Global self-management of network and telecommunication information systems and services

Anasser Ag Rhissa, Adil Hassnaoui
GET INT, CNRS Samovar
Institut National des Télécoms, 9 Rue charles Fourier 91011 Evry FRANCE
Anasser.Ag-Rhissa@int-evry.fr, adil.hassnaoui@int-evry.fr

Abstract

The objectives of this paper are to introduce the characteristics of self-management (autonomic management) architectures, their comparison with current architectures, their challenges and present our global self-management architecture based on OGSA (Open Grid Services Architecture) and Peer-to-Peer model. The process of SLA (Service Level Agreement) contract negotiation using semantic ontology in our self-management architecture is explained. The autonomic management architectures of CISCO and IBM are briefly described and compared with our self-management architecture.

Keywords *: Self-management, Autonomic Computing, Autonomic Management, Peer-to-Peer, Global QoS management, Grid computing, OGSA, SLA (Service Level Agreement).*

1 Introduction

Nowadays in most of management systems the adaptation to change situations in accordance with business policies are not autonomic, and they don't generally manage themselves. In order to do so, self-management information systems and autonomic architectures are needed.

This paper is organized as follow. After an introduction, the second section introduces a comparison between current and autonomic computing, the third section outlines advantages and limitations of link between autonomic computing and OGSA. The fourth section presents examples of IBM autonomic computing Initiative and the framework ASF (Adaptive Services Framework) which is designed by IBM and CISCO and outlines some challenges of self-management. The fifth section introduces our autonomic management architecture based on OGSA (Open Grid Services Architecture) and Peer-to-Peer model. A comparison between our autonomic architecture, OGSA and the autonomic architecture of IBM and CISCO is made in the sixth section. Conclusion and perspectives are given in the last section.

2 Comparison between current and autonomic computing architectures.

An autonomic system is made of a connected set of autonomic elements that contain resources and deliver services to humans and other autonomic elements. Autonomic elements will manage their internal behaviors and their relationships with other autonomic elements in accordance with policies that humans or other elements have established [8]. Autonomic computing consists of self-managing systems that means self-configuring, self-healing, self-protecting and self-optimizing which are summarized in table 1. This table contains also the comparison between current computing and autonomic computing.

The role of autonomic element consists on providing its services and managing its own behavior. To do so autonomic element monitors behavior through sensors, analyzes those data, then planes what action(s) should be taken, and executes that (those) action(s) through effectors. That creates a control loop [8] which allows to manage the systems (see figure 1).

The biggest challenge is building closed control loops, the most important concept of self-management.

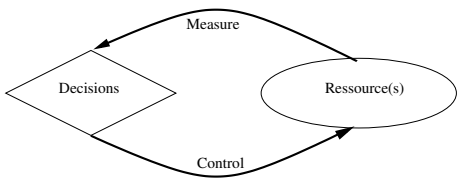

Figure 1. control loop.

Concept	Current computing	Autonomic computing
Self-configuration	Installing, configuring, and integrating systems are time-consuming and error-prone.	Automated configuration of resources and systems follows high-level policies. The Rest of the system adjusts automatically and seamlessly.
Self-healing	Problem determination in large, complex systems can take a long time	System automatically detects, diagnoses, and repairs localized software and hardware problems.
Self-Optimization	Hundreds of manually set, nonlinear tuning parameters and their number increases with each release.	Resources and systems continually seek opportunities to improve their own performance and efficiency.
Self-protection	Detection and recovery from attacks and cascading failures is manual.	System automatically protects itself against malicious attacks, cascading failures. It anticipates and prevents system wide failures.

Table 1. Four areas of self management.

3 Autonomic Computing (AC) and Open Grid Services Architecture (OGSA).

Autonomic computing proposes a solution for self-management system based on a service-oriented architectural approach (Web services or OGSA infrastructure). OGSA combines web services and grid computing with open interfaces, it can be seen [6] as an extension and a refinement of the emerging Web Services architecture. By combining these two approaches (autonomic computing and OGSA), the autonomic computing profits from the advantages of OGSA such as computational capacity and allows to integrate service mobility in management operations.

By adding the autonomic functionality to the OGSA and by using grid services, this approach makes it possible to provide higher QoS (Quality of Service) and a great availability of the services.

