at the user level. This method also prepares the system for simulation by creating *GridInformationService*, *GridSimShutdown*, and *GridStatistics*. The GridSim class supports static methods for sending and receiving messages between entities directly or via network entities, managing and accessing handle to various GridSim core entities, and recording statistics.

- *Input* - This class defines a port through which a simulation entity receives data from the simulated network. It maintains an event queue to serialize the data-in-flow and delivers to its parent entity. Simultaneous inputs can be modeled using multiple instances of this class.

- *Output* - This class defines a port through which a simulation entity sends data to the simulated network. It maintains an event queue to serialize the data-out-flow and delivers to the destination entity. Simultaneous outputs can be modeled by using multiple instances of this class.

- *PE (Processing Element)* - It is used to simulate CPU. Its capability is defined in terms of MIPS rating.
- *PEList* - It maintains a list of PEs of each machine.
- *Machine* - It represents a uniprocessor or shared memory multiprocessor machine.
- *MachineList* - It simulates a collection of machines. GridSim users define the connectivity among the machines in a collection.
- *ResourceCharacteristics* - It represents static properties of a resource such as resource architecture, OS, management policy (time or space shared), cost and time zone at which the resource is located along resource configuration.
- *GridResource* - It extends the GridSim class and gains communication and concurrent entity capability. The process of creating a Grid resource is as follows:
  - PE objects with a suitable MIPS/SPEC rating are created,
  - they are assembled together in order to create a machine.
  - one or more objects of the *Machine* are grouped to form a resource. A resource having a single machine with one or more PEs is managed as a time-shared system using round robin scheduling algorithm. A resource with multiple machines is treated as a distributed memory cluster and is managed as a space-shared system using first-come first served scheduling policy or its variants.
- *GridSimStantardPE* - It defines MIPS rating for a standard PE or enables the users to define their own MIPS/SPEC rating for a standard PE. This value can be used for creating PEs with relative MIPS/SPEC rating for GridSim resources and creating Gridlets with relative processing requirements.
- *ResourceCalendar* - This class implements a mechanism to support modeling local load on Grid resources that may vary according to the time zone, time, weekends, and holidays.
- *GridInformationService* - It provides Grid resource registration, indexing and discovery services.
- *Gridlet* - This class acts as job package that contains job length in MI, the length of input and out data in bytes, execution start and end time and the originator of job.
- *GridletList* - It maintains a list of Gridlets and supports methods for organizing them.
- *GridSimTags* - It contains various static command tags that indicate a type of action that needs to be undertaken by GridSim entities when they receive events.
- *ResGridlet* - It represents a Gridlet submitted to the resource for processing. It contains Gridlet object along with its arrival time and the ID of machine and PE allocated to it.

- *GridStatistics* - It records statistical data reported by other entities. It stores data objects with their label and timestamp. At the end of simulation, a *report-writer* entity queries recorded statistics of interest for report generation.
- *Accumulator* - It provides a placeholder for maintaining statistical values of a series of data added to it. It can be queried for mean, sum, standard deviation, and the largest and smallest values in the data series.
- *GridSimShutdown* - It waits for termination of all user entities to determine the end of simulation and signals the report-writer to generate the report. It signals also to other entities the end of the simulation.
- *GridSimRandom* - It provides static methods for incorporating randomness in data used for any simulation. The idea is to reflect the uncertainty and the randomness that could be present in the nature itself (e.g. the execution time of a Gridlet on a particular resource, can vary depending on the local load)

The relationship between GridSim packages is shown in Fig. B.3.

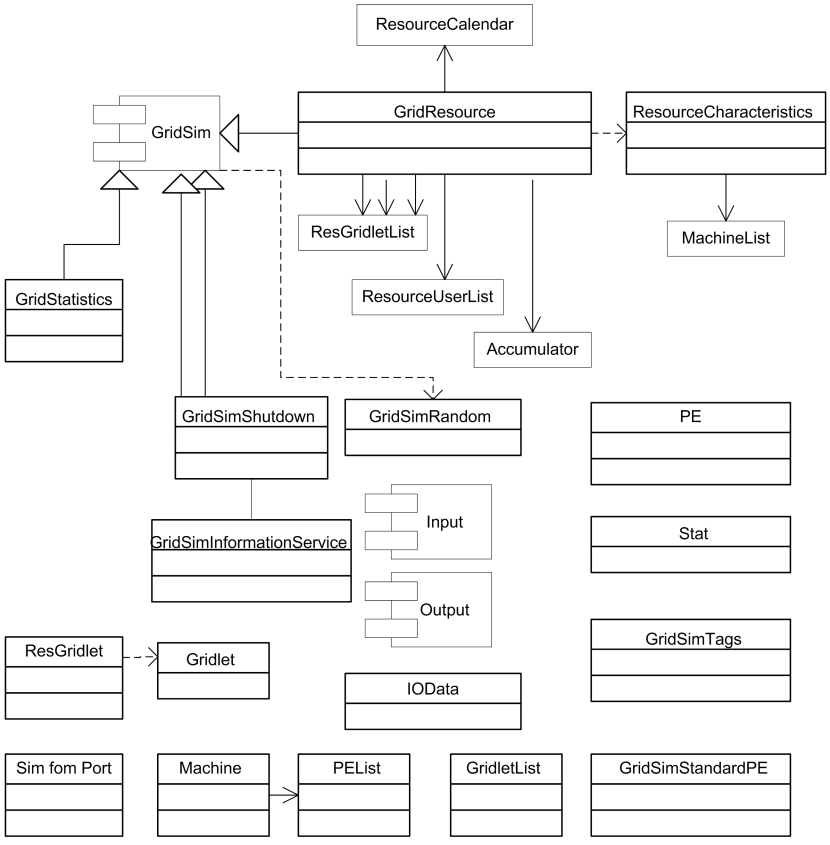


Figure B.3: Relationship Between GridSim Packages.

B.1.4 GridBroker Java Package Design

The key components of the broker are:

- *Experiment* - It is a placeholder for representing the configuration of simulation experiments. It provides methods for updating and querying the experiment parameters and status. The user entity invokes the broker entity and passes its requirements via the experiment object. On receiving an experiment from its user, the broker schedules Gridlets according to the optimization policy set for the experiment.
- *UserEntity* - It simulates the user and manages the forwarding of the user's requirements to the broker, recording of parameters of interest once results are received and ends the simulation.
- *Broker* - It simulates the Grid resource broker. Once an experiment is received from the user entity, it carries out resource discovery, determines deadline and budget values based on D and B factors, schedules the tasks, receives the results of application processing and records parameters of interest. On user's behalf it ends the simulation.
- *BrokerResource* - It is a placeholder for the broker to maintain a detailed record on the resources it uses for processing user applications. It is used for maintaining resource characteristics, a list of Gridlets assigned to the resource, the actual amount of MIPS available to the user and a report on the Gridlets processed. These measurements help in scheduling jobs dynamically at runtime.
- *ReportWriter* - It is used for creating a report at the end of each simulation.

The relationship between GridBroker packages is shown in Fig. B.4.

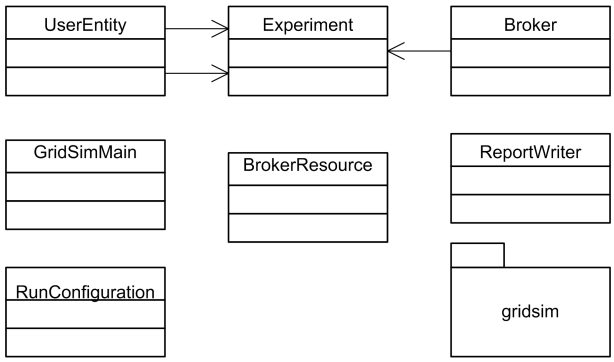


Figure B.4: Relationship Between GridBroker Packages.

B.1.5 Modeling Simulations with GridSim

Simulations with GridSim are in general modeled according to the following procedure:

1. Grid resources of different capability and configuration together with users with different requirements are created.

2. The user creates an experiment that contains an application description and sends the requirements to the broker via the experiment interface (a number of Gridlets are created and all parameters associated with jobs are defined).

3. The resource broker accesses the GIS and enquires the resource for its capability including cost and then develops a scheduling policy for assigning Gridlets to resources and coordinates the execution.

4. Gridlets that are mapped to specific resource are added to the Gridlets list in the broker resource.

5. For each of the resources, a number of Gridlets, according to the usage policy in order to avoid overloading resources with single user jobs, is selected.

6. The dispatcher then submits Gridlets to resources.

7. When the Gridlet processing completes, the resource returns it to the broker's Gridlet receptor module, which then measures and updates the runtime parameter. The prediction of the job consumption rate for making scheduling decisions.

8. Steps 4-7 continue until all the Gridlets are processed or the broker exceeds deadline or budget limits. The broker then returns updated experimental data along with processed Gridlets back to the user.

For facilitating the construction of different scenarios, GridSim adopts a graphical user interface (Visual Modeler - VM) with the following features:

- it enables the creation of many Grid users and resources with different requirements and characteristics,
- it generates into Java code the simulation scenario which can be compiled and run with GridSim,
- it saves project files and retrieves scenarios in/from XML language.

B.2 Introduced Changes to GridSim Components

In the following, the infrastructural and other changes done to the GridSim toolkit and GridSim components are going to be presented.

B.2.1 Infrastructural Changes

Changes to the infrastructure of the GridSim toolkit include improvements to the usability of the toolkit, the way how resource entities are created, simulation management and management of personal trust experience for every participant.

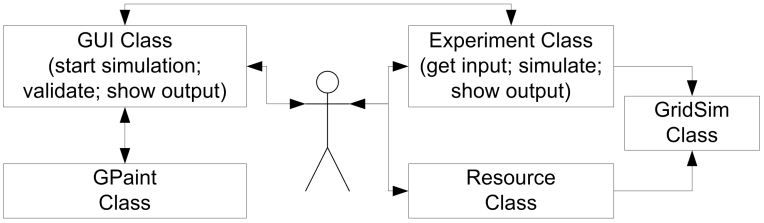


Figure B.5: Relationship of Class GUI to Other Classes.

Graphical User Interface (GUI)

Through the Graphical User Interface (GUI), a better management of the simulation scenarios is achieved. It is implemented in a separate class (class GUI). The relationship of this class to the others in the system is shown in Figure B.5.

The components used in the GUI class are from the Java Swing package. The types of components used are JMenu, JTextField, JButton, JTextArea, JFrame and JApplet. The user may load a preconfigured simulation scenario (class Experiment) from the menu "Simulation" and start the simulation through the button "Start Simulation". The overall output is displayed in the JTextArea on the right side. In the JTextArea on the left, information related to a specific Grid node could be achieved. The node number can be specified under "Node Info".
Specifying the provider's number, a consumer collaborated with, it is possible to obtain a graphical representation of the behavior (single behavior trust elements, according to the user's trust requirements and also a view of the absolute behavior trust (ABhvT)) of the participant during the last collaboration.
Upon the request, the output of the simulation could also be saved as a text file.
Events are handled through an action listener. The action listener listens only to the clicking of the JButtons.
The main method, is separated from class GridSim and integrated in part into the "action performed" method of the class GUI. In turn, the "chosen" Experiment class will be executed.

