Self-Configuration of Grid Nodes Using a Policy-Based Management Architecture

Félix J. García[1], Óscar Cánovas[2], Gregorio Martínez[1], and Antonio F. Gómez-Skarmeta[1]

[1] Department of Information and Communications Engineering
[2] Department of Computer Engineering
University of Murcia, 30071, Murcia, Spain
{fgarcia,gregorio,skarmeta}@dif.um.es, ocanovas@ditec.um.es

Abstract. During the past years, Grid Computing has emerged as an important research field concerning to large-scale resource sharing. Several research and academic institutions have widely adopted this technology as a mechanism able to integrate services across distributed, heterogeneous, and dynamic virtual organizations. However, this rapid expansion is exposing the need to provide effective means to manage virtual organizations, especially those questions related to the insertion and maintenance of grid nodes. This paper proposes a flexible and automated method based on policies to cope with these requirements. Policies constitute a mechanism for specifying the behaviour a grid node must exhibit. In this work we present a policy-based management architecture, which makes use of COPS-PR to exchange information about specific parameters related to grid nodes based on Globus Toolkit 2.4. Moreover, we have also added new classes to the Framework PIB (Policy Information Base) in order to represent configuration data about the different components of this toolkit.

1 Introduction and Rationale

Increasingly, computing systems are becoming more decentralized and distributed. In addition, companies are realizing that they can save costs by outsourcing some elements of their computing systems. This continuous decentralization makes essential the control on how the resources are dynamically assembled. Work within this field has lead to the development of Grid Technologies [1], widely adopted by scientific and academic institutions.

These collections of resources and users from geographically distributed organizations unified by a common goal are called *virtual organizations (VO)* [2]. VOs introduce challenging management and policy issues, resulting from complex relationships between local sites and the goals of the VO with respect to resource allocation or security administration. Since VO resources controlled by grid nodes are located within multiple organizations, maintaining a consistent system-wide configuration among the grid nodes requires a consistent manner to join an existing virtual organization. A VO is defined by the set of elements it is composed of, and also by the different policies governing these elements. Therefore, on the one hand,

M. Bubak et al. (Eds.): ICCS 2004, LNCS 3038, pp. 158–165, 2004.

any potential grid node joining the VO must initially conform to the overall policy, which involves an appropriate distribution of the related policy and its enforcement. On the other hand, this situation can be more complex if we consider the fact that resources may be highly dynamic, which can lead to the definition of policies that can vary during time. Changes on the VO policy must be notified to the existing grid nodes in order to adapt their behaviour to the current policy. Despite its importance, relatively little system-oriented research has addressed this topic. An exception that has encouraged our work is [3], although it follows a different approach based on active networks.

Thus, the management of the overall Grid system in an automated and flexible manner is becoming more and more important. Manual configuration of systems is an arduous task and error-prone. The concept of policy-based management (PBM) addresses some of these issues and offers possible solutions. Policies allow the administrators of a Grid node to specify the behaviour they want it to exhibit, through a set of selections made from a range of parameters related to the nodes. Moreover, policies can be dynamically changed according to the evolution of the system or management strategy (for example, with the insertion of a new trusted third party related to a new organization in the VO, or a new information server). The main goal is to adapt a dynamic VO to different configurations rather than hard-coding a particular one. This can be accomplished defining the different roles that can be played by the grid nodes, and therefore specifying the policy that will be associated to those roles. In next sections we will present several roles for grid nodes depending on the components they include.

Policies are defined using high-level tools and languages, and then are distributed, either directly or through the use of an intermediate repository, to special policy servers called policy decision points (PDPs). Finally, the grid nodes, acting as policy enforcement points (PEP), contact the PDPs in order to obtain an instance of the policy. It is possible that a particular grid node (or set of nodes) has some preconfigured internal policy statements and then a conflict can appear with the VO general policy. For solving this, policies have a level of priority. In the case that a conflict cannot be solved with this procedure, it will be properly reported to the network managers involved in the policy definition.

In this work, regarding the communication protocol between PDPs and PEPs, we have selected the COPS (Common Open Policy Service) protocol [4], and specifically its extension for provisioning policy models, COPS-PR (COPS Usage for Policy Provisioning) [5]. As we will show in next sections, Grid systems can be integrated with COPS by extending a PIB (Policy Information Base), i.e., a data structure containing policy data.

This paper is organized as follows. Section 2 provides an overview of the Globus Toolkit and specifies how the use of policies can be integrated into the management of the different components of the toolkit. Section 3 presents the elements and protocols of the policy-based management architecture for Grid Computing. Section 4 outlines our proposed Grid information model and the PIB extensions that have been designed to exchange data about configuration of grid nodes making use of the Globus Toolkit. Finally, we conclude the paper with our remarks and some future directions derived from this work.

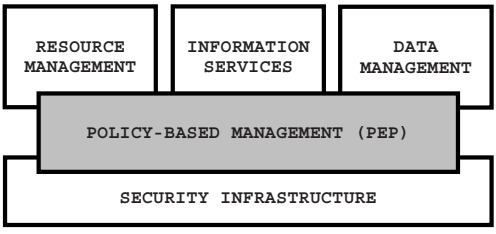

Fig. 1. View of the Globus Toolkit with PBM support

2 PBM Support for the Globus Toolkit

The software toolkit developed within the Globus project comprises a set of components that implement basic services for security, resource location, resource management, data access, communication, and so on. Globus is built as a layered architecture of services, from which developers can select to build custom tailored tools or applications on top of them. These services are distinct and have well defined interfaces, and therefore can be integrated into applications/tools in an incremental fashion. This paper is based on GT 2.4 since, nowadays, it is in use at hundreds of sites and by dozens of major Grid projects worldwide, which can benefit from the integration of policy-based management architecture for self-configuration of their grid nodes. Although OGSA (Open Grid Services Architecture) [6] represents the natural evolution of GT 2.4, and it is being the main standardization effort for Grid technologies, in this paper we focus on GT 2.4 due to its high degree of adoption by the community.

Globus Toolkit components (Figure 1) include the Grid Security Infrastructure (GSI) [7], which provides a single-sign-on, run-anywhere authentication service, with support for delegation of credentials, local control over authorization, and mapping from global to local user identities; the Grid Resource Access and Management (GRAM) [8] protocol and service, which provides remote resource allocation and process creation, monitoring, and management services; the Metacomputing Directory Service (MDS) [9], an extensible Grid information service that provides a uniform framework for discovering and accessing system configuration and status information; and data management services, that is, a high-speed data movement protocol like GridFTP and mechanisms related to dataset replicas.

As it is shown in Figure 1, the policy-based management component overlaps the three pillars of the toolkit and the security infrastructure, establishing their configuration parameters according to the VO general policy. It is layered on top the GSI since it uses its security services, while acting as policy enforcement point, in order to contact to the PBM architecture, as we will introduce in Section 3. This module modifies the configuration parameters either during the join operation to a VO, or as a consequence of a maintenance task in a VO, as for instance the insertion, deletion or modification of other grid nodes. Going into details, some of the configuration values of MDS that might be established during the join operation are network parameters, distinguished names, GIIS servers, valid GRIS reporters, information providers, etc. However, some of these parameters can change over time with the insertion or deletion of GIIS servers, GRIS reporters, job managers, access

rules to the directory information tree, or bind methods. Analogously, GRAM parameters can be initially specified (such as, network parameters, list of supported job managers, nodes acting as resource coordinators) and can be further modified (new resource coordinators, information to be published in the MDS). The same procedure is applied for the data management services and the security infrastructure. Therefore, on the one hand, we need a model able to reflect, in a structured manner, these and other configuration parameters and, on the other hand, a mechanism to distribute and translate that model to a data representation format, as we will see in Sections 3 and 4.

3 A PBM Architecture Providing Self-Configuration

As stated before, policy-based management (PBM) is a term describing a new concept and architecture to control and provision dynamic configurations and behaviours for network nodes and services. In the context of this research it is used for providing self-configuration to grid nodes.

