Management of Outsourced Systems

S P Rana, D Robertson, P Sellek and Dr. M Fisher

BT

Abstract: This paper aims to describe a policy based management system for outsourced businesses. We describe a prototype which offers custom real-time and historical views over the performance of a hosted application and the ability to control the resources available to the application remotely and dynamically.

1. Introduction.

Outsourcing is the process of subcontracting business operations and support to an organisation outside of the company. The main motivation for outsourcing the management of applications is to achieve acceptable levels of performance at the lowest cost. Our idea is to support customer applications on flexible computing infrastructure and to give customers explicit control over the IT resources deployed for their benefit. Control has to be offered in a meaningful way, presented in terms that make business sense rather than in terms of performance statistics whose relationship to the parameters that matter to the customer may be indirect.

A prototype has been built to investigate the issues involved in presenting application performance to the customer in a meaningful way. It also allows the customer to control the resources he uses even though the computing infrastructure is owned and managed by an external provider. The approach taken here to achieve this uses policies to specify the behaviour of the system in a flexible way and an event-based system for propagating changes in the state of the system in an efficient way. The use of policies allows selection between the options built into systems to be made and modified at runtime, rather that being hard coded into processes.

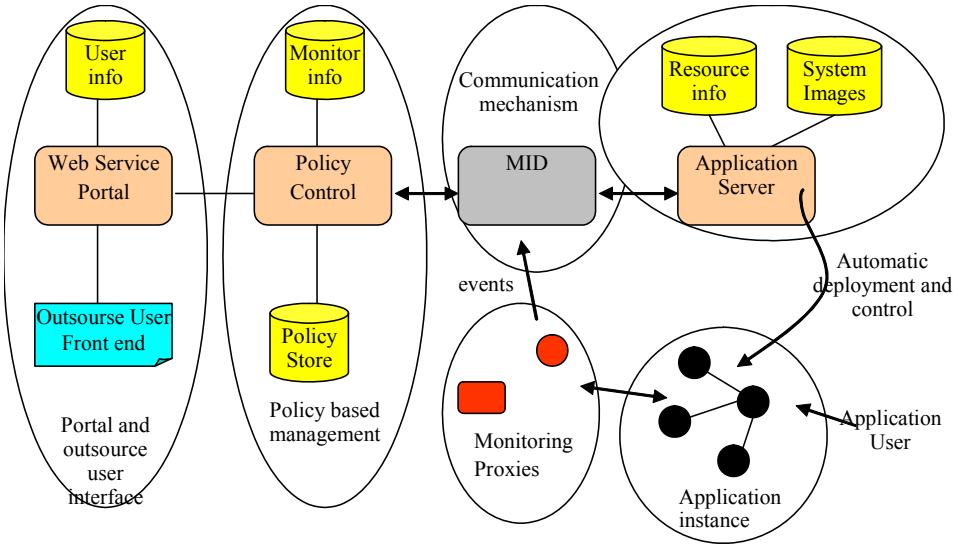

Figure 1 Out Sourced System overview

2. System Overview

The prototype's main function is to demonstrate the technical feasibility of customer control over their business process. It can be divided into five major parts, as shown in Figure 1. We assume that the entire system is distributed, i.e. the end user or the application provider can access (monitor and control) their hosted applications over the internet.

Portal and outsource user Interfaces

This is the user/owner front end of the outsourced application, where user interactions happen. This is built using web server technologies, in principle giving the users flexibility to use the system from anywhere on the Internet. The Web Server is based on JAX-RPC (Java API for XML – Remote Procedure Call), XML(eXtensible Markup Language) and SOAP (Simple Object Access Protocol). JAX-RPC allows us to build an interface that is accessible over the Internet using standard protocols. User Interface uses HTML (HyperText Markup Language), JSP (Java Server Pages) [1], JFreeChart (an open source charting tool) [2] and Java Servlets. JSP adds

functionality and allows the creation of dynamic interactive web pages. The JFreeChart software allows data to be translated into graphs, which can then be incorporated into the web page. The servlet technology allows for dynamic webcontent programming.

When a user targets his browser to a specific URL hosting the Web Portal, the servlet allows dynamic generation of an appropriate HTML page depending on the parameters included in the HTTP request sent from the client. Several roles were defined for different types of user, depending on these the users are allowed to perform different activity. For example the users with the role Admin, are allowed to monitor application, control them and can also create new users. The Super Users are allowed to monitor and control application and the Users can only monitor application. The Super Admin can initiate the creation of the application.

Communication mechanism

The MID [3] (Management Information Distribution System) is acting as a simple publish/subscribe message bus. The system is managed using policies and events, which are distributed using MID. MID is capable of distributing XML events using a number of means, e.g. using socket communication, RMI, SOAP etc. The event consumer register their interest with MID by sending a policy.

Management Infrastructure

The system is managed using an approach based on policies and events. Our policy-based approach can be viewed as providing the linkage between loosely coupled components or subsystems. We are looking for a unified view of management that will give a consistency of overall design while imposing as few constraints as possible on the designers of individual systems. This is necessary if products from a variety of sources are to be integrated effectively.

A policy [4, 5] is a rule or set of rules which determine the behaviour of a part of the system. . The high level semantics are that a policy is interpreted by a subject and gives it some rules relating to carrying out actions on one or more targets, possibly dependent on some conditions. A policy syntax based on XML Schema allows considerable extensibility within an overall framework which is sufficient for generