Towards a Flexible Middleware for Autonomous Integrated Management Applications

Mehdi Kessis, Pascal Déchamboux*, Claudia Roncancio**, Thierry Coupaye*, Alexandre Lefebvre**
**France Telecom, Research & Development, MAPS/AMS*
28 chemin du Vieux Chêne, 38243 Meylan CEDEX, France
{Firstname.Lastname}@francetelecom.com
***LSR-IMAG*
{Firstname.Lastname}@imag.fr

Abstract

Enterprise and global-scale systems today might have thousands to millions of geographically distributed nodes and this number will increase over time. Managing efficiently such scattered systems becomes increasingly complex and requires powerful management capabilities. Traditional solutions to manage and control them seem to have reached their limits. In recent years, integrated management systems and services as well as autonomic systems have raised much interest in distributed systems and software engineering. This paper discusses the architectural issues facing the design of large-scale distributed management systems. Then it suggests a new flexible and scalable integrated management middleware to handle management problems in large-scale networked and heterogeneous systems.

Key words:
Integrated management, autonomic, component-based architecture.

1. Introduction

In recent years there has been a considerable growth in the use of distributed systems (peer-to-peer networks, clusters, pervasive computing, sensor networks, etc). These systems are scattered in our companies, administrations, homes and even in our pockets. Typically, they consist of large numbers of heterogeneous computing devices connected by communication networks, using various operating systems, resources and services, and user applications running on them. The dependability of our societies on such systems is becoming more and more noticeable. However, they become larger and more complex, heterogeneous and scattered. Traditional solutions to manage and to control them seem to have reached their limits. The size and the complexity of distributed system make it hard or even impossible to manage each of their components. Today, enterprise networks may count tens to thousands of managed resources. Telecommunication operators may manage thousands to millions of resources. For instance, France Telecom is managing more than one million "Livebox" home gateways. Due to the increased cost and complexity of managing such infrastructures manually, distributed computing systems are moving towards more autonomous operation and management.

This paper discusses an integrated management middleware that deals with such complex environments. This middleware plays the role of management broker that can be used by administrators or by management applications (i.e., management processes). It offers them (a) a customized view of the system through resource monitoring functions and (b) resource control management functions. We believe that such an infrastructure may offer a high degree of autonomy through management applications, by automating actions on groups of resources. We note that, in order to be able to automate such processes, a rich information model as well as a powerful programmatic model are needed.

This paper overviews our ongoing research. We aim at investigating new approaches to handle scalability, autonomy and heterogeneity issues in system management. After discussing new management needs and challenges, this paper proposes a middleware architecture to deal with these issues. Our work focuses on the monitoring activity, bringing flexibility and adaptability to the proposed solution.

2. Distributed management challenges

Today, large-scale distributed systems management is facing several major challenges. In this study, we focus on four of these challenges: autonomy, scalability, heterogeneity, and administrative isolation.

2.1. Autonomy

Managing efficiently such scattered systems becomes increasingly complex and requires powerful management capabilities. Traditional solutions to manage and control them seem to have reached their limits. In recent years, integrated management systems and services, autonomic systems have raised much interest in distributed systems and software engineering [14]. An autonomic system is capable to repair, configure, heal and protect itself [14]. The emerging field of autonomic distributed computing addresses the challenge of how to design and build distributed computing systems that can manage, heal and optimise themselves. Distributed computing systems are moving towards increasingly autonomous operation and management, in which their interacting components can organise, regulate, repair and optimise themselves without human intervention. [32]. These systems are intended to tackle administration complexity that is out of reach of human administrators, for instance handling a large number of alarms and notifications. Besides, automating management may reduce cost and improve efficiency. To automate management, we need at least three key elements: (a) representation, observation and monitoring capabilities, (b) decision rules and mechanisms and (c) control mechanisms.

2.2. Scalability

Scalability is a major problem for large-scale distributed systems. There is no commonly accepted definition of it [9]. In this paper, we consider the following definition of scalability: *"A scalable system is one that maintains constant, or slowly degrading, overheads and performance as its size increases*" [8]. In the past years, centralised network management has shown inadequacy for efficient management of large heterogeneous networks. As a result, several distributed approaches have been proposed to overcome the problem [21]. This is a main concern of our work because an enterprise network is an order of magnitude less complex than the infrastructure of some service providers, who monitor thousands to millions of resources. There are two key aspects of scalability involved in system management: the size of networks and the number of users. Service providers create extreme demands on both aspects[1]. Management systems should accommodate large numbers of participating nodes and they should allow applications to monitor large numbers of managed resources. Grouping and distributing management operations may improve scalability of the management system.