The proposed architecture is composed of five elements as depicted in Figure 2. It was designed following the recommendations defined by the IETF, mainly through the Policy Framework working group, which provides a way of representing and managing policies in a vendor-independent, interoperable, and scalable manner. These features have a key significance in our research.

The administrator interacts, using his policy console (usually a web browser) with one of the policy management tools (PMT) existing in the system. This allows him to create, modify and/or delete policies specifying how the grid nodes belonging to one particular virtual organization need to be configured. In this definition phase, the communication protocol in use is HTTPS (i.e. HTTP + SSL), which provides confidentiality, integrity, and mutual authentication between the policy console and the PMT server. Moreover, the administrator using his private key digitally signs policy documents. This signature is validated by the intermediate nodes (PMTs, PDPs, and PEPs) existing in the PBM architecture.

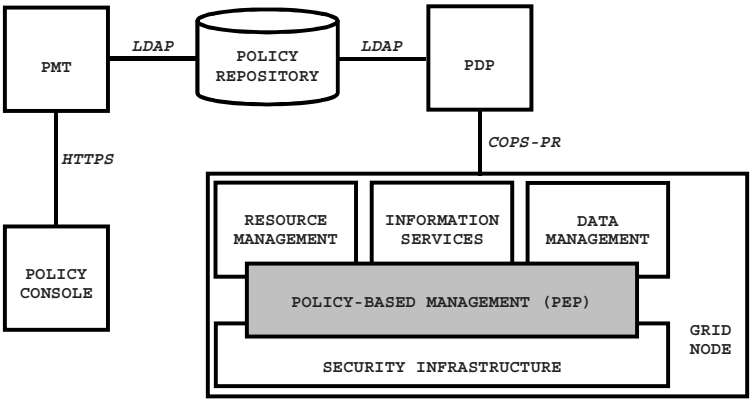

Fig. 2. General Policy-Based Management Architecture

Each PMT is composed of two main components: a policy GUI, which is an editor that can generate or edit policy documents, and a policy validator that validates every policy specification before it is stored in the policy repository. This validation process is done using a policy model, which defines the high-level syntax of every grid configuration parameter being managed using policies. Its definition, presented in next section, is based on the PCIM [10] and PCIMe [11] information models defined by the IETF.

Once the relevant digitally signed policies are stored in the policy repository using LDAP, they are conveniently retrieved by the servers in charge of taking decisions in the management system. They are called policy decision points (PDPs).

Each time one of these servers receives a new policy information (because a new document has been created, or an existing one has been modified or deleted), it has to validate its signature and decide whether this policy needs to be applied to each of the policy enforcement points (PEPs) connected to it. Moreover, when a new PEP is active in the network it needs to get or update the internal configuration it is using. As commented above, in our proposal a PEP is a component existing in every grid node.

In both cases, a communication, preferably secure, needs to be established between the PDP and the PEP. For this, we are making use of COPS-PR as a policy transfer protocol. The main reason behind this decision is that policies in COPS-PR are defined, not per individual interface (as it is in SNMP, for example), but on a per-role basis. This allows us to manage and configure grid nodes as a whole and not as a collection of individual hosts. This is also one of the major reasons of applying the PBM paradigm in this scenario.

The exchange of COPS-PR messages between a PDP server and a PEP (grid node) is secured with a TLS connection from the PEP to the PDP. This secure socket is provided by the GSI component (and its public key related information) existing in every grid node.

4 GRID-VO Policy Configuration Information Model

VO policies that are exchanged between the components of a PBM system may assume a variety of forms as they travel from a PMT to a repository, or from a PDP to a PEP. At each level, policies are represented in a way that is convenient for the current task. This means that there are several translation levels from the abstract policy to the particular PEP (grid node) configuration. Thus, it is convenient to define a configuration information model, which by definition is independent of any particular data storage mechanism and access protocol, and that can be used as a bridge in the translation process between the abstract policy and the particular configuration. The translation process is roughly expressed as follows:

1. The informal VO policy is expressed by a human policy maker (e.g., *'All job information must not be published'*).
2. An administrator analyzes the grid components and determines the existing roles [10]. A role may be assigned to different grid nodes, and each node may play several roles.
3. The VO administrator models the informal policy using the GRID-VO policy configuration information model (GRID-PCIM), thus creating a formal

representation of the abstract policy (e.g. *'If there is GRAM support then it does not permit to publish jobs'*).

4. The VO administrator assigns roles to the policy groups created in the previous step matching the roles of the network elements assigned in step 2.
5. A PDP translates the abstract policy created in step 3 into specific configuration parameters for all nodes affected by the new policy (i.e. grid nodes that have a role matching the policy role).
6. For each PEP (grid node) in the VO, the PDP (or an agent of the PDP) issues the appropriate node-specific instructions needed to enforce the policy.

During the second step, the administrator needs to know the roles that can be assigned to the grid nodes. Although some grid scenarios may present richer roles, we have identified four basic grid roles (which can also be combined). These are the following:

- *Individual-node*. Independent grid node not pertaining to a cluster of workstations (e.g. PCs belonging to individual users).
- *Cluster-node*. A cluster node makes reference to a computing device, which is part of a cluster and is controlled by the configuration policy related to the cluster. Software products for management of clusters (e.g. CONDOR, PBS) sometimes impose specific requirements about the behaviour of a grid node.
- *Coordinator-node*. A coordinator is a role assigned to those nodes acting as resource brokers, co-allocators, or, more generally, nodes which need to have a wider vision about the configuration and features of the VO in order to coordinate several nodes.
- *Index-node*. Nodes containing information about the VO, for monitoring or discovering services (e.g. GIIS servers).

During the third step, the administrator uses GRID-PCIM, which defines an information model for enforcing grid configuration. We have proposed GRID-PCIM based on the IETF Policy Core Information Model (PCIM) [10] an its extensions [11], and all models derive from and uses classes defined in the DMTF Common Information Model (CIM). Figure 3 shows the GRID-VO policy classes representing the set of policies contained in a grid node.

The PolicyGroup class represents the set of policies that are used in a grid node. The VO administrator, in step 4, associates this set of policies to a grid node, which is represented by the class System, depending on the roles. In the model, the class GRIDAction represents GRID actions that are policy-enforced configurations. Classes MDSAction, GRAMAction, DataServicesAction and GSIAction models the actions related to each grid component. The rest of classes are fully specified in [10] and [11].

As commented before, our architecture makes use of COPS-PR in order to exchange policy information between a PDP and a PEP. Both elements uses a general policy structure termed Policy Information Base (PIB). PIB defines a low-level policy representation and is described by a well-defined model of classes, which are referred to Provisioning Classes (PRCs). The instances of these classes are referred to Provisioning Instances (PRIs), and form the PIB. By using PIB in conjunction with the COPS-PR protocol, a PEP can inform to the PDP about its capabilities and roles. On the other hand, the PDP uses the PIB to provide configuration information to the PEP. If a change occurs in the virtual organization (i.e. a new GIIS is added or replaced), configurations are automatically updated from the PDPs to ensure operational consistency without any human intervention.

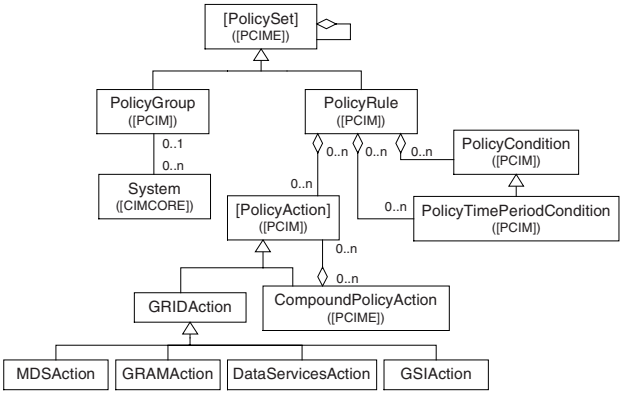

Fig. 3. UML Diagram of GRID Policy Classes

We have defined a set of the provisioning classes (PRC), based on GRID-PCIM, containing parameters for GRAM, MDS, Data Services and GSI. We have formed the following PRC groups for each component:

