Global Discovery Service for JMX Architecture

Jacek Midura, Kazimierz Balos, and Krzysztof Zielinski

Deapartment of Computer Science
AGH University of Science & Technology
Al. Mickiewicza 30, 30-059 Krakow, Poland
{kbalos,kz}@agh.edu.pl

Abstract. Distributed computing in wide area networks requires special approach to solve the problem of communication between secure areas. Since there is no possibility to establish connection from the outside to secure region and no possibility to discover hidden resources (i.e. grids), special mechanisms are required to do this job. Article presents a new approach to solve these two problems by introduction of agent oriented design for establishing connections and resources discovery. Moreover, presented architecture stands for resources access transparency and is suitable for resources management in grids, especially using JMX (Java Management Extension) technology.

Keywords: Discovery service, resources management, firewall, JMX, MBean

1 Introduction

JMX (Java Management Extension) [1] specifies a framework for distributed resources management. It offers general architecture suitable for construction of monitoring systems and management applications. The spectrum of its applicability covers among others grids systems [4,5], active networks, and mobile systems. Full exploitation of JMX functionality is influenced by security policy and communication limitation imposed on Internet by firewalls.

Monitoring and management of distributed managed resources over Internet requires their dynamic discovery and binding to a management application at the first place. The built into JMX implementations standard discovery services do not work over firewalls. This is a basic constraint that reduces a scope of JMX applications over public networks.

The main goal of this paper is to present Global Discovery Service (GDS) for JMX architecture. This service is called global because it has been designed to work over firewalls. The proposed service could be used not only for resource discovery but offers also mechanisms to take control over resources hidden by firewalls. In the design and implementation phases every measures have been taken to make GDS scalable and lightweight.

The paper is structured as follows. In Section 2 major requirements of Global Discovery Service have been discussed. In Section 3 GDS concept and architecture have been described. GDS implementation details have been presented in Section 4. The paper is ended with conclusions.

M. Bubak et al. (Eds.): ICCS 2004, LNCS 3038, pp. 114–118, 2004.

2 Requirements of Global Discovery Service

The JMX architecture described in [1,2] is fully operational in a wide area network which allows communication with RMI protocol via firewalls. The discovery services exploit broadcast communication that imposes even more harder to fulfill requirement on routers configuration over the public network.

Dynamic discovery can be divided into two categories: passive discovery and active discovery. In the first case client notifies the server that its state has changed. In second scenario there are special objects: *Discovery Client* on the client side and *Discovery Responder* on the agent side. *DiscoveryResponder* notices the request sent via multicast and then replies using unicast transmission.

Discovery mechanism is usually supported by the heartbeat mechanism, which is used for connection failure detection. To monitor the connection, the connector client sends periodic heartbeats (ping requests) to the connector server that acknowledges them by sending a reply (ping responses). If heartbeat message is lost, the connector retries until either the connection is reestablished or the number of retries has been exhausted and connection is released

The typical configuration of Internet firewalls allows communication via HTTP or HTTPS protocol under the condition that this communication is initiated from a secure region to a public network. This asymmetry creates the additional difficulties not only in JMX discovery services activity but also in standard management operations invocation.

Most of very important nowadays applications represent collection of systems connected via public network. The typical example are grids that represent collection of clusters connected into one system via Internet. Another example are mobile systems that should be localized despite its connectivity to secure part of the system. Though these problems could be resolved by VPN establishment over the public network, this solution breaks our assumption that JMX based applications operate over standard Internet.

During the design phase of GDS the following major requirements have been taken into account:

- the service should provide possibility of management and monitoring resources through MBeans located in the secure region, hidden by firewall, using only HTTP/HTTPS connections.,
- it should provide discovery of resources hidden in the secure area,
- the service should also provide mechanism supporting localization of resources in a network topology, especially provide information about agents' neighborhood,
- the implementation should be lightweight and scalable and should be fully compliant with services already defined within JMX framework.

3 GDS Concept and Architecture

During the design phase of GDS a few problems have to be resolved step by step. The process of obtaining a final solution is repeated in this section to justify the role of the proposed system components.

The most important problem concerns the management application which isn't able to establish connection to MBeans Server located in a secure region. The reason is that communication must be initiated from the secure region. To solve this problem a new element called *Join Agent* has been introduced. Function of this agent is initiation of communication connection with Management Application from secure region on one side and with MBean Server on the other. To discover MBean Server in the secure region Join Agent may use the standard JMX Discovery Services. This solution has still three disadvantages:

- It is impossible to discover of MBean Servers by Management Application,
- It is very difficult to chose a good strategy when Join Agent should establish connection to Management Application – there is no information when this application has been started.
- It is necessary to inform Join Agent in some way where Management Application has been started.

To eliminate these drawbacks the second element called *Mediator Agent* has been introduced. It is placed outside the secure region (or in the same region as Management Application) and by assumption is active all the time. The system activity sequence in this case is as follows: first Mediator Agent is started, next in any time MBean Server via Join Agent could establish communication with Mediator Agent. The Management Application (one or more) may also open communication with Mediator Agent obtaining this way access to all already connected MBean Servers. The proposed solution has been depicted in Fig. 1. It is free of drawbacks of the two elements solution and makes possible:

- discovery of MBean Servers connected to Mediator Agents,
- establishing communication at any time because Mediator Agents is always active so the Join Agents can access it,
- easy finding of Mediator Agents by Join Agents because localization of it could be fixed under well know address.

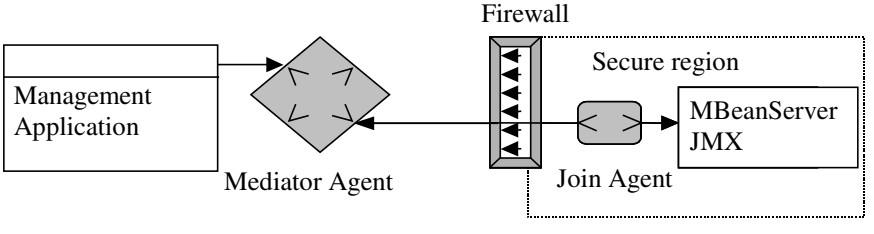

Fig. 1. GDS system elements

To finally clarify the concept of GDS it is necessary to explain how the operation invocation could be forwarded between Mediator Agent to Join Agent by the connection opened in the opposite direction. JMX notification mechanism is employed for this purpose. Join Agents is implemented as MBean which is registered with Mediator Agents. Mediator forwards operations' invocations encapsulated in a notification events. Internal structure of the Mediator Agents may be summarized as follows:

- allows connection of Join Agents and Management Applications – connector servers have to be installed for that purpose,

– forwards management operations' invocations to suitable Join Agents,
– makes possible searching for connected agents with Discovery Service.

The last interesting feature of presented system is its ability to display neighborhood relations between discovered agents. To achieve this goal there is performed special discovery procedure with TTL initially set to 1 and increased subsequently.

4 GDS Implementation

Concept of global discovery mechanism inspired the GDS prototype development, which is presented in this section. GDS implementation was split into three modules: management application, Join Agent and Mediator Agent. All elements of GDS infrastructure excluding management application are designed either as MBean Servers or MBeans, what makes them consistent with infrastructure of existing management systems. Such an approach makes management methods invocations of all MBeans transparent to client, which should bother neither with complexity of communication through secure regions nor with the discovery issues [9].

MBean Server with monitored resources and Join Agent is connected to Mediator Agent, which dynamically creates another MBean Server, called MBean Server Proxy. MBean Server Proxy is special implementation of JMX *MBeanServer* interface, and is equipped only with required connectors and mechanisms for forwarding methods and returning results from remote MBean Server. Having this special implementation of *MBeanServer* interface, there is no possibility to distinguish between real MBean Server and MBean Server Proxy by client application, what is the main goal and advantage of presented GDS [9].