Grid Resource Creation

Resources are created in the same way as in the GridSim toolkit. The introduced change, as shown in Figure B.6, allows each user to "own" a resource. The idea behind this is that every Grid participant is potentially able to play both roles, consumer and provider, either separately or at the same time. In our simulation environment, each user "*owns*" a resource where other participants in the environment (other than himself) could send their jobs to be processed. Every other resource in the environment, depending on their capabilities and user's specific requirements (trust requirements included), is a potential partner for the upcoming collaborations.
Resources have the same characteristics as in GridSim. These characteristics are specified in the file, namely, resource name, architecture, operating system, number of machines, number of PEs, MIPS (millions instructions per second) of each PE, time zone, processing cost

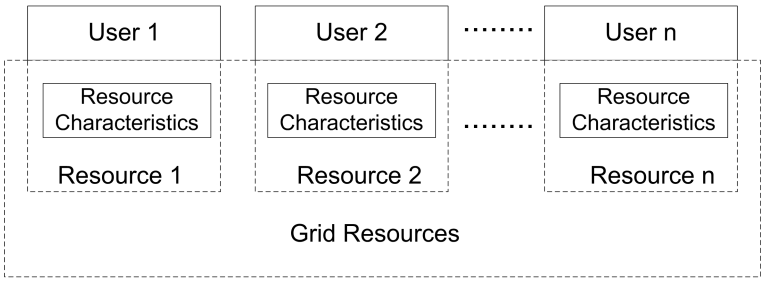


Figure B.6: Creating Grid Resources.

(expressed in Grid dollars), communication speed, random seed and resource load during peak hour, off-peak hour and holiday.

Class Experiment

The main method is separated from class GridSim. For each of the experiments, a separated class is constructed. Each Experiment class is in charge of simulating a single simulation scenario.
According to the characteristics of the scenarios that will be simulated, the needed changes are going to be introduced in the respective Experiment classes:

- number of users/resources,
- users/resources basic characteristics/requests,
- users/resources trust requirements (initialization strategies, experiences considered, error tolerance, identity and/or behavior trust considerations, behavior trust elements of interest, etc.),
- verification strategies and sub-strategies,
- deviations in the behavior of certain participant(s), introduced with different frequencies (affecting any, a collection or the entirety of the behavior trust elements)

The main purpose for the Experiment class is simulation and organization of the output

Trust Container

Each of the consumers in the environment creates a storage component where the trust values regarding all the collaboration partners are saved for personal use during future direct collaborations or to be offered to the others in the form of recommendations.
Before starting a collaboration, each user, after sorting out the most suitable resources according to his "computational requests", consults the personal trust container. To assess the trustworthiness of every single resource, the respective trust values (values) stored are going to be used. If personal experience with the single resources exists, then this experience is going to be used, otherwise, the initialization values expressed by the user according to the initialization strategy ("coldstart" or "warmstart") will be used.

After every verification (either "online" or "offline"), these trust values will be updated accordingly.

The purpose of such a container is to create a "history of personal direct collaborations", thus personal experience for each user.

B.2.2 Other Changes

The other changes affect part of the existing classes in GridSim for adapting it to the trust model implementation and simulation needs.

Certification Chanins - Assigning Certificates to Participants

A certificate in this model is represented by a "virtual document" possessed by participants. Here, only the names of the "issuer" and of the "receiver" are specified. Every participant could be an issuer of a certificate. At the top there are some certification authorities named after some German cities like: Marburg, Giessen and Frankfurt. The "certificates" are not really *issued* by the certification authorities; they are generated at the participants' site once they are created. GridSim already offers a way of assigning identities to users and resources. The respective certification authorities are generated according to the Figure B.7.

```
.....

cachain = new Hashtable();
String[] cas = new String[3];
cas[0]="Marburg";
cas[1]="Gießen";
cas[2]="Frankfurt";
int ca = 0;

.....

for (int count=0;count<num_user;count++){
        if (count<=1){
                int carandom = GridSim.rand.intSample(3);
                uArray2[count].idpartner_= cas[carandom];
                cachain.put(uArray2[count].name_, uArray2[count].idpartner_);
        }
        if (count<4&&count>1){
                int carandom = GridSim.rand.intSample(2);
                uArray2[count].idpartner_= uArray2[carandom].name_;
                cachain.put(uArray2[count].name_, uArray2[count].idpartner_);
        }
        if (count<=num_user&&count>=4){
                int carandom = GridSim.rand.intSample(4);
                uArray2[count].idpartner_= uArray2[carandom].name_;
                cachain.put(uArray2[count].name_, uArray2[count].idpartner_);
        }
```

Figure B.7: Generating Certification Authorities.

It means that from the entirety of the participants present in the environment:

- part of them is randomly "certified" directly by the "first order" certification authorities (Marburg, Giessen or Frankfurt),
- part of them is randomly "certified" by the "second order" certification authorities (participants which were directly certified from the "first order" CAs and
- the rest is randomly "certified" by the "third order" certification authorities (participants "certified" from the "second order" CAs).

A typical certification chain resulting from the simulations is represented by the certification graph in Fig. B.8.

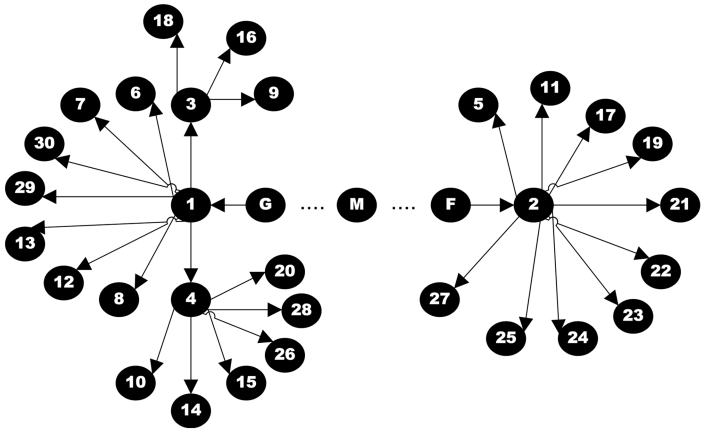


Figure B.8: Example of an Established Certification Chain Among Participants in the Environment.

Integration of Participants' Trust Requirements

The participants' (users or resources) normal requirements and characteristics are extended with the trust requirements. Normal requirements and characteristics under GridSim are: For every user:

- user name;
- baud rate; max. simulation time; scheduling strategy; successive experiment delay;
- gridlet size; gridlet min. deviation; gridlet max. deviation;
- length size; length min. deviation; length max. deviation;
- file size; file min. deviation; file max. deviation ;
- output size; output min. deviation; output max. deviation;
- budget and deadline.

For every resource:

- resource name;
- baud rate; peak load; off-peak load; holiday load;
- architecture; operating system; time zone; Grid$ per application operation (price in "Grid dollars"); allocation policy; list of machines (together with their PEs and MIPS).

Figure B.9 is an example of the integration of the participant's trust requirements to the other requirements mentioned above.

```
//Specification of a User
String nameU = "User_Name";
gArray[0]= Experiment.creatRes("Resource_Name", errorRate);
GridResource resource0 = gArray[0];
 uArray2[0]= new Experiment(username, 560.00, 0, total_resource, 0,
                    weightPersonal, weightThird,
resource0,acceptRecs,verifFreq,verifStrat,errorTolerance,collabID);
```

Figure B.9: Integration of the Trust Requirements for a GridSim User Entity.

In Fig. B.9:

- errorRate - percentage of erroneous tasks a resource should send back to the consumer;
- weightPersonal - importance of the previous personal experience (if any) or personal disposition expressed through the initial trust in the decision making process. Its value vary in $[0.0, 1.0]$;
- weightThird - importance of third parties' experience (recommendations) in the decision making process. Its values vary also in $[0.0, 1.0]$ and the sum with weightPersonal should be 1.0;
- acceptRecs - participants in the environment to get recommendations from.Its values are 1.0 (known partners only), 0.5 (known partners only and their known partners), 0.0 (everyone);
- verifFreq - minimal verification frequency;
- verifStrat - the verification strategy. It has the values "ONLINE" and "OFFLINE";
- errorTolerance - tolerance of the consumer toward the possible errors that a provider introduces during the collaboration. Its minimal value is 0;
- collabID - identity trust value for the considered partner(s). The considered values during the experiments were 1.0 (participants certified directly from first order CAs only), 0.5 (participants certified directly from first order CAs and everyone that was certified through these participants), 0.0 (everyone in the environment).

Verification Strategies

The implementation of the "online" and "offline" verification strategies is shown in Fig. B.10 and B.11.

```
if (this.verstrat_=="ONLINE"){
 if (verUs&&resErr){
  Integer g = (Integer)this.resanfrG_.get(tempKey);
  g +=1;
  this.resanfrG_.put(tempKey, g);
  Integer tempvalue1_ = (Integer)this.resnonerrorG_.get(tempres.get_name());
  Integer tempvalue2_ = (Integer)this.resanfrG_.get(tempres.get_name());
  oldTrust=(oldTrust+(tempvalue1_.doubleValue()/tempvalue2_.doubleValue())) / 2;
  Integer z=tempvalue2_-tempvalue1_;
  this.foundErrors +=1;
  if(z>this.maxerror_){
    errorflag_="1";
                     }
  }
  else{
    Integer g = (Integer)this.resanfrG_.get(tempKey);
    g +=1;
    this.resanfrG_.put(tempKey, g);
    Integer ge = (Integer)this.resnonerrorG_.get(tempKey);
    ge +=1;
    this.resnonerrorG_.put(tempKey, ge);
    Integer tempvalue1_=(Integer)this.resnonerrorG_.get(tempres.get_name());
    Integer tempvalue2_ = (Integer)this.resanfrG_.get(tempres.get_name());
   oldTrust=(oldTrust+(tempvalue1_.doubleValue()/tempvalue2_.doubleValue())) / 2;
    }
  if (errorflag_=="1"){
    this.gkeit_.put(tempKey,new Double(-1.0));
    tempg.put(gridlet.getGridletID(),new Double(-1.0));
    this.gA_.put(tempKey,tempg);
  }
  else{
    tempg.put(gridlet.getGridletID(),oldTrust);
    this.gA_.put(tempKey,tempg);
    this.gkeit_.put(tempKey,oldTrust);
  }
 }
  // stores the received Gridlet into a new GridletList object
  this.receiveList_.add(gridlet);
  tempres.statusflag_="FREE";
  if (verUs) {
    //Trust calculation
    this.verification=-((1-this.vermin_)*oldTrust)+1;
    gridlet.checked_=true;
  }
  tempDouble = 0.0;
  tempDoubleV = 0.0;
}
```

Figure B.10: "Online" Verification Strategy.