[1] The international Engineering Consortium (http://www.iec.org), Performance Management of Next Generation Networks.

2.3 Heterogeneity

The information model is a key feature in any management system [2]. It offers a view of managed resources (network, services, applications, etc.) to management applications. Today's networks involve heterogeneous resources. A service failure can be related to network or to application failures. In order to rapidly identify causes of failures and to understand the behaviour of complex managed resources, it is important to be able to describe heterogeneous managed resources and their interactions in the same way. The main object of integrated management [12] is to integrate different types of management (policy, user, network, services) in a unique infrastructure. Such infrastructure offers a complete view of the managed environment. To do so, we need a common description and representation of managed resources and their interactions.

2.4. Administrative Isolation

Traditionally, in large-scale networked systems, elements are grouped in managed domains. *"A domain is a set of objects to which a common management policy applies"* [18]. Each management domain is a logical partition of managed resources and management services. The set of managed objects may include computers, people, privileges, software processes, etc, depending on the purpose for which the domain is defined. In order to deal with large-scale networked and interconnected systems, domain management tools are needed. Such tools offer to the administrator the possibility to create, extend or merge new logical domains from existing primitive domains. Let us consider the France Telecom home gateway example. An example of primitive management domain could be the set of gateways related to a particular DSLAM. A management domain might contain the logical partition of gateways of a particular geographic zone, while another might contain the set of "LB1234" gateway model, in order to update their firmware. Administrator needs automated tools to build and to interact with management domains.

3. Towards a flexible and scalable integrated management middleware

3.1 A component-based model

Flexibility is a required property for managing large-scale networked systems. [11,7]. With a model such as the one proposed by Fractal [1], we believe that we can design, build and dynamically reconfigure component-based management infrastructure.

Fractal is a generic component model focusing on reconfiguration using flexible composition of components. It adopts a recursive view of components that may be nested. A component owns a *membrane* (i.e., a set of controllers), which realizes arbitrary forms of control over the content of the component. Composite components include other sub-components. A component sends and receives invocations through access points called interfaces. Such interactions require communication channels (named *bindings*) between some of the component interfaces. Figure 1 illustrates the architecture of a Fractal component. This composite component contains two sub-components and exposes control interfaces C_1 and C_2 as well as functional interfaces C_S and C_c.

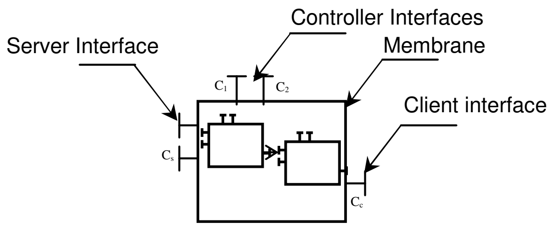

Figure 1 Example of Fractal components

Three main Fractal features are of particular interest:

a) *Component hierarchy*: composite components recursively contain components, ending with primitive components.

b) *Component sharing*: a (sub) component can be contained in several composite components. Typically, this feature can be used to model resources that are intrinsically shared.

c) *Components dynamicity:* bindings between components can be manipulated at runtime and is particularly interesting for management purpose. The model allows the definition of flexible bindings, since bindings may be themselves components.

Such properties are very interesting to design and build management domains. Figure 4 shows an example of mapping between domains and Fractal components. Disjoint domains are represented by disjoint components. Overlapping domains are represented by shared component. Hierarchical domains can be represented by composite domains.

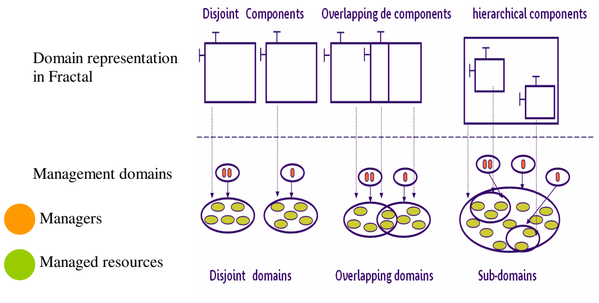

Figure 2 Management Domains in Fractal

3.2. Global management architecture

This section introduces the global architecture of an integrated management middleware that is positioned between management applications and managed resources. Managed resources are the set of physical resources (switches, PC, PDA, Set-Top Box, etc.) and logical resources (all or only a part of the OS, middleware, applications, services, etc.) available in an operator network.