However the OGSA function only addresses self-healing, within autonomic computing by using Globus HeartBeat Monitor (HBM).We have to add self-Configuring, self-Optimizing, self-protecting. Clearly for autonomic computing to be effective in heterogeneous environments requires more researches on specifying an architecture for autonomic functions and events. A key element of autonomic computing will be the ability to correlate those events to determine what occurring in the environment and then start the corresponding autonomic functions [9].

To summarize OGSA seems to be primordial to accelerate the implementation of autonomic applications. However, to expand the applications beyond the level of a single enterprise, OGSA needs to more issues concerning: use of WSDL (Web Services Description Language) extensions, heterogeneous and end-to-end management, security, grid Service manageability and availability of grid service.

4 Examples of autonomic architectures and open problems about self-management

4.1 Autonomic computing Initiative (IBM) and Adaptive Services Framework (CISCO).

The autonomic computing Initiative (ACI) of IBM is based on the control loop and the four area of self-management(see table 1).

Cisco and IBM, made the decision to collaborate on an Adaptive Services Framework (ASF) [10] based on the Adaptive Network Care (ANC) of CISCO and the Autonomic Computing Initiative(ACI) of IBM [8].

ASF is a set of proposed interfaces and formats that allow customers to interact with service providers.

The SSP (Support Service Provider) acts as a proxy (mediation gateway) to achieve the actions of the autonomic manager for integrating multiple vendors services.

4.2 Challenges of self management

Every aspect of autonomic computing presents significant challenges [7]. The life cycle of an individual autonomic element or a relationship among autonomic elements reveals several challenges such as :

- The management of the relationships among autonomic elements,
- Ontologies and Semantics based reasoning,
- Autonomic elements location,
- Negotiation between Autonomic elements,
- Determination of self-management policy,
- Cooperation and learning in autonomic environment,

5 PARIS: our generic and global self-management architecture

5.1 Overview of our self-management architecture

At GET INT, the research works related to INT/AGIRS [1] [2] [3] [4] [5] has designed a generic architecture for the autonomic management of the heterogeneous networks and services, named PARIS (Platform for the autonomic Administration of netwoRks and Integration of multimedia Services). As depicted in figure 2, this architecture is divided into three generic classes.

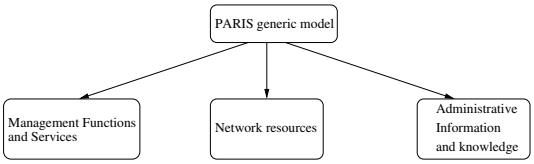

Figure 2. Overview of PARIS generic model

The main components of PARIS are:

- *Administrative information and knowledge:* This information shows the management resources use, like the services profile, the managers availability and the state of the managed resources. Therefore, this class helps the administrators to manage complex networks by providing strategic information and knowledge to the organization and by defining management policies, in order to provide them a dynamic network management.
- *Management functions and services :* This class gathers all the necessary resources only for management.
- *Network resources :* This class represents the resources which are managed by the services of management system.

The components of PARIS are organized in three-layers. The bottom level represents physical devices such as switches, routers and hosts, as well as logical services such as VLANs, IP networks, file servers, and web services. The medium level represents the autonomic management level which gathers all the necessary resources for autonomic management services. The top level is dedicated to SLS (Service Level Specification) and administrative information and knowledge.

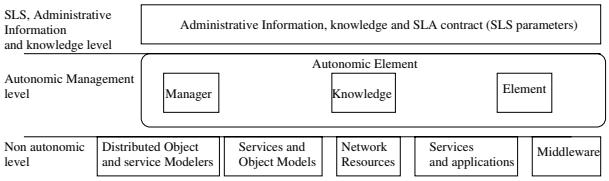

Figure 3. Overview of layering view of PARIS architecture

In the bottom level (see figure 3), composed with non autonomic resources, services, applications and systems, we use OGSA for virtualisation, self-healing, computational and middleware capabilities. So, for the autonomic level (level 2) all the resources of layer 3 are considered as services and are transparents.

In order to deliver an integrated service to customers the different interconnected service providers must cooperate through their management domains using their business policies and objectives.

5.2 Global QoS policy based network and services self-management

Our QoS criterias are flexibility, scalability, safety, delay, jitter, mainly availability and survivability. According to the comparison (table 2) between P2P and hierarchical architectures, we have choosen an hybrid architecture for our autonomic management (self-management) architecture.