Class GridResource

In order to have some "natural" resource behaviors during the simulations, some status flags are implemented:

```
if (this.verstrat_=="OFFLINE"){
  for(int f=0;f<this.receiveList_.size();f++){
  gridlet = (Gridlet) this.receiveList_.get(f);
  String tempKey = gridlet.getResourceName(gridlet.getResourceID());
  Hashtable tempg=(Hashtable)gA_.get(tempKey);
  oldTrust = (Double)this.gkeit_.get(tempKey);
  boolean verUs = this.verification<=Math.random();
  if (verUs&&gridlet.error_==true){
    Integer g = (Integer)this.resanfrG_.get(tempKey);
    g +=1;
    this.resanfrG_.put(tempKey, g);
    Integer tempvalue1_=(Integer)this.resnonerrorG_.get(tempres.get_name());
    Integer tempvalue2_ = (Integer)this.resanfrG_.get(tempres.get_name());
   oldTrust=(oldTrust+(tempvalue1_.doubleValue()/tempvalue2_.doubleValue()))  /2;
    this.foundErrors += 1;
  }
  else{
    Integer g = (Integer)this.resanfrG_.get(tempKey);
    g +=1;
    this.resanfrG_.put(tempKey, g);
    Integer ge = (Integer)this.resnonerrorG_.get(tempKey);
    ge +=1;
    this.resnonerrorG_.put(tempKey, ge);
    Integer tempvalue1_=(Integer)this.resnonerrorG_.get(tempres.get_name());
    Integer tempvalue2_ = (Integer)this.resanfrG_.get(tempres.get_name());
   oldTrust=(oldTrust+(tempvalue1_.doubleValue()/tempvalue2_.doubleValue()))  /2;
  }
  tempg.put(gridlet.getGridletID(),oldTrust);
  this.gA_.put(tempKey,tempg);
  this.gkeit_.put(tempKey,oldTrust);
  if (verUs){
  gridlet.checked_=true;
  this.verification=-((1-this.vermin_)*oldTrust)+1;
  }
 }
}
```

Figure B.11: "Offline" Verification Strategy.

- free - the resource is online and accepts the user requests,
- busy - the resource is online but accepts no requests,
- off - resource is "offline".

The idea behind this categorization is to simulate the following behavior trust elements:

- availability - if free is "true" than the specific resource available,
- accessibility - if busy is "true" than resource is available but not accessible,
- if off is "true", than the specific resource is neither accessible nor available.

Introducing Processing Errors

Errors in the responses a resource sends back to a user are also implemented through an "error flag". Errors are introduced either randomly or at a certain frequency.
An example of the error "integration" in the response a resource send back to a user is given in Figure B.12.

```
boolean resErr = Math.random()<=tempIntG;
boolean verUs = Math.random()<=this.verification;
String errorflag_="0";
if (resErr) {
    this.totalErrors +=1;
    gridlet.error_=true;
}
```

Figure B.12: Introducing Errors.

Class ResourceCharacteristics

An additional change to the RessourceCharacteristics class is the "error" variable (Fig. B.13).

```
Variabel public Double Error_
```

Figure B.13: "Error" Variable to the Resource Characteristics.

It is used for declaring the frequency of error introduction from a specific resource to its response. Allowed values, which specify the percentage of responses containing an error, are in the segment $[0, 1]$.

Class Gridlet

Two new variables added to class Gridlet, *checked_* and *error_* (Fig. B.14 and Fig. B.15).

```
public   Boolean   checked_   =
false;
```

Figure B.14: Variable for showing if a Gridlet was verified.

The *checked_* variable has two statuses:

- "false" - it means that the Gridlet was not checked for errors and
- "true" - the Gridlet was verified for any possible errors.

The *error_* variable has also two statuses:

- "false" - it means that the Gridlet was not containing any error and
- "true" - the Gridlet was erroneous.

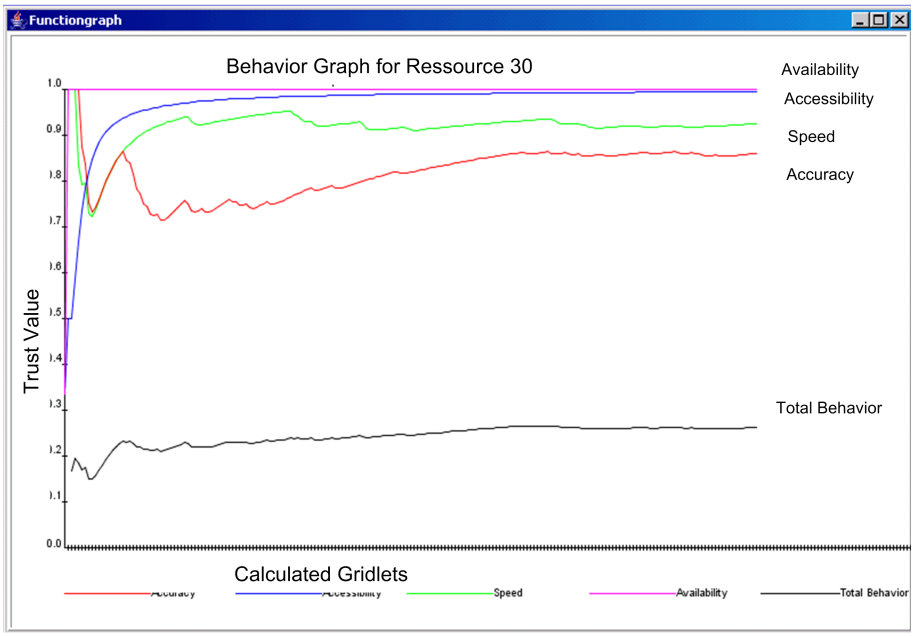


Figure B.15: Variable for showing if a Gridlet is erroneous.

Simulation Time

The total simulation time is calculated in seconds based on:

- time needed for sorting out the present resources based on their trust value and user/application trust requirements. The time is a function that depends on the time needed for calculating identity trust and behavior trust (personal experience or third parties' experience; ABhvT or RBhvT);
- time to gather trust information;
- time to evaluate partners;
- time to send Gridlets to resources;
- time for processing the Gridlet at the Grid resources;
- time to receive the Gridlets back;
- time for verification.

This is shown also in Fig. B.16.

Simulation Output

The output of the simulation is text and graphical. In the text, output information on the simulation process is given. This information has to do with the participants involved in the

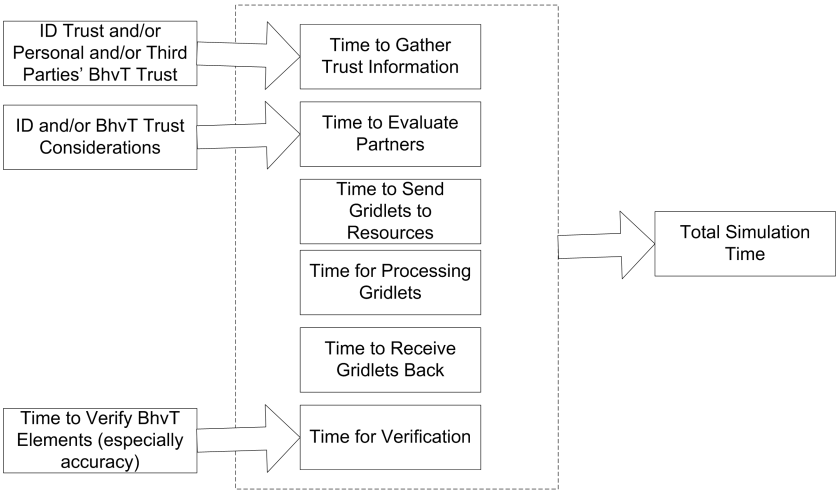


Figure B.16: Total Simulation Time.

simulation, participants' characteristics, behavior of the participants during the simulation, simulation statistics (total simulation time, time for processing Gridlets, results verified), etc.

In the graphical output, upon user request, it is possible to have for every participant a graphical representation of the behavior that its counterparts showed during the very last collaboration. Here, absolute behavior trust (ABhvT) and all the behavior trust elements under verification (RBhvT) are presented. An example is presented in Fig. B.17 and B.18.

Furthermore, each of the participants has the possibility to compare the behavior of each of their partners during the last collaboration with the bahavior shown by them previously (Fig. B.19).

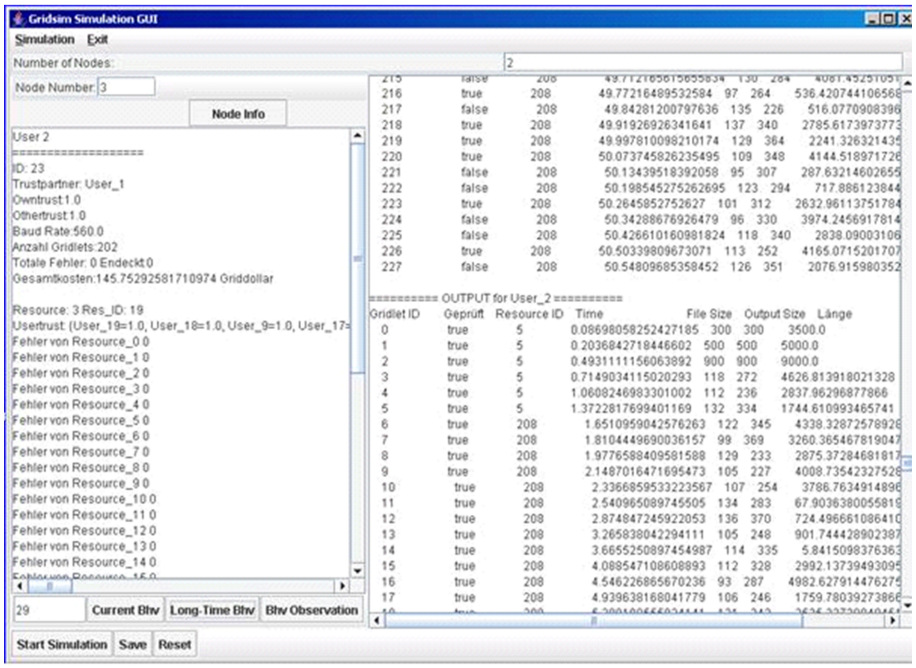


Figure B.17: Simulation Output in the Graphical User Interface.

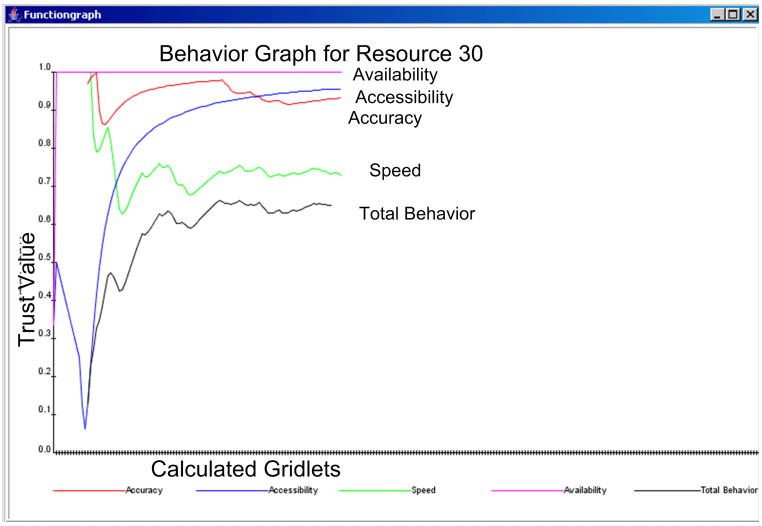


Figure B.18: Graphical Representation of the Behavior of a Participant in the Last Collaboration (for the participant under observation, except for availability all the other behavior trust elements contained errors).

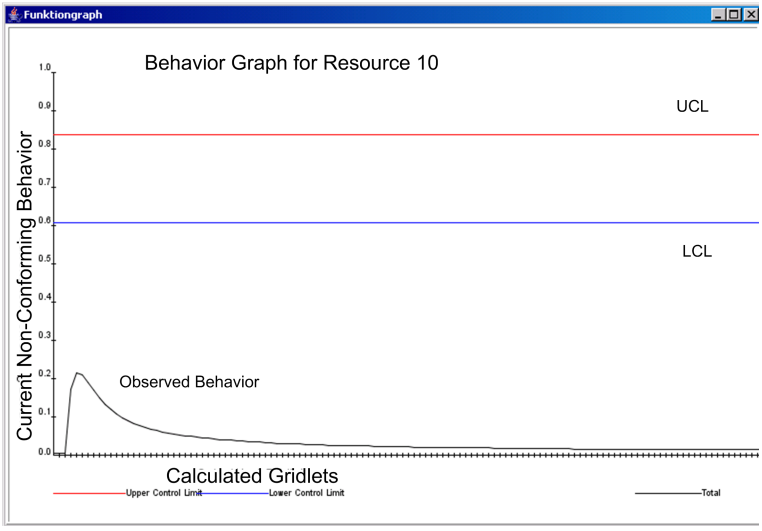


Figure B.19: Comparing Current Behavior to the One shown Previously (the shown results were obtained for an injected error rate of 5%, clearance number 0, minimal verification frequency 10%; limits calculated according to formulas 5.30 and 5.31).

Bibliography

[1] Against TCPA.
http://www.againsttcpa.com/.

[2] ALLEXPERTS.
http://www.allexperts.com.

[3] Black Sabbath Game.
http://www.imjustabill.com/blacksabbathgame.

[4] Bricks: A Performance Evaluation System for Grid Computing Scheduling Algorithms.
http://ninf.apgrid.org/bricks/.

[5] ChicSim(The Chicago Grid Simulator).
http://people.cs.uchicago.edu/ krangana/ChicSim.html.

[6] CNET.
http://www.cnet.com.

[7] Commodity Grid (CoG) Kits.
http://wiki.cogkit.org/index.php/Main_Page.

[8] Condor.
http://www.cs.wisc.edu/condor/.

[9] Distributed Simjava.
http://www.thesimguy.com/Projects/websim/simjava/.

[10] eBay.
http://www.ebay.com.

[11] EPINIONS.
http://www.epinions.com.

[12] Erdös at Wikipedia.
http://en.wikipedia.org/wiki/Erdös_number.

[13] Erdös Number Project.
http://www.oakland.edu/enp/.

[14] Fifth Utility Solutions.
http://www.utilitycomputing.org.

[15] Flow Assurance & Optimisation of Oil & Gas Production: Research & Development. http://www.feesa.net/research.htm.

[16] Free Haven System. http://www.freehaven.net.

[17] Globus Monitoring and Discovery Service. http://www.globus.org/toolkit/mds.

[18] Globus Security. http://www.globus.org/toolkit/docs/4.0/security/.

[19] Globus Toolkit. http://www.globus.org.

[20] Globus Toolkit Gridmap File. http://www.globus.org/toolkit/docs/4.0/security/key-index.html.

[21] GRID Certification Authorities. http://marianne.in2p3.fr/ca/ca-table-ca.html.

[22] Grid Computing Definition at Wikipedia. http://www.wikipedia.org.

[23] Grid eXplorer (GdX). http://www.lri.fr/ fci/GdX/.

[24] Grid Portal Development Kit (GPDK). http://doesciencegrid.org/projects/GPDK/.

[25] Grid'5000. http://www.grid5000.org.

[26] GridFtp. http://www.globus.org/grid_software/data/gridftp.php.

[27] GridSim: A Grid Simulation Toolkit for Resource Modelling and Application Scheduling for Parallel and Distributed Computing. http://www.gridbus.org/gridsim/.

[28] IBM Trust Establishment Policy Language. http://www.haifa.il.ibm.com/projects/software/e-Business/TrustManager/index.html.

[29] IBM Trust Establishment Policy Language. http://www.haifa.il.ibm.com/projects/software/e-Business/TrustManager/PolicyLanguage.html.

[30] JavaSim. http://javasim.ncl.ac.uk/.

[31] Kasparov's Number. http://www.chessnetwork.com/ncn/july20.htm.

[32] LCG Risk Analysis - 30 Oct 2003 (v2).
http://proj-lcg-security.web.cern.ch/proj-lcg-security/RiskAnalysis/risk.html.

[33] Legion.
http://legion.virginia.edu.

[34] Merriam Webster Dictionary.
http://www.m-w.com/.

[35] MicroGrid: Online Simulation Tools for Grids, Distributed Systems and the Internet.
http://www-csag.ucsd.edu/projects/grid/microgrid.html.

[36] National Research Grid Initiative (NAREGI).
http://www.naregi.org/index_e.html.

[37] OPENPRIVACY.
http://www.openprivacy.org.

[38] OptorSim.
http://www.gridpp.ac.uk/demos/optorsimapplet/.

[39] Oracle of Bacon.
http://oracleofbacon.org/.

[40] Oracle of Baseball.
http://www.baseball-reference.com/oracle/.

[41] PARSEC: Parallel Simulation Environment for Complex Systems.
http://pcl.cs.ucla.edu/projects/parsec/.

[42] PERMIS.
http://www.permis.org/.

[43] PKI.
http://www.pki-page.org/.

[44] Prisoner's Dilemma.
http://en.wikipedia.org/wiki/Prisoner's$_d ilemma$.

[45] Security Assertion Markup Language (SAML).
http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=security.

[46] SETI@HOME.
http://setiathome.ssl.berkeley.edu.

[47] Shibboleth.
http://shibboleth.internet2.edu/.

[48] SimGrid Project.
http://simgrid.gforge.inria.fr/.

[49] SimJava.
http://www.dcs.ed.ac.uk/home/hase/simjava/.

[50] TeraGrid.
http://www.teragrid.org/.

[51] The Distributed ASCI Supercomputer 2 (DAS-2).
http://www.cs.vu.nl/das2/.

[52] Trusted Computing @ Wikipedia.
http://en.wikipedia.org/wiki/Trusted_computing.

[53] UNICORE.
http://www.unicore.org.

[54] Web Service Level Agreements (WSLA) Project - SLA Compliance Monitoring for e-Business on Demand.
http://www.research.ibm.com/wsla/.

[55] Windows Vista Home Page.
http://www.microsoft.com/windowsvista/.

[56] World Wide Web Consortium.
http://www.w3.org/.

[57] ISO 8402: Quality Management and Quality Assurance - Vocabulary., 1994.
http://www.standards.com.au/PDFTEMP/Previews/OSH/as/as8000/8400/8402.pdf.

[58] Advogato's Trust Metric., 2000.
http://www.advogato.org/trust-metric.html.

[59] XML Encryption Syntax and Processing., December 2002.
http://www.w3.org/TR/xmlenc-core/.

[60] XML-Signature Syntax and Processing., February 2002.
http://www.w3.org/TR/xmldsig-core/.

[61] Grid Computing Basics: The Evolution., 2004.
http://www.grid.org/about/gc/evolution.htm.

[62] eXtensible Access Control Markup Language - XACML., 2005.
http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=xacml.

[63] Web Services Secure Conversation Language (WS-SecureConversation)., February 2005.
http://specs.xmlsoap.org/ws/2005/02/sc/WS-SecureConversation.pdf.

[64] ABDUL-RAHMAN, A., AND HAILES, S. A Distributed Trust Model. In *Proceedings of the Workshop on New security Paradigms, Langdale, Cumbria, United Kingdom* (1997), pp. 48–60.

[65] ABDUL-RAHMAN, A., AND HAILES, S. Using Recommendations for Managing Trust in Distributed Systems. In *Proceedings of the IEEE International Conference on Communication, Kuala Lumpur, Malaysia* (1997).

[66] ABDUL-RAHMAN, A., AND HAILES, S. Supporting Trust in Virtual Communities. In *Proceedings of the Thirtythird Hawaii International Conference on System Sciences, Hawaii, USA* (2000), vol. 6, p. 6007.

[67] ABERER, K., AND DESPOTOVIC, Z. Managing Trust in a Peer-2-Peer Information System. In *Proceedings of the Tenth International Conference on Information and Knowledge Management (CIKM), Atlanta, GA, USA* (2001), pp. 310–317.

[68] ABRAMSON, D., SOSIC, R., GIDDY, J., AND HALL, B. Nimrod: A Tool for Performing Parameterised Simulations Using Distributed Workstations. In *Proceedings of Fourth IEEE Symposium on High Performance Distributed Computing. Los Alamitos, CA, USA* (1995).

[69] ADAMS, C., AND FARRELL, S. IETF RFC 2510: PKI Certificate Management Protocols., March 1999.
ftp://ftp.rfc-editor.org/in-notes/rfc2510.txt.

[70] AKEROF, G. The Market for Lemons: Quality Uncertainty and the Market Mechanisms. *The Quarterly Journal of Economics 84* (1970), 488–500.

[71] AL-ALI, R., AMIN, K., VON LASZEWSKI, G., HATEGAN, M., RANA, O., W. D., AND ZALUZEC, N. QoS Support for High-Performance Scientific Applications. In *Proceedings of the IEEE/ACM Fourth International Symposium on Cluster Computing and the Grid (CCGrid 2004). Chicago IL, USA: IEEE Computer Society Press* (2004).

[72] AL-RIYAMI, S., AND PATERSON, K. Certificateless Public Key Cryptography. In *Proceedings of Advances in Cryptology - ASIACRYPT 2003, Ninth International Conference on the Theory and Application of Cryptology and Information Security (Asiacrypt), Taipei, Taiwan* (2003), pp. 452 – 473.

[73] ALI, A. S., RANA, O., AND WALKER, D. WS-QoC: Measuring Quality of Service Compliance. In *Proceeding of the Second International Conference on Service Oriented Computing - Short Papers (ICSOC), New York, USA* (2004), pp. 16–25.

[74] ALUNKAL, B., VELJKOVIC, I., AND VON LASZEWSKI, G. Reputation-Based Grid Resource Selection. In *Workshop on Adaptive Grid Middleware, New Orleans, Louisiana, USA* (2003).

[75] AMENDOLIA, S., ESTRELLA, F., HASSAN, W., HAUER, T., MANSET, D., MCCLATCHEY, R., ROGULIN, D., AND SOLOMONIDES, T. MammoGrid: A Service Oriented Architecture Based Medical Grid Application. In *Proceedings of the Third International Conference on Grid and Cooperative Computing, Wuhan, China* (2004), pp. 939–942.

[76] AMIN, K., VON LASZEWSKI, G., AND MIKLER, A. A Trust Evaluation Model for Ad Hoc Grids.
http://www.cogkit.org/viewcvs/viewcvs.cgi/papers/src/adhoc-trust/src/vonLaszewski-adhoc-trust.pdf.

[77] ANTONIU, G., BERTIER, M., BOUGE, L., CARON, E., DESPREZ, F., JAN, M., MONNET, S., AND SENS, P. GDS: an Architecture Proposal for a Grid Data-Sharing Service. Tech. Rep. 2005-28, Laboratoire de l'Informatique du Paralllisme (LIP), June 2005.

[78] ASHRI, R., RAMCHURN, S. D., SABATER, J., LUCK, M., AND JENNINGS, N. R. Trust Evaluation Through Relationship Analysis. In *Proceedings of the Fourth International Joint Conference on Autonomous Agents and Multi-Agent Systems, Utrecht, The Netherlands* (2005), pp. 1005–1011.

[79] AZZEDIN, F., AND MAHESWARAN, M. Evolving and Managing Trust in Grid Computing Systems. In *Conference on Electrical and Computer Engineering, Canada.* (2002), I. C. S. Press, Ed., pp. 1424–1429.