A middleware relying on a flexible overlay network is a promising approach to overcome issues outlined in section 2. Such an approach allows management applications to construct an abstract view of the underlying network infrastructure and to federate different networks (IP, ad-hoc, etc). The middleware we propose builds an overlay network of mediation nodes that support management domains. The overlay network is functionally independent of the network infrastructure. It is formed by mediation nodes, which are interconnected through logical links. Figure 3 illustrates the global architecture of the proposed management infrastructure. Nodes collaborate in order to respond to queries of management applications. Each node represents one or many management domains. A node may contain one or many sub-domains (hierarchical relation). It interacts with several domains (links between nodes). This overlay builds an abstract representation of physical management domains (geographical for example) through management domains that correspond to specific management needs. (e.g., set of gateways model "LB1234"). Each node of the overlay offers a set of management services or tasks (information repository management, resource location service, query service, etc). As it can be noticed, these services cover only non functional management aspects delegated by management applications to the middleware.

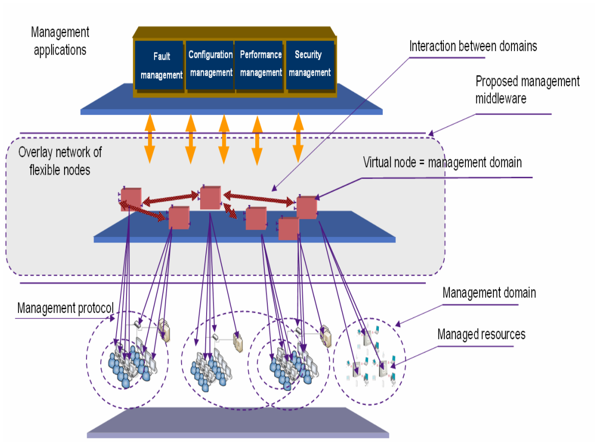

Figure 3 Global architecture of the management middleware

Both nodes structure and their interconnections are managed by the middleware administrator in a transparent way. The middleware hides management applications the complexity of the underlying infrastructure. The resulting overlay network is totally flexible, extensible and dynamically reconfigurable. Nodes and interconnections are dynamically reconfigurable.

To better understand the behaviour of these nodes, let us zoom inside one of them as depicted in Figure 4. Inside a node, a set of components work together in order to achieve a function. In Figure 4, the node offers 4 services: (i) events management service (that handles events and routes them to interested entities), (ii) a CIM repository (representation of the managed infrastructure), (iii) a query management service (for querying CIM repositories) and (iv) a repository for naming resources.

We believe that management applications requires rich information modelling to take into consideration physical and logical interdependent resources. Common Information Model (CIM) [4] is a standard for defining device, network and application characteristics so that system and network administrators and management programs can control heterogeneous devices and applications.

CIM also allows for vendor extensions. The adoption of such a model is key to overall interoperability for information storage and for system management environment.

The proposed middleware is based on two API:

a) *A configuration and deployment API*: It concerns the mediation nodes and management services. Network administrator, can build, deploy and configure the management middleware. After configuration and deployment, the middleware is ready to be requested by management applications.

b) *A mediation API*: This API is used by management applications. It permits them to communicate, indirectly, with managed resources, through our middleware.

Although both interfaces have different concerns, they are both managed by administrators at different level. Furthermore, the proposed middleware should use itself for its own management issues.

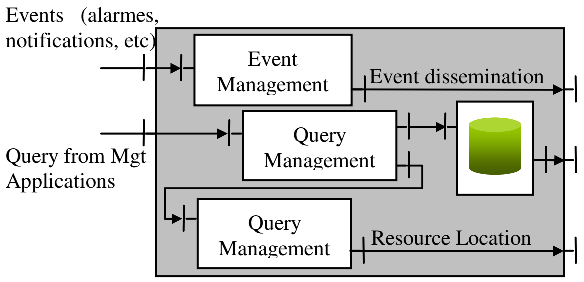

Figure 4 Internal structure of mediation node

4. Discussion

The choice of component-based architectures for managing networks and services have several advantages. V. Wadel et al. [17] consider them when designing management solutions within the world of telecommunication (flexibility, modularity, clear design, etc.).

The management applications are designed in order to be independent from the size of the network or from the number of properties to ensure. The management middleware aims at routing the requests over management information, supporting persistence of this information when necessary, locating resources, etc. It provides mediation nodes whose one of the primary roles is to ensure that its functions whatever are the conditions of the infrastructure that support the management network (i.e., an overlay network). The objective is that this middleware can adapt to such diverse situations as sensors network or as computation grids. This means that flexibility and adaptability are among the main challenges we target. These properties should ensure that our middleware can adapt to the various functional requirements of management applications, and can also adapt its own behaviour to the resources dedicated to its operation. We argue that component-based architecture is a major enabler towards this goal.