The global QoS Policy management is based on a peer-to-peer approach (Peer-to-Peer QoS cooperation) between different operators' policy domains and a hierarchical approach in an operator's policy domain. An end-to-end QoS negotiation will take place to achieve the global business and policy goals.

The figure 4 shows an overview of the global QoS policy based services and network management. It represents the peer-to-peer and hierarchical approach management. The following figures will describe these approaches in more details. In an operator's domain, management functions are organized in three levels. The top level contains global management policy and SLS parameters to negotiate with other operators. Once the two operators agreed, The SLA is trans-

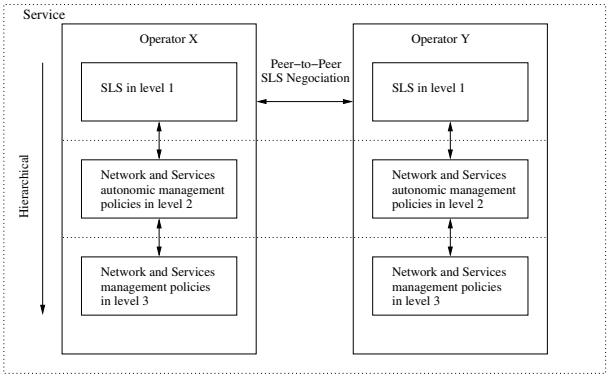

Figure 4. Global QoS policy based services and network management.

mitted to the second level (autonomic level) for enforcement then to the third level (non-autonomic level).

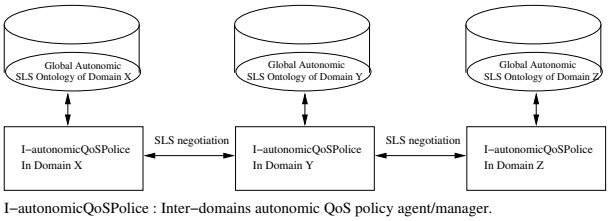

Figure 5. Peer-to-Peer QoS policy cooperation between different operators Domains.

The figure 5 highlights the QoS policy cooperation between the *Inter-domains-autonomicQoSPolicy Agents/Managers* of each operator domain : Each *I-autonomicQoSPolice* in one domain negotiates SLS parameters with other peer domains according to the global QoS Objectives.

Criterias / Architecture	P2P	Hierarchical
Response time	Slow/Medium	Fast
Survivability, Availability, Reliability	High	Low/medium
Scalability, Flexibility, Safety	High	low
Load for policy exchange	High	low
Manager between domains	No	Yes
Organization	Dynamic	Static

Table 2. Comparison between P2P and Hierarchical architectures.

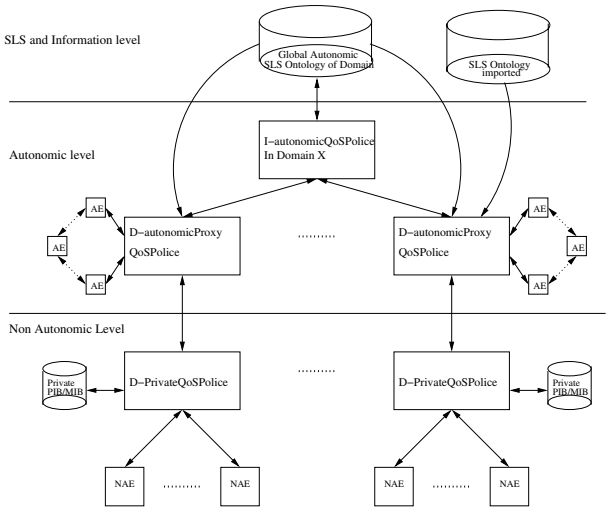

Figure 6. Hierarchical QoS policy cooperation in an operator Domain.

In the hierarchical approach (see figure 6) we distinguish clearly the three levels of an operator's domain. The *I-autonomicQoSPolice* interacts with the SLS ontology in the Administrative information and SLS level to get the global QoS policy and manage the *Domains-AutonomicProxyQoSPolicy Agent/Manager*. Each *D-autonomicProxyQoSPolice* can recover its own QoS policy from the SLS ontology of the operator's domain and transmit it to Autonomic Element (AE) which manage themselves. By the same way the *D-autonomicProxyQoSpolice* allows to manage the Non-Autonomic Element (NAE) by using the *Domain-PrivateQoSPolicy Agent/Manager* of each sub-domain (i.e SNMP Domain, TMN Domain...) which recover the policy management information from its private PIB/MIB(Policy Information Base/Management Information Base).