[80] AZZEDIN, F., AND MAHESWARAN, M. Integrating Trust into Grid Resource Management Systems. In *International Conference on Parallel Processing, Vancouver, B.C., Canada* (2002), p. 47.

[81] AZZEDIN, F., AND MAHESWARAN, M. Towards Trust-Aware Resource Management in Grid Computing Systems. In *Proceedings of the Second IEEE/ACM International Symposium on Cluster Computing and the Grid, Berlin, Germany* (2002), pp. 452–457.

[82] AZZEDIN, F., AND MAHESWARAN, M. A Trust Brokering System and Its Application to Resource Management in Public-Resource Grids. In *Proceeding of Eighteenth International Parallel and Distributed Processing Symposium (IPDPS), Santa Fe, New Mexico* (2004), p. 22a.

[83] BACHARACH, M., AND GAMBETTA, D. Trust as Type Interpretation. In *Trust and Deception in Virtual Societies*, C. CASTELFRANCHI and Y. TAN, Eds. Kluwer Publisher, 2001.

[84] BAILEY, B., GURAK, L., AND KONSTAN, J. Do You Trust Me? An Examination of Trust in Computer-Mediated Exchange. In *Human Factors and Web Development.*, second ed. Mahwah, NJ: Lawrence Erlbaum, 2002.
http://www.isc.umn.edu/research/papers/Ecommerce2000.pdf.

[85] BARBER, B. *Logic and Limits of Trust.* New Jersey: Rutgers University Press., 1983.

[86] BERMAN, F., FOX, G., AND HEY, A., Eds. *Grid Computing: Making the Global Infrastructure a Reality.* Wiley Publishing, 2003.

[87] BLAZE, M., FEIGENBAUM, J., IOANNIDIS, J., AND KEROMYTIS, A. D. The Role of Trust Management in Distributed System Security. *Secure Internet*

Programming: Security Issues for Distributed and Mobile ObjectsJ. VITEK, J., and JENSEN, C., Lecture Notes in Computer Science, Springer, Berlin 1603 (1999), 183–210.

[88] BLAZE, M., FEIGENBAUM, J., AND LACY, J. Decentralized Trust Management. In *Proceedings of the IEEE Symposium on Security and Privacy, Oakland, CA, USA* (1996), pp. 164–173.

[89] BLAZE, M., IOANNIDIS, J., AND KEROMYTIS, A. Trust Management for IPsec. *ACM Transactions on Information and System Security 5, Issue 2* (2002), 95–118.

[90] BLOOMBERG, J., AND SCHMELZER, R. A Guide to Securing XML and Web Services., January 2004. http://whitepapers.itsj.com/detail/RES/1073404572_221.html.

[91] BRAINOV, S., AND SANDHOLM, T. Contracting with Uncertain Level of Trust. *Computational Intelligence 18, Issue 4* (2002), 501–514. http://www.cs.cmu.edu/ sandholm/contracting_with_uncertain_trust.ci.pdf.

[92] BRIN, S., AND PAGE, L. The Anatomy of a Large-Scale Hypertextual Web Search Engine. *Computer Networks and ISDN Systems 30, Issue 1-7* (1998), 107–117.

[93] BULLER, D. B., BURGOON, J. K., BUSLIG, A. L., AND ROIGER, J. F. Interpersonal deception. *Journal of Language and Social Psychology 13* (1994), 396–417.

[94] BUSKENS, V. The Social Structure of Trust. *Social Networks 20* (1998), 265–289.

[95] BUYYA, R. *Economic-based Distributed Resource Management and Scheduling for Grid Computing.* PhD thesis, Monash University, Melbourne, Australia, April 2002. http://www.buyya.com/thesis.

[96] CARBONE, M., NIELSEN, M., AND SASSONE, V. A Formal Model for Trust in Dynamic Networks. In *Proceedings of the First International Conference on Software Engineering and Formal Methods* (2003), IEEE Computer Society Press, pp. 54–63.

[97] CARTER, J., BITTING, E., AND GHORBANI, A. Reputation Formalization for an Information-Sharing Multi-Agent Sytem. *Computational Intelligence 18, Issue 2* (2002), 515–534.

[98] CASANOVA, H., AND DONGARRA, J. Netsolve: A Network Server for Solving Computational Science Problems. Technical report cs-95-313, University of Tennessee, November 1995.

[99] CASTELFRANCHI, C. Artificial Liars: Why Computers will (Necessarily) Deceive Us and Each Other. *Ethics and Information Technology. Kluwer Academic Publishers. 2* (2000), 113–119.

[100] CASTELFRANCHI, C., AND FALCONE, R. Principles of Trust for MAS: Cognitive Anatomy, Social Importance, and Quantification. In *Proceedings of the International Conference on Multi-Agent Systems, Paris, France* (1998), pp. 72–79.

[101] CASTELFRANCHI, C., AND FALCONE, R. Social Trust: Cognitive Anatomy, Social Importance, Quantification and Dynamics. In *Autonomous Agents Workshop on "Deception, Fraud, and Trust in Agent Societies", St. Paul, Minnesota, USA* (1998), pp. 35–49.

[102] CASTELFRANCHI, C., AND FALCONE, R. Trust and Control: A Dialectic Link. *Applied Artificial Intelligence Journal 14, Issue 8* (2000), 799–823. http://www.istc.cnr.it/T3/download/Trust-and-control.pdf.

[103] CASTELFRANCHI, C., AND FALCONE, R. Socio-Cognitive Theory of Trust. In *Open Agent Societies: Normative Specifications in Multi-Agent Systems.*, J. PITT, Ed. Wiley, London, UK, 2005. http://alfebiite.ee.ic.ac.uk/docs/papers/D1/ab-d1-cas+fal-soccog.pdf.

[104] CHANG, B., CRARY, K., DELAP, M., HARPER, R., LISZKA, J., MURPHY, T., AND PFENNING, F. Trustless Grid Computing in ConCert. In *Proceedings of Third International Workshop on Grid Computing, Berlin, Germany* (2002), pp. 112–125. http://www-2.cs.cmu.edu/_concert/papers/grid2002/grid2002.pdf.

[105] CHATTERJEE, S., SABATA, B., AND SYDIR, J. ERDoS QoS Architecture. Tech. rep., SRI Technical Report ITAD-1667-TR-98-075. Menlo Park, CA, USA, 1998.

[106] CHEN, R., AND YEAGER, W. Poblano: A Distributed Trust Model for Peer-to-Peer Networks. Tech. rep., Sun Microsystems, 2003. http://www.jxta.org/docs/trust.pdf.

[107] CHOPRA, K., AND WALLACE, W. Trust in Electronic Environments. In *Proceedings of the Thirtysixth Hawaii International Conference on System Sciences, Big Island, Hawaii* (2003), pp. 331–340.

[108] CHRISTIANSON, B., AND HARBISON, W. Why Isn't Trust Transitive? In *Proceedings of the International Workshop on Security Protocols, London, UK* (1997), Springer-Verlag, pp. 171–176.

[109] CHU, Y., FEIGENBAUM, J., LAMACCHIA, B., RESNICK, P., AND STRAUSS, M. Referee: Trust Management for Web Applications. *Computer Networks and ISDN Systems 29, Issue 8* (1997), 953–964.

[110] COMPAQ, HEWLETT-PACKARD, IBM, INTEL, AND MICROSOFT. Building a Foundation of Trust in the PC., 2000. http://www.trustedpc.org.

[111] COMPAQ, HEWLETT-PACKARD, IBM, INTEL, AND MICROSOFT. TCPA Design Philosophies and Concepts Version 1., 2000. http://www.trustedcomputing.org/docs/designv1_0final.pdf.

[112] COMPAQ, HEWLETT-PACKARD, IBM, INTEL, AND MICROSOFT. TCPA PC Specific Implementation Version 1.00., 2001. http://www.trustedcomputing.org/docs/TCPA_PCSpecificSpecification_v100.pdf.

[113] COMPAQ, HEWLETT-PACKARD, IBM, INTEL, AND MICROSOFT. TCPA Main Specification version 1.1b., 2002.
http://www.trustedcomputing.org/docs/main v1_1b.pdf.

[114] COMPAQ, HEWLETT-PACKARD, IBM, INTEL, AND MICROSOFT. TPM FAQ., 2002.
http://www.trustedcomputing.org/docs/TPM_QA_071802.pdf.

[115] CORNELLI, F., DAMIANI, E., AND CAPITANI, S. D. Choosing Reputable Servents in a P2P Network. In *Proceedings of the Eleventh International Conference on World Wide Web, Honolulu, Hawaii, USA* (2002), pp. 376–386.

[116] CRAMPTON, J., LIM, H., PATERSON, K., AND PRICE, G. A Certificate-Free Grid Security Infrastructure Supporting Password-Based User Authentication. In *Proceedings of the Sixth Annual PKI R&D Workshop, Gaithersburg, Maryland, USA* (2007).

[117] CZAJKOWSKI, K., FERGUSON, D., FOSTER, I., FREY, J., GRAHAM, S., MAGUIRE, T., SNELLING, D., AND TUECKE, S. From Open Grid Services Infrastructure to WSResource Framework: Refactoring & Evolution. Version 1.1., May 2004.
http://www.chinagrid.net/dvnews/upload/2005_04/05040200359561.pdf.

[118] CZYZYK, J., MESNIER, M., AND MORE, J. The Network-Enabled Optimization System (NEOS) Server. Preprint mcs-p615-0996, Argonne National Laboratory, Argonne, IL, USA, 1996.

[119] DAMIANI, E., DI VIMERCATI, D., PARABOSCHI, S., SAMARATI, P., AND VIOLANTE, F. A Reputation-Based Approach for Choosing Reliable Resources in Peer-to-Peer Networks. In *Proceedings of the Ninth ACM Conference on Computer and Communications Security, Washington, USA* (2002), pp. 207–216.

[120] DAMIANI, E., DI VIMERCATI, D. C., PARABOSCHI, S., SAMARATI, P., AND VIOLANTE, F. A Reputation-Based Approach for Choosing Reliable Resources in Peer-to-Peer Networks. In *Proceedings of the Ninth ACM Conference on Computer and Communications Security, New York, USA* (2002), pp. 207–216.

[121] DE ROURE, D., BAKER, M., JENNINGS, N., AND SHADBOLT, N. The Evolution of the Grid. In *Grid Computing-Making the Global Infrastructure a Reality.*, G. BERMAN, F.and Fox and A. HEY, Eds. Wiley Publishing, 2003, pp. 65–100.

[122] DE ROURE, D., JENNINGS, N., AND SHADBOLT, N. Research Agenda for the Semantic Grid: A Future E-Science Infrastructure., 2001.
www.semanticgrid.org/v1.9/semgrid.pdf.

[123] DECAIRE, M. The detection of deception via non-verbal detection cues. Law Library Edited By SYCAMNIAS, E., November 2000.
http://www.kameleonstudios.com/lawlibrary/Doc47.pdf.

[124] DEMCHENKO, Y. "White Collar" Attacks on Web Services and Grids. Draft Version 0.3.

http://staff.science.uva.nl/ demch/analytic/draft-grid-security-incident-03.pdf, March 2005.

[125] DEMCHENKO, Y., GOMMANS, L., DE LAAT, C., AND OUDENAARDE, B. Web Services and Grid Security Vulnerabilities and Threats Analysis and Model. In *Proceedings of the Sixth IEEE/ACM International Workshop on Grid Computing, Seattle, Washington, USA* (November 2005), pp. 262–267.