The Fractal component model we rely on allows the definition of components as assemblies of components and provides total control over the component used as well as the bindings between them. Hence, at the very end, the overlay can be considered a component composed of other components (e.g., the mediation nodes), managing them and their relationships as well. The overlay can then be configured and reconfigured in order to respond to management needs.

Autonomous management is seen as the only means to deal with large-scale management. This approach will lead to very complex applications to support the processes composing this autonomous management environment.

The information model on which the management applications rely is highly distributed by nature. So should be the middleware for supporting them. We consider that good work has been done for specifying the information model, especially with CIM [4]. Our objective is to be able to organise the management of this information space in a distributed manner while ensuring its safety, scalability, correctness (i.e., in a sense considering the implementation of a reliable distributed CIMOM). We also consider much simpler interfaces to manipulate this information model, especially within our Java implementation context. For example, we consider pure Java objects accessible through technologies such as EJB[2] or JDO[3] for giving access to the CIM repository.

5. Related Works

Several works studied large-scale systems and network management, during these two last decades [6, 10, 20, 8].

CIM/WBEM [6], a DMTF[4] standard, proposes a web-based management architecture. WBEM is built around the CIM model. The main WBEM architecture is composed of three main elements (management applications as client, WBEM server, WBEM providers (probes and actuators)). This architecture follows a flat and centralized model (Manager/Agent model). There is no M to M (Manager to Manager) communication. WBEM servers can be used to manage enterprise environments. However, it does not scale to large distributed environments. The administrative isolation is implemented through the concept of namespace (logical view of CIM instances and classes).

Yalagandula et al proposed SDIMS (Scalable Distributed Information Management System) [5]. It consists of a building block for large-scale distributed services. SDIMS aggregates information about large-scale networked systems and provides detailed views of information (and events) and summary views of global information. It ensures four properties: scalability, flexibility, administrative autonomy and robustness. This work concerns neither resource heterogeneity nor autonomic behavior.

Renesse et al [8] proposed a similar work: Astrolabe. This system gathers, disseminates and aggregates information about zones. A zone is recursively defined to be either a host or a set of non-overlapping zones. It supports scalability through hierarchy (zone hierarchy), flexibility through mobile code, robustness through a randomized peer-to-peer protocol and security through certificates. Each Astrolabe zone has a set of aggregation functions that calculates the attributes for the zone's MIB (SNMP like Management Information Base). Astrolabe is designed under the assumption that MIBs will be relatively small objects, a few hundred or even thousand bytes, not millions which limit its scalability.

Anerousis et al [10] proposed Marvel. This system is a distributed computing environment that allows the creation of scalable management services using intelligent agents and the world-wide web. Marvel builds on top of existing element management agents a hierarchy of servers that aggregate the underlying information in a synchronous or asynchronous fashion. Marvel is based on an information model that generates computed views of management information. These views follow an object-oriented model to store management information. Marvel requires that managed elements be organized into groups. Users can dynamically define these groups based on any factor that makes sense such as location or functionality. The object implementation of Marvel's views is proprietary and not extensible. Besides it does not address autonomy issue.

Recently, Bouchenak et al [15] proposed the JADE framework. It is an environment for implementing autonomic administration software. The main idea of this work consists on modelling the administrated system as a component based software architecture which provides means to configure the environment. A prototype of Jade was developed and used for deployment and fault management of clustered J2EE application. This work provides administrative isolation through composition and sharing relations. This work is based on an ad-hoc information model. The global vision of the proposed work in this paper is coherent and complementary with the autonomic management vision proposed in the European project IST Selfman [3], to which we actively participate.

6. Conclusion and Future works

In this paper we have studied large scale networked heterogeneous systems problem. We have identified four important properties that large-scale management

[2] http://java.sun.com/products/ejb/docs.html

[3] http://java.sun.com/products/jdo/

[4] Distributed Task Force Management; URL: http://www.dmtf.org

systems have to respect: autonomy, scalability, administrative isolation and heterogeneity. We suggest the architecture of a flexible middleware that respects these properties. The proposed middleware is based on the Fractal component model. It offers the administrator the possibility to build a scalable and dynamically reconfigurable overlay network of mediation nodes. Each node represents on or many management domains. The middleware offers the possibility to build, aggregate, select, and query management domains according to administrator needs. All these operations can be done without interrupting management applications activity. The different nodes cooperate to offer management applications several services (event filtering, aggregation, routing, storage management, etc). Scalability is achieved through the support of domain and through the distribution of the management activity.