- *GRAM PIB classes Group*. These PRCs specify the parameters related to resource management: supported job managers, publication of job status, network parameters, etc.
- *MDS PIB classes Group*. These classes specify the configuration parameters related to the MDS: network configuration of GRIS and GIIS servers, valid schemas, access control rules, valid registrants, trusted GIIS, protection of communications between GRIS and GIIS, etc.
- *Data Services PIB classes Group*. Configuration of data management services: parameters related to the GridFTP protocol and replicas of datasets.
- *GSI PIB classes Group*. PRCs specifying the cryptographic options, trusted third parties, authorization services, delegation restrictions, etc.

5 Conclusions and Future Work

This paper contributes to the definition of a policy-based management (PBM) architecture intended to provide a flexible and automated self-configuration method for the grid nodes belonging to a virtual organization. It allows managers to partially delegate management tasks in the form of specific policies.

VO administrators interact with this architecture by defining high-level policies in a Policy Management Tool (PMT). These policies are then directly distributed from the PMTs to special policy servers, called *Policy Decision Points* (PDPs). The PDPs process these policies and take policy decisions. These policies are properly distributed using COPS-PR to the grid nodes, which act as *Policy Enforcement Points* (PEPs).

As a statement of direction, we are currently working on the integration of XML technologies with the PBM architecture and the PIB model presented in this paper. They will be used for self-configuring grid nodes belonging to a Globus 3.0-based virtual organization.

Acknowledgments. This work has been partially funded by the SENECA Piramide project (PB/32/FS/02) and the EU Euro6IX project (IST-2001-32161)

References

1. Foster, I., Kesselman, C. (eds.): The Grid: Blueprint for a New Computing Infrastructure. Morgan Kaufmann (1999)
2. Foster, I.: The Anatomy of the Grid: Enabling Scalable Virtual Organizations. Proceedings of Euro-Par (2001)
3. Galis, A., Gelas, J., Lefèvre, L., Yang, K.: Active Network Approach to Grid Management. Proceedings of International Conference on Computational Science (2003) 1103-1112
4. Durham, D., Boyle, J., Cohen, R., Herzog, S., Rajan, R., Sastry, A.: The COPS (Common Open Policy Service) Protocol. Request For Comments RFC 2748. IETF (2000)
5. Chan, K., Durham, K., Gai, S., Herzog, S., McCloghrie, K., Reichmeyer, F., Seligson, J., Smith, A., Yavatkar, R.: COPS Usage for Policy Provisioning. Request For Comments RFC 3084. IETF (2001)
6. Foster, I., Kesselman, C., Nick, J. M., Tuecke, S.: The Physiology of the Grid. Global Grid Forum (2002)
7. Welch, V., Siebenlist, F., Foster, I., Bresnahan, J., Czajkowski, K.: Security for Grid Services. Proceedings of 12th IEEE Internation Symposium on High Performance Distributed Computing (2003) 48-57
8. Czajkowski, K., Foster, I., Karonis, N., Kesselman, C., Martin, S., Smith, W., Tuecke, S.: A Resource Management Architecture for Metacomputing Systems. Proceedings of IPPS/SPDP(1998), 62-82
9. Czajkowski, K., Fitzgerald, S., Foster, I., Kesselman, C.: Grid information services for distributed resource sharing. Proceedings of 10th Internation Symposium on High Performance Distributed Computing (2001)
10. Moore, B., Ellesson, E., Strassner, J., Westerinen, A.: Policy Core Information Model - Version 1 Specification. Request For Comments RFC 3060. IETF (2001)
11. Moore, B.: Policy Core Information Model (PCIM) Extensions. Request for Comments RFC 3460. IETF (2003)