Fig. 2 shows the main window of management application connected to remote agents and managing hidden resources.

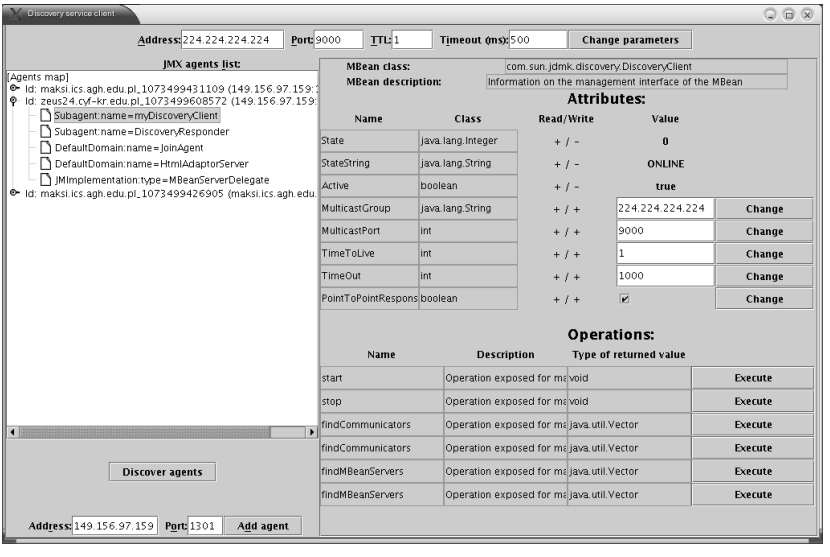

Fig. 2. Screenshot of management application

5 Conclusions

Presented architecture and its first implementation prove that dynamic resources discovery is not reserved only for local area networks, but can be fully operational in wide area networks, as it is in the Internet. GDS provides not only the global discovery utility functions, but makes discovered resources fully transparent for management applications.

Plans for the future GDS evolution should involve moving from JDMK to JMX reference implementation in order to create GDS package not limited by proprietary license. In some cases traffic encryption should be also considered, what probably can be achieved using standard secure communicators found in Sun's implementation of JMX Remote API [2].

References

1. Sun Microsystems: Java Management Extension Reference Implementation (JMX), http://java.sun.com/products/JavaManagement/
2. Sun Microsystems: JMX Remote API Specification (JMX Remote API), http://developer.java.sun.com/developer/earlyAccess/jmx/
3. Sun Microsystems: Java Dynamic Management Kit Specification v. 4.2 (JDMK), http://java.sun.com/products/jdmk/, JDMK42_Tutorial.pdf, 121-128
4. The Open Grid Services Infrastructure Working Group (OGSI-WG): OGSI Specification, http://www.ggf.org/ogsi-wg
5. The Open Grid Services Architecture Working Group (OGSA-WG): Globus Tutorial, www.globus.org/ogsa/
6. M. Sylvain, G. Leduc: A Dynamic Neighborhood Discovery Protocol for Active Overlay Networks, IWAN2003 Conference, Kyoto Japan (10-12 Jan. 2003)
7. Object Management Group (OMG): Notification Service Specification. New Edition, v.1.0 (2000)
8. A. Laurentowski, K. Zieliński: Experiences from Implementation and Evaluation of Event Processing and Distribution of Notifications in an Object Monitoring Service, The Computer Journal Vol. 47 No. 2 (2003) 1-16
9. J. Midura: Monitoring system of distributed computer resources. M.A. thesis, Kraków (2003, in polish) 31-32, 50-70
10. L. Bizon, M. Rozenau: Web services application in computer system integration. M.A. thesis, Kraków (2003, in polish) 33-37
11. R. Stallings: SNMP, SNMP v2.0 and CMIP. A Practical Guide to Network Management Standards, Addison Wesley (1993)