[126] DEUTSCH, M. Cooperation and Trust: Some Theoretical Notes. In *Nebraska Symposium on Motivation. Nebraska University Press* (1962), M. R. JONES, Ed., pp. 275–319.

[127] DEUTSCH, M. *The Resolution of Conflict: Constructive and Destructive Processes.* New Haven and London: Yale University Press, 1973.

[128] DING, L., KOLARI, P., GANJUGUNTE, S., F. T., AND JOSHI, A. Modeling and Evaluating Trust Network Inference. In *Proceedings Seventh International Workshop on Trust in Agent Societies at AAMAS* (2004).

[129] DING, L., ZHOU, L., AND FININ, T. Trust Based Knowledge Outsourcing for Semantic Web Agents. In *Proceeding of IEEE/WIC International Conference on Web Intelligence, Halifax, Canada* (2003), pp. 379–387.

[130] DIX, J., NANNI, M., AND SUBRAHMANIAN, V. S. Probabilistic Agent Programs. *ACM Transactions on Computational Logic 1, Issue2* (October 2000), 208–246.

[131] DOERNEMANN, T., HEINZL, S., DOERNEMANN, K., MATHES, M., SMITH, M., AND FREISLEBEN, B. Secure Grid Service Engineering for Industrial Optimization. In *Proceedings of the Seventh International Conference on Optimization: Techniques and Applications (ICOTA7)* (2007).

[132] DONEY, P., AND CANNON, J. An Examination of the Nature of Trust in Buyer-Seller Relationships. *Journal of Marketing 61* (1997), 35–51.

[133] DYSON, J., GRIFFITHS, N., LIM CHOI KEUNG, H., JARVIS, S., AND NUDD, G. Trusting Agents for Grid Computing. In *Proceedings of the IEEE International Conference on Systems, Man and Cybernetics, The Hague, The Netherlands* (2004), pp. 3187– 3192.

[134] EWERTH, R., FRIESE, T., GRUBE, M., AND FREISLEBEN, B. Grid Services for Distributed Video Cut Detection. In *Proceedings of the Sixth IEEE International Symposium on Multimedia Software Engineering, Miami, USA, IEEE* (2004), pp. 164–168.

[135] EWERTH, R., GLLAVATA, J., GOLLNICK, M., MANSOURI, F., PAPALILO, E., SENNERT, R., WAGNER, J., FREISLEBEN, B., AND GRAUER, M. Methoden und Werkzeuge zur rechnergesttzten medienwissenschaftlichen Analyse. Siegener Periodicum zur Internationalen Empirischen Literaturwissenschaft., 20, H. 2 2003. 306-320.

[136] FALCONE, R., AND CASTELFRANCHI, C. Trust Dynamics: How Trust is Influenced by Direct Experiences and by Trust Itself. In *Proceedings of the Third International Joint Conference on Autonomous Agents and Multiagent Systems, Washington, DC, USA* (2004), pp. 740–747.

[137] FERREIRA, L., LUCCHESE, F., YASUDA, T., YAU LEE, C., QUEIROZ, C., MINETTO, E., AND MUNGIOLI, A. Grid Computing in Research and Education., April 2005.
http://www.redbooks.ibm.com/redbooks/pdfs/sg246649.pdf.

[138] FORD, W., SOLO, D., HOUSLEY, R., AND POLK, W. IETF RFC 3280:Internet X.509 Public Key Infrastructure., April 2002.
ftp://ftp.rfc-editor.org/in-notes/rfc3280.txt.

[139] FOSTER, I. What is the Grid? A Three Point Checklist. *Grid Today - Daily News and Information for the Global Grid Community 1*, 6 (July 2002).

[140] FOSTER, I., AND KESSELMAN, C. *The Grid: Blueprint for a New Computing Infrastructure.*, second ed. Morgan Kaufmann, 2003.

[141] FOSTER, I., KESSELMAN, C., NICK, J., AND TUECKE, S. The Physiology of the Grid: An Open Grid Services Architecture for Distributed Systems Integration. Open Grid Service Infrastructure WG, Global Grid Forum, 2002.

[142] FOSTER, I., KESSELMAN, C., AND TUECKE, S. The Anatomy of the Grid: Enabling Scalable Virtual Organizations. *International Journal of Supercomputer Applications* (2001).

[143] FOSTER, I., KISHIMOTO, H., SAVVA, A., BERRY, D., DJAOUT, A., GRIMSHAW, A., HORN, B., MACIEL, F., SIEBENLIST, F., SUBRAMANIAM, R., TREADWELLL, J., AND VON REICH, J. The Open Grid Services Architecture., 2005.
http://www.gridforum.org/documents/GWD-I-E/GFD-I.030.pdf.

[144] FRIEDMAN, B., PETER, H., KHAN, J., AND HOWE, D. Trust Online. *Communications of the ACM 43, Issue 12* (2000), 34–40.

[145] FRIESE, T. *Service-Oriented Ad Hoc Grid Computing.* PhD thesis, University of Marburg, 2006.

[146] FRIESE, T., SMITH, M., AND FREISLEBEN, B. Hot Service Deployment in an Ad Hoc Grid Environment. In *Proceedings of the Second Int. Conference on Service-Oriented Computing (ICSOC'04), New York, USA* (2004), ACM Press, pp. 75–83.

[147] GAMBETTA, D. Can we Trust Trust? In *Trust: Making and Breaking Cooperative Relations, edited by GAMBETTA, D.* 1990, pp. 213–237.
http://www.sociology.ox.ac.uk/papers/trustbook.html.

[148] GERMANO, C. Walking the Web of Trust. In *Proceedings Ninth IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises* (2000), pp. 153–158.
http://www.olymp.org/_caronni/work/papers/wetice-web-final.pdf.

[149] GOLBECK, J. A Definition of Trust for Computing with Social Networks. Tech. rep., MINDSWAP, University of Maryland, College Park,, February 2005.
www.mindswap.org/papers/TrustDef.doc.

[150] GOLBECK, J. *Computing and Applying Trust in Web-Based Social Networks.* PhD thesis, University of Maryland, College Park., 2005.
http://trust.mindswap.org/papers/GolbeckDissertation.pdf.

[151] GOLBECK, J., PARSIA, B., AND HENDLER, J. Trust Networks on the Semantic Web. In *Proceedings of Cooperative Intelligent Agents, Helsinki, Finland* (2003), pp. 238–249.

[152] GOLLBECK, J., AND HENDLER, J. Accuracy of Metrics for Inferring Trust and Reputation in Semantic Web-Based Social Networks. In *Proceedings of the Fourteenth International Conference on Knowledge Engineering and Knowledge Management, Whittlebury Hall, UK* (2004), pp. 116–131.

[153] GOLLBECK, J., AND HENDLER, J. Inferring Reputation on the Semantic Web. In *Proceedings of the Thirteenth International World Wide Web Conference, New York, USA* (2004).

[154] GRANDISON, T., AND SLOMAN, M. A Survey of Trust in Internet Applications. *IEEE Communications Surveys & Tutorials 3*, 4 (2000).

[155] GRID COMPUTING TEAM. Grid Computing: Solution Briefs., 2005.
http://www.redbooks.ibm.com/redpapers/pdfs/redp3891.pdf.

[156] HALBERSTADT, A., MUI, L., AND MOHTASHEMI, M. A Computational Model of Trust and Reputation. In *Proceedings of the Thirtyfifth Hawai'i International Conference on System Science, Hawaii, USA* (2002).
http://www.cdm.lcs.mit.edu/people/lmui/docs/TrustReputationModel.ps.

[157] HARDIN, R. Trust & Trustworthiness. Russell Sage Foundation, New York, USA., 2002.

[158] HICKMAN, K. The SSL Protocol (version 2), Netscape Communications Corporation., February 1995.

[159] HUYNH, T. D., JENNINGS, N. R., AND SHADBOLT, N. R. FIRE: An Integrated Trust and Reputation Model for Open Multi-Agent Systems. In *Proceedings of the Sixteenth European Conference on Artificial Intelligence, Valencia, Spain* (2004), pp. 18–22.

[160] HUYNH, T. D., JENNINGS, N. R., AND SHADBOLT, N. R. An Integrated Trust and Reputation Model for Open Multi-Agent Systems. *Journal of Autonomous Agents and Multi-Agent Systems 13, Issue 2* (2006), 119–154.

[161] HWANG, K., AND TANACHAIWIWAT, S. Trust Models and NetShield Architecture for Securing Grid Computing. *Journal of Grid Computing* (2003).

[162] IBM. Access Control Meets Public Key Infrastructure, or: Assigning Roles to Strangers. IEEE Symposium on Security and Privacy., 2000. http://www.hrl.il.ibm.com/TrustEstablishment/paper.asp.

[163] ISMAIL, R., AND JOSANG, A. The Beta Reputation System. In *Proceedings of the Fifteenth Bled Conference on Electronic Commerce* (2002).

[164] JONKER, C. M., SCHALKEN, J. J., THEEUWES, J., AND TREUR, J. Human Experiments in Trust Dynamics. In *Proceedings of Trust Management, Second International Conference, iTrust 2004, Oxford, UK* (2004), pp. 206–220.

[165] JOSANG, A. The Right Type of Trust for Distributed Systems. In *Proceedings of the 1996 workshop on New security paradigms, Lake Arrowhead, California, USA* (September 1996), pp. 119–131.

[166] JOSANG, A. A Model for Trust in Security Systems. In *Second Nordic Workshop on Secure Computer Systems* (1997).

[167] JOSANG, A. Artificial Reasoning with Subjective Logic. In *Proceedings of the Second Australian Workshop on Commonsense Reasoning, Perth, Australia* (1997).

[168] JOSANG, A. Prospectives for Modeling Trust in Information Security. In *Australian Conference on Information Security and Privacy, Sydney, NSW, Australia* (1997), pp. 2–13.

[169] JOSANG, A. A Subjective Metric of Authentication. In *European Symposium on Research in Computer Science, Louvain-la-Neuve, Belgium* (1998), pp. 329–344.

[170] JOSANG, A. Reliability Analysis with Uncertain Probabilities. In *Fourth International Conference on Probabilistic Safety Assessment and Management, New York, USA* (1998).

[171] JOSANG, A. An Algebra for Assessing Trust in Certification Chains. In *Network and Distributed Systems Security* (1999).

[172] JOSANG, A., ISMAIL, R., AND BOYD, C. A Survey of Trust and Reputation Systems for Online Service Provision. *Decision Support Systems* (2005).

[173] JOSANG, A., AND KNAPSKOG, S. A Metric for Trusted systems. In *Twenty First National Security Conference, Budapest, Hungary* (1998), pp. 541–545.

[174] JOSANG, A., AND TRAN, N. Trust Management for E-Commerce., 2000. http://citeseer.nj.nec.com/375908.html.

[175] JUST, M., AND VAN OORSCHOT, P. Addressing the Problem of Undetected Signature Key Compromise. In *Proceedings of the Network and Distributed System Security Symposium,San Diego, California, USA* (1999).

[176] KAGAL, L., FININ, T., AND ANUPAM, J. Trust-Based Security in Pervasive Computing Environments. *IEEE Computer 34, Issue 12* (2001), 154–157. http://www.cs.umbc.edu/_lkagal1/papers/computer-article.pdf.

[177] KAMVAR, S., SCHLOSSER, M., AND GARCIA-MOLINA, H. The EigenTrust Algorithm for Reputation Management in P2P Networks. In *Proceedings of the Twelfth International Conference on World Wide Web, Budapest, Hungary* (2003), pp. 640–651.