This way, the hierarchical approach allows an effective QoS policy cooperation in the domain and limits the fault management propagation and topology changes.

5.3 Scenario of SLA negotiation process in our peer-to-peer architecture

In this scenario, we will consider a virtual web hosting, on our P2P architecture, in which clients negotiate their services parameters using a web services ontology (specifica-

tion of the conceptualization as a hierarchy of concepts).

- **Semantic negotiation using ontology.**
  The web services ontology used contains a generic part about web services standard characteristics and a specific (local) one. For example, in the generic part, a web service belongs to a community service which is provided by a (or a set of) service provider(s). The QoS (Quality of Service) provided to the clients could be one or many of the following SLS parameters: availability, security, reliability,..., survivability. In the local part of this ontology, we have web hosting specific parameters such as operating system, transfer rate and storage disk space. A file SLAC (Service level Agreement Configuration) is used by the clients to negotiate their SLA contract using policies. An extract of the grammar of this ontology is presented below:

SLAC=<head><body>

<head>=<set of partners linked by this contract><period coverned>

<body>=<configuration data with access rights>
<service-data><policy rules>

<service-data>=<Hosting-plan><local-QoS><generic-QoS>

<Hosting-plan>=<Hosting domain(s) parameters>
<directory structure><access list><archive format>

<local-QoS>=<operating system><transfer rate>...<disk space>

<generic-QoS>=<availability><security>...
<scalability><survivability>

<policy rules>=<set of If-Condition(s)-Action(s)>

Some of these policy rules can be dedicated to penalities when the SLA contract between the client and the operator in not respected. In our prototype, this ontology is developed in OWL: this OWL description is out of the scope of this paper.

- **Process of SLA negotiation.**
  A negotiation agent will assume for the client the negotiation process using SLAC ontology file (see figure 7). Such an agent is located in a peer server on an operator's network. This peer server is called a Peer Autonomic Negotiation ServEr (PANSE).

After a client request to the UDDI (Universal Description Discovery Integration)/WSIL (Web Service Inspection Language) registry, if many responses are delivered from

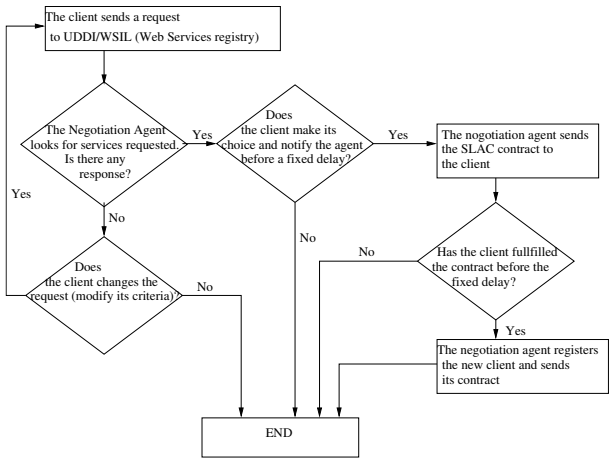

Figure 7. SLA negotiation process.

one or many operators, the client has to make a choice between them by selecting the best hosting plan (service) according to its SLS parameters. The selected hosting plan could be located in a PANSE (Peer Autonomic Negotiation ServEr) on the same peer domain as the client or on another peer domain. This negotiation process based on the semantic described in the ontology is dynamic because the SLS parameters, the peer negotiation servers and the peer domains selected could change at any time. So this process can contribute to the management of the mobility of users and their services profiles.

6 comparison between autonomic management architectures

In this chapter, we compare (see table 3) the autonomic computing Initiative(ACI) of IBM, the Service Framework of CISCO and the Open Grid Service Architecture (OGSA) with our architecture PARIS.

In table 3, it appears that our autonomic management architecture PARIS is more suitable to take into account business needs of ICT (Information and Communication Technology) operators and telecom managers, in term of global governance of their information systems, to support semantic and autonomic negotiation of configuration and services parameters and to permit self-organization in an operator's peer domain by using shared administrative information, ontology and self-governing capabilities of autonomic elements.