We believe that the proposed middleware can be a powerful building block for autonomous management applications. A set of management policy can be defined for each management domain that we build. Automatic actions can be assigned to each of them. Scalability is achieved through distribution of the management infrastructure and grouping management operations. Heterogeneity is achieved through CIM local repositories, managed by the node of our overlay network. In these repositories, heterogeneous resources are described in a standard way. Administrative isolation is achieved through the different possible composition relations proposed by the Fractal component model. Autonomy is achieved through the reflexive aspect of the Fractal components.

We are studying the possibility to integrate a data steam management system to handle, in a scalable way, streams of events sent by probes and network equipments. This feature is particularly interesting for large event-based monitoring systems in real-time context. Besides, we are studying the possibility to make some of our node mobile. The ProActive[5] technology, based on the Fractal component model, has already experienced such an approach.

7. References

[1] E. Bruneton, T. Coupaye, and J.-B. Stefani. "Recursive and Dynamic Software Composition with Sharing". Proceedings of the Seventh International Workshop on Component-Oriented Programming (WCOP02), Malaga, Spain, June 10-14, 2002.

[2] J.-P. Martin-Flatin, "Toward Universal Information Models in Enterprise Management", in Proc. VLDB 2001 Workshop on Databases in Telecommunications (DBTel 2001), Rome, Italy, September 2001.

[3] P. Van Roy, A. Ghodsi, JB Stefani, S. Haridi, T. Coupaye, A. Reinefield, E. Winter and R. Yap. " Self management of large-scale distributed systems by combining structured overlay networks and components". Workshop IST NoE CoreGrid Integration, Greece, Nov 2005.

[4] Common Information Model Standard, URL: http://www.dmtf.org/standards/cim/

[5] P. Yalagandula and M. Dahlin, "A scalable distributed information management system ", Proceedings of the 2004 conference on Applications, technologies, architectures, and protocols for computer communications, session Distributed information systems, Pages: 379 – 390, Portland, Oregon, USA, 2004.

[6] Java Specification Request N°48 (JSR 48): WBEM Services Specification, URL: http://www.jcp.org/en/jsr/detail?id=48

[7] J. Won-Ki Hong, J. Kim and J. Park: "A CORBA-Based Quality-of-Service Management Framework for Distributed Multimedia Services and Applications". IEEE Network, Vol. 13, No. 2, (1999) 70-79

[8] R. V. Renesse, K. P. Birman, and W. Vogels, "Astrolabe: A Robust and Scalable Technology for Distributed System Monitoring, Management, and Data Mining", ACM Transactions on Computer Systems, Vol. 21, No. 2, May 2003, Pages 164–206.

[9] M. D. Hill, "What is scalability?". ACM SIGARCH Computer Architecture News, December 1990.

[10] N. Anerousis and G. Hjálmtysson, "View-based Management of Services in a Programmable Internetwork". Proc. of the 2000 Network Operations and Management Symposium, Honolulu, HI, April 2000.

[11] G. Goldszmidt, "Distributed Management by Delegation". 1996. Ph.D Thesis – Graduate, School of Arts and Sciences, Columbia University, New York.

[12] H.G. Hegering, S. Abeck and B. Neumair. "Integrated Management of Networked Systems: Concepts, Architectures, and their Operational Application". Morgan Kaufmann Publishers, 1999.

[14] J. O. Kephart and D. M. Chess." The vision of autonomic computing". IEEE Computer, 36(1):41–50, January 2003.

[15] S. Bouchenak, N. de Palma and D. Hagimont, "Autonomic administration of clustered J2EE applications". Proceedings of IFIP/IEEE International Workshop on Self-Managed Systems & Services (SelfMan 2005). 2005.

[16] M. Kahani, H.W. Peter Beadle, "Decentralized Approaches for Network Management", in SIGCOMM, July 1997.

[17] V. Wade, D. Lewis, C. Malbon, T. Richardson, L. Sorensen and C. Stathopoulos, "Component Integration Technologies for Telecoms Management Systems", TCD-CS, Technical Report, Trinity College Dublin Computer Science Department, 1999.

[18] M. Sloman, J-D. Moffett, "Domain model of autonomy". ACM SIGOPS European Workshop 1988

[5] http://www-sop.inria.fr/oasis/ProActive/