[178] KAPLAN, R. A Matter of Trust. In *Information Security Management Handbook.*, T. . KRAUSE, Ed. Auerbach Publishers. Boca Raton, Florida., 2004, pp. 727–740.

[179] KOHL, J., AND NEUMAN, B. C. The Kerberos Network Authentication Service. IETF RFC 1510., 1993.

[180] LAMSAL, P. Understanding Trust and Security. http://www.cs.helsinki.fi/u/lamsal/asgn/trust/UnderstandingTrustAndSecurity.pdf, October 2001.

[181] LENSTRA, A., AND YACOBI, Y. User Impersonation in Key Certification Schemes. *Journal of Cryptology 6, Issue 4* (1993), 225 – 232.

[182] LIM, H., AND ROBSHAW, M. On Identity-Based Cryptography and Grid Computing. In *Proceedings of the Fourth International Conference in Computational Science (ICCS), Krakow, Poland* (2004), pp. 474 – 477.

[183] LIM, H., AND ROBSHAW, M. A Dynamic Key Infrastructure for Grid. In *Proceedings of the European Grid Conference (EGC), Amsterdam, The Netherlands* (2005), pp. 255–264.

[184] LIN, C., VARADHARAJAN, V., WANG, Y., AND PRUTHI, V. Enhancing Grid Security with Trust Management. In *Proceedings of the IEEE International Conference on Services Computing, Shanghai, China* (2004), pp. 303–310.

[185] LINDSTROM, P. Attacking and Defending Web Services., January 2004. http://forumsystems.com/papers/AttackingandDefendingWS.pdf.

[186] LINN, J., BOEYEN, S., ELLISON, G., KARHULUOMA, N., MACGREGOR, W., MADSEN, P., SENGODAN, S.AND SHINKAR, S., AND THOMPSON, P. Liberty Trust Models Guidelines. http://www.oasis-open.org/committees/download.php/6158/sstc-saml-trustmodels-2.0-draft-01.pdf, July 2003.

[187] LOCK, R., AND SOMMERVILLE, I. Grid Security and its Use of X.509 Certificates. http://www.comp.lancs.ac.uk/computing/research/cseg/projects/dirc/papers/gridpaper.pdf.

[188] LUHMANN, N. *Trust and Power.* Chichester: John Wiley, 1979.

[189] LUHMANN, N. Familiarity, Confidence, Trust: Problems and Alternatives. In *Trust: Making and Breaking Cooperative Relations, edited by GAMBETTA, D.* 1990, pp. 94–107. http://www.sociology.ox.ac.uk/papers/trustbook.html.

[190] MA, T., CHEN, L., WANG, C., AND LAU, C. G-PASS Security Infrastructure for Grid Travelers. In *Proceeding of Third International Conference on Grid and Cooperative Computing, Wuhan China* (2004), pp. 301–308.

[191] MARSH, S. *Formalizing Trust as a Computational Concept.* PhD thesis, University of Stirling, 1994.

[192] MASSA, P., AND AVESANI, P. Controversial Users Demand Local Trust Metrics: An Experimental Study on Epinions.com Community. In *Proceedings of the Twentyfifth American Association for Artificial Intelligence Conference, Pittsburgh, Pennsylvania, USA* (2005), pp. 121–126.

[193] MAXIMILIEN, E., AND SINGH, M. A Framework and Ontology for Dynamic Web Services Selection. *IEEE Internet Computing 8, Issue 5* (2004), 84–93.

[194] MAXIMILIEN, E., AND SINGH, M. Toward Autonomic Web Services Trust and Selection. In *Proceedings of Second International Conference on Service Oriented Computing (ICSOC 2004), New York, USA* (2004), pp. 212–221.

[195] MAXIMILIEN, E., AND SINGH, M. Agent-Based Trust Model Involving Multiple Qualities. In *Proceedings of the Fourth International Joint Conference on Autonomous Agents and Multiagent Systems, Utrecht, The Netherlands* (2005), pp. 519–526.

[196] MAYER, R., DAVIS, J., AND SCHOORMAN, D. An Integrative Model of Organizational Trust. *Academy of Management Journal 20, Issue 3* (1995), 709–734.

[197] MCKNIGHT, D. H., AND CHERVANY, N. The Meanings of Trust. *Trust in Cyber-Societies - LNAI* (2001), 27–54.

[198] MENEZES, A., VAN OORSCHOT, P., AND YANSTONE, S. *Handbook of Applied Cryptography.*, fifth ed. 2001.
http://www.cacr.math.uwaterloo.ca/hac/.

[199] MILGRAM, S. The Small-World Problem. *Psychology Today* (1967), 60–67.

[200] MILOJICIC, D., KALOGERAKI, V., AND LUKOSE, R. Peer-to-Peer Computing. Technical report: Hpl-2002-57, 2002.
http://www.hpl.hp.com/techreports/2002/HPL-2002-57R1.pdf.

[201] MITCHELL, R. W. A Framework for Discussing Deception. *Deception, Perspectives on Human and Nonhuman Deceit, Edited by MITCHELL, R. W., and THOMPSON, N. S., Albany, N.Y., State University of New York Press* (1986), 3–40.

[202] MITTAG, H., AND RINNE, H. *Statistical Methods of Quality Assurance.* Chapman & Hall/CRC, 1993.

[203] MOELLERING, G. The Trust/Control Duality: An Integrative Perspective on Positive Expectations of Others. *International Sociology 20, Issue 3* (2005), 283–305.
http://www.mpifg.de/people/gm/downloads/Mollering_TrustControlDuality_IS_20(3)_283-305.pdf.

[204] MUI, L., MOHTASHEMI, M., AND HALBERSTADT, A. A Computational Model of Trust and Reputation. In *Proceedings of the Thirtyfifth Annual Hawaii International Conference on System Sciences, Hawaii, USA* (2002).

[205] NAQVI, S. *Architecture de Scurit pour les Grands Systmes, Ouverts, Rpartis et Htrognes.* PhD thesis, INFRES, Tlcom Paris, 2005.
http://pastel.paristech.org/bib/archive/00001575/01/Theseore Naqvi.pdf.

[206] NAVQI, S., AND RIGUIDEL, M. Threat Model for Grid Security Services. In *Proceedings of European Grid Conference, Amsterdam, The Netherlands* (2005), pp. 1048–1055.

[207] NEGM, W. Anatomy of a Web Services Attack: A Guide to Threats and Preventative Countermeasures., March 2004.
http://whitepapers.itsj.com/detail/RES/$1084293354_2$94.*html*.

[208] NEGM, W. Bringing Balance to Web Services., February 2004.
http://www.forumsystems.com/papers/04BringingBalancesSecurity.pdf.

[209] NETWORK ASSOCIATES INC. An Introduction to Cryptography.
http://www.fi.pgpi.org/doc/pgpintro/.

[210] NEUMAN, C., YU, T., HARTMAN, S., AND RAEBURN, K. The Kerberos Network Authentication Service (V5).
http://www.ietf.org/rfc/rfc4120.txt.

[211] OZIER, W. Risk Analysis and Assessment. In *Information Security Management Handbook. Fifth Edition.*, T. . KRAUSE, Ed., fifth ed. Auerbach Publishers. Boca Raton, Florida., 2004, pp. 795–820.

[212] PAPALILO, E., AND FREISLEBEN, B. Towards a Flexible Trust Model for Grid Environments. In *Proceedings of the First International Conference on Grid Services Engineering and Management (GSEM), Erfurt, Germany.* (2004), LNCS 3270, Springer-Verlag, pp. 94–106.

[213] PAPALILO, E., AND FREISLEBEN, B. Combining Incomparable Public Session Keys and Certificateless Public Key Cryptography for Securing the Communication Between Grid Participants. In *Proceedings of International Conference on Grid Computing, High-Performance and Distributed Applications (GADA'07), Vilamoura, Algarve, Portugal* (2007), R. Meersman and Z. T. et al. (Eds.): OTM 2007, Eds., LNCS 4804, Springer-Verlag, pp. 1264 – 1279.

[214] PAPALILO, E., AND FREISLEBEN, B. Managing Behaviour Trust in Grids Using Statistical Methods of Quality Assurance. In *Proceedings of the Third International Symposium on Information Assurance and Security (IAS07), Manchester, United Kingdom* (2007), IEEE Computer Society Press, pp. 319 – 324. (Best paper award).

[215] PAPALILO, E., AND FREISLEBEN, B. Managing Behaviour Trust in Grid Computing Environments. *International Journal of Information Assurance and Security (JIAS)* (2008).

[216] PAPALILO, E., FRIESE, T., SMITH, M., AND FREISLEBEN, B. Trust Shaping: Adapting Trust Establishment and Management to Application Requirements in a Service-Oriented Grid Environment. In *Proceedings of the Fourth International Conference on Grid and Cooperative Computing (GCC), Beijing, China.* (2005), LNCS 3795, Springer-Verlag, pp. 47–58.

[217] PATEL, J., TEACY, W., JENNINGS, N., AND LUCK, M. A Probabilistic Trust Model for Handling Inaccurate Reputation Sources. In *Proceedings of Third International Conference on Trust Management, Rocquencourt, France* (2005), pp. 193–209.

[218] PATIL, V., AND SHYAMASUNDAR, R. Trust Management for E-transactions. In *Sadhana: Academy Proceedings in Engineering Sciences* (2005), vol. 30, Indian Academy of Sciences, pp. 141–158.
http://www.ias.ac.in/sadhana/Pdf2005AprJun/Pe1296.pdf.

[219] PEARLMAN, L., WELCH, V., FOSTER, I., KESSELMAN, C., AND TUECKE, S. A Community Authorization Service for Group Collaboration. In *Proceedings of the Third IEEE International Workshop on Policies for Distributed Systems and Networks* (2002), p. 50.

[220] PETERS, R., COVELLO, V., AND MCCALLUM, D. The Determinants of Trust and Credibility in Environmental Risk Communication: An Empirical Study. *Risk Analysis 17, Issue 1* (1997), 43–54.
http://www.centerforriskcommunication.com/pubs/crc2tables.pdf.

[221] PETRIE, C., AND WIGGINS, M. Pattie Maes on Software Agents: Humanizing the Global Computer. IEEE Internet Computing., July 1997.

[222] QUETIER, B., AND CAPPELLO, F. A Survey of Grid Research Tools: Simulators, Emulators and Real Life Platforms. In *Proceedings of the Seventeenth IMACS World Congress on Scientific Computation, Applied Mathematics and Simulation, Paris France* (2005).

[223] RAMCHURN, S. D., SIERRA, C., GODO, L., AND JENNINGS, N. A Computational Trust Model for Multi-Agent Interactions Based on Confidence and Reputation. In *Proceedings of the Sixth International Workshop of Deception, Fraud and Trust in Agent Societies, Melbourne, Australia* (2003), pp. 69–75.

[224] RAN, S. A Model for Web Services Discovery With QoS. *ACM SIGecom Exchanges 4, Issue 1* (March 2003), 1–10.

[225] REGAN, K. A Social Reputation Model for Electronic Marketplaces Sensitive to Subjectivity, Deception and Change. Master's thesis, University of Waterloo, Ontario, Canada, 2006.

[226] RESNICK, P., KUWABARA, K., ZECKHAUSER, R., AND FRIEDMAN, E. Reputation Systems. *Communications of the ACM 43, Issue 12* (2000), 45–48.