6.1 Comparison between ASF, ACI & OGSA.

After the analyze of ASF (CISCO/IBM) and ACI (IBM) architectures, we noticed that there are several points of

similarity between them. Both of them are based on service oriented architecture and they use similar standards to develop Web services. However the ASF framework proposes five levels of security (Authentication, Authorization, Encryption, Data Privacy, Signature) contrary to ACI and OGSA (which represents several gaps of security).

Criterias / Architecture	OGSA	ACI	ASF	PARIS
Self-configuring	-	+	+	+
Self-healing	+	+	+	+
Self-protecting	-	+/-	+	+/-
Self-optimizing	-	+	+	+
Services oriented Architecture and virtualization	+	+	+	+
Taking into account business needs of ICT operators and telecom managers: Global governance of their information systems	-	-	-	+
Taking into account mobility and nomadisme	+/-	+/-	+/-	+/-
Complete self-organization, dynamic and end-to-end Qos management	-	+/-	+/-	+/-
Interface with non autonomic environment and complete integration	+/-	-	+	+/-
Semantic and automatic negotiation of configuration and services parameters	-	-	-	+

Table 3. Comparison between OGSA and autonomic management architectures.

6.2 Advantages of our autonomic management architecture

The global P2P management architecture in our administrative information and SLS layer supports *concurrent* multi-manager control of network elements. The regrouping of manager-element roles improves *safety* by eliminating the state synchronization problem between managers and elements. The replacement of management agents by Autonomic Management Elements improves reliability through reductions in the size and complexity of implementing managed network services. The P2P management architecture also provides scalable monitoring and control of network elements. Management functions can be safely distributed across multiple managers due to the protection of transactional concurrency control. The unification of the manager and element roles in a peering relation enables the delegation of management functions, effectively distributing management load and supports *self-healing* in the face of local network failures.

This new peer-to-peer architecture benefits from the advantage of both approaches, autonomic computing and peer-to-peer, in order to allow an autonomic and dynamic management and to provide to the user a service with a satisfactory quality of service (availability).

7 Conclusion and Prospective work

Our peer-to-peer autonomic management architecture offers several advantages over the traditional manager-agent (client-server) architectures by creating a flexible, scalable, reliable and survivable environment supporting safe multi-manager access. The unification of the traditional roles of manager and element allows management functions to be distributed in different elements supporting autonomic behavior.

Future research will determine the granularity of distribution (service, node, Autonomic Element...), will extend the security and mobility management aspects and will detail the complete integration of non-autonomic devices, such as hubs, switches, etc. The other points of our research will be to define exactly how autonomic elements interact between themselves to allow a cooperation and learning in autonomic environment and how to make possible a reliable and complete global governance of the information systems of organizations (virtual operators, extended enterprises, etc.).

References

[1] A. Ag Rhissa and al., Results of AGIRS project, GET INT, http://www.int-evry.fr/recherche, december, 2004.

[2] A. Ag Rhissa, AGIRS project, Web technologies and information systems, Scientific meeting of GET at ENST Paris, october 14, 2004.

[3] F. Benayoune et L. Lancieri, Models of Co-operations in Peer-to-Peer Networks-A Survey , 3rd European Conference on Universal Multiservice Networks, Vol. 3262: 327-336, 2004.

[4] F. Benayoune, Adaptive management of content services for new generations of mobile networks, ongoing work of PhD thesis, Directors of thesis: A. Ag Rhissa et P. Vincent, INT/AGIRS and France Télécom R&D Caen, 2004.

[5] A. Hassnaoui, Autonomic and dynamic management of networks and services, outgoing work on the PhD thesis, Directors of thesis: A. Ag Rhissa et P. Vincent, INT LOR/AGIRS, 2004.

[6] I. Foster and C. Kesselman and J. Nick and al., The Physiology of the Grid: An Open Grid Services Architecture for Distributed Systems Integration, Globus Project, 2002.

[7] K. Herrmann, G. Muhl and K. Geihs, Self Management: The Solution to Complexity or Just Another Problem?, in the IEEE DISTRIBUTED SYSTEMS ONLINE, Vol. 6, No. 1, 2005.

[8] J. Kephart and D.M. Chess., The vision of autonomic computing, in the IEEE Computer Journal, Vol. 36: 41-50, 2003.

[9] Roy Sterritt, Towards Autonomic Computing: Effective Event Management, in the 27th Annuel NASA Goddard/IEEE Software Engineering Workshop (SEW-27'02), 2003.

[10] T. Studwell and K. Sankar, Adaptive Services Framework, Wd-asf-1.00, 2003.