[227] RUTKOWSKA, J. Subverting Vista Kernel.
http://blackhat.com/html/bh-usa-06/bh-us-06-speakers.html
http://www.invisiblethings.org/papers/Joanna rutkowska - subverting vista kernel.ppt.

[228] SABATER, J., AND SIERRA, C. REGRET: A Reputation Model for Gregarious Societies. In *Proceedings of the Fourth Workshop on Deception, Fraud and Trust in Agent Societies, Montreal, Canada* (2001), pp. 61–69.

[229] SABATER, J., AND SIERRA, C. Reputation and Social Network Analysis in Multi-Agent Systems. In *Proceedings of the First International Joint Conference on Autonomous Agents and Multiagent Systems, Bologna, Italy* (2002), pp. 475–482.

[230] SAXENA, A., AND SOH, B. Pairing-Based Cryptography for Distributed and Grid Computing. In *Proceeding of the IEEE International Conference on Communications (ICC), Istanbul, Turkey* (2006), pp. 2335–2339.

[231] SCHILLO, M., FUNK, P., AND ROVATSOS, M. Using Trust for Detecting Deceitful Agents in Artificial Societies. *Applied Artificial Intelligence 14, Issue 8* (2000), 825–848.

[232] SCHOPF, J. Grids: The Top Ten Questions., March 2003. www.globus.org/alliance/publications/papers/topten.final.pdf.

[233] SCHRECK, G., SCHADLER, T., RUTSTEIN, C., AND TSENG, A. The Fabric Operating System. Techstrategy report, September 2003.

[234] SCHRIDDE, C., PICHT, H., HEIDT, M., SMITH, M., AND FREISLEBEN, B. Secure Integration of Desktop Grids and Compute Clusters Based on Virtualization and Meta-Scheduling. In *Proceedings of the Germane e-Science Conference* (2007).

[235] SEN, S., AND SAJJA, N. Robustness of Reputation-based Trust: Booblean Case. In *Proceedings of the First International Joint Conference on Autonomous Agents and Multiagent Systems, Bologna, Italy* (2002), pp. 288–293.

[236] SHAMIR, A. Identity-Based Cryptosystems and Signature Schemes. In *Proceedings of CRYPTO* (1984), pp. 47–53.

[237] SHEN, Z., WU, X., WANG, Y., PENG, W., AND ZHANG, H. Group Key Management in Grid Environment. In *Proceedings of the First International Multi-Symposium on Computer and Computational Sciences (IMSCCS), Hangzhou, Zhejiang, China* (2006), pp. 626–631.

[238] SINGER, D., MILLER, J., AND RESNICK, P. Rating Services and Rating Systems. Tech. rep., REC-PICS-services-961031, 1996. http://www.w3.org/TR/REC-PICS-services.

[239] SKOW, D. A Grid is an Automated Error Amplifier., June 2004.

[240] SLOMAN, M. Trust Management in Internet and Pervasive Systems. *IEEE Intelligent Systems 19, Issue 5* (2004), 77–79.

[241] SMITH, M., FRIESE, T., ENGEL, M., AND FREISLEBEN, B. Countering Security Threats in Service-Oriented On-Demand Grid Computing Using Sandboxing and Trusted Computing Techniques. *Journal of Parallel and Distributed Computing 66*, 9 (2006), 1189–1204.

[242] SMITH, M., FRIESE, T., ENGEL, M., FREISLEBEN, B., KOENIG, G., AND YURCIK, W. Security Issues in On-Demand Grid and Cluster Computing. In *Sixth IEEE International Symposium on Cluster Computing and the Grid Workshops (CCGRIDW'06)* (2006), IEEE Press, p. 24.

[243] SMITH, M., FRIESE, T., AND FREISLEBEN, B. Towards a Service-Oriented Ad Hoc Grid. In *Proceedings of the Third International Symposium on Parallel and Distributed Computing, Cork, Ireland* (2004), pp. 201–208.

[244] SMITH, M., FRIESE, T., AND FREISLEBEN, B. Intra-Engine Service Security for Grids Based on WSRF. In *Proceedings of the 2005 IEEE/ACM International Sysmposium on Cluster Computing and Grid (CCGRID'05), Cardiff, UK* (2005), IEEE Press, pp. 644–653.

[245] SONNENREICH, W., AND ALBANESE, J. *Network Security Illustrated.* McGraw-Hill, 2003.

[246] SRINIVASAN, L., AND TREADWELL, J. An Overview of Service-oriented Architecture, Web Services and Grid Computing. *HP Software Global Business Unit* (November 2005).
http://h71028.www7.hp.com/ERC/downloads/SOA-Grid-HP-WhitePaper.pdf.

[247] STALLINGS, W. *Cryptography and Network Security.*, fourth ed. 2006. ISBN-10: 0131873164.

[248] STEWART, K. J. Transference as a Means of Building Trust in World Wide Web Sites. In *Proceeding of the Twentieth international Conference on Information Systems, Atlanta, GA, USA* (1999), pp. 459–464.

[249] STEWART, K. J., AND ZHANG, Y. Effects of Hypertext Links on Trust Transfer. In *Proceedings of the Fifth International Conference on Electronic Commerce, New York, USA* (2003), pp. 235–239.

[250] TEACY, W., PATEL, J., JENNINGS, N., AND LUCK, M. Coping with Inaccurate Reputation Sources: Experimental Analysis of a Probabilistic Trust Model. In *Proceedings of the Fourth International Joint Conference on Autonomous Agents and Multiagent Systems, Utrecht, The Netherlands* (2005), pp. 997–1004.

[251] TEACY, W., PATEL, J., N.R., J., AND LUCK, M. Travos: Trust and Reputation in the Context of Inaccurate Information Sources. *Autonomous Agents and Multi-Agent Systems 12, Issue 2* (2006), 183–198.

[252] TIE-YAN, L., HUAFEI, Z., AND KWOK-YAN, L. A Novel Two-Level Trust Model for Grid. In *Proceedings of Fifth International Conference on Information and Communications Security, Huhehaote, China* (2003), pp. 214–225.

[253] TSUGAWA, M., AND FORTES, J. A Virtual Network (ViNe) Architecture for Grid Computing. In *Proceedings of Twentieth International Parallel and Distributed Processing Symposium (IPDPS), Rhodes Island, Greece* (2006), p. 10.

[254] VENDATASUBRAMANIAN, N., AND NAHRSTEDT, K. An Integrated Metric for Video QoS. In *Proceedings of the Fifth ACM International Multimedia Conference, Seattle, USA* (1997), pp. 371–380.

[255] WAGUIH, H. A Proposed Trust Model for the Semantic Web. *Transactions on Engineering, Computing and Technology V11 11* (2006), 39–44. http://www.enformatika.org/data/v11/v11-8.pdf.

[256] WANG, Y., AND VASSILEVA, J. Bayesian Network-Based Trust Model. In *International Conference on Web Intelligence, Halifax, Canada* (2003), pp. 372–378.

[257] WANG, Y., AND VASSILEVA, J. Trust and Reputation Model in Peer-to-Peer Networks. In *Peer-to-Peer Computing, Linkping, Sweden* (2003), pp. 150–.

[258] WATERS, B., FELTEN, E., AND SAHAI, A. Receiver Anonymity via Incomparable Public Keys. In *(CCS), Washington, D.C., USA* (2003), pp. 112–121.

[259] WATTS, D., AND STROGATZ, S. Collective Dynamics of "Small-World" Networks. *Nature 393* (1998), 440–442.

[260] WINSLETT, M., YU, T., SEAMONS, K. E., HESS, A., JACOBSON, J., JARVIS, R., SMITH, B., AND YU, L. Negotiating Trust on the Web. *IEEE Internet Computing, 6, Issue 6* (2002), 30–37.

[261] XIONG, L., AND LIU, L. A Reputation-Based Trust Model for Peer-to-Peer Ecommerce Communities. In *Proceedings of IEEE Conference on e-Commerce, New York, USA* (2003), pp. 275–284.

[262] XIONG, L., AND LIU, L. PeerTrust: Supporting Reputation-Based Trust for Peer-to-Peer Electronic Communities. *IEEE Transactions on Knowledge and Data Engineering 16, Issue 7* (July 2004), 843–857.

[263] YU, B., AND SINGH, M. A Social Mechanism of Reputation Management in Electronic Communities. In *Proceedings of the Fourth International Workshop on Cooperative Information Agents: The Future of Information Agents in Cyberspace, London, UK* (2000), pp. 154–165.

[264] YU, B., AND SINGH, M. An Evidential Model of Distributed Reputation Management. In *Proceedings of the First International Joint Conference on Autonomous Agents and Multiagent Systems, New York, USA* (2002), pp. 294–301.

[265] YU, B., AND SINGH, M. Detecting Deception in Reputation Management. In *Proceedings of the Second International Joint Conference on Autonomous Agents and Multi-Agent Systems, New York, USA* (2003), pp. 73–80.

[266] YU, T., AND WINSLETT, M. Policy Migration for Sensitive Credentials in Trust Negotiation. In *Proceedings of the ACM Workshop on Privacy in the Electronic Society, Washington D.C., USA* (2003), pp. 9–20.

[267] YU, T., WINSLETT, M., AND SEAMONS, K. Interoperable Strategies in Automated Trust Negotiation. In *ACM Conference on Computer and Communications Security, Philadelphia, Pennsylvania, USA* (2001), pp. 146–155.

[268] ZACHARIA, G. Collaborative Reputation Mechanisms for Online Communities. Master's thesis, Massachusetts Institute of Technology, 1999.

[269] ZENG, K., AND FUJITA, T. Methods, Devices and Systems for Generating Anonymous Public Keys in a Secure Communication System. Patent No. 20060098819, May 2006.
http://www.freepatentsonline.com/20060098819.html.

[270] ZHAO, S., LO, V., AND GAUTHIERDICKEY, C. Result Verification and Trust-Based Scheduling in Peer-to-Peer Grids. In *Proceedings of the Fifth IEEE International Conference on Peer-to-Peer Computing, Konstanz, Germany* (2005), pp. 31–38.

[271] ZHENG, J., VEINOTT, E., BOS, N., OLSON, J. S., AND OLSON, G. M. Trust Without Touch: Jumpstarting Long-Distance Trust with Initial Social Activities. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, New York, USA.* (2002), pp. 141–146.

Lebenslauf

10.05.1975	Geboren in Tirana, Albanien.
1982 – 1993	Schulausbildung an der Grundschule "Mihal Grameno" (Tirana) und an dem Gymnasium "Ismail Qemali" (Tirana). Abschluß: Abitur
1993 – 1998	Studium an der Polytechnischen Universität Tirana, Fakultät für Elektrotechnik. Abschluß: Diplom-Ingenieur
1998 – 2004	Wissenschaftlicher Mitarbeiter an der Polytechnischen Universität Tirana, Fakultät für Elektrotechnik.
2000 – 2002	Wissenschaftlicher Mitarbeiter an der Universität Siegen, Fachbereich Elektrotechnik und Informatik. Teilnahme an das Projekt "Stability Pact for Southeastern Europe", finanziert durch Deutscher Akademischer Austauschdienst (DAAD).
2002 – 2007	Wissenschaftlicher Mitarbeiter an der Philipps-Universität Marburg, Fachbereich Mathematik und Informatik.
2007 – *heute*	IT Berater bei Altran CIS, Frankfurt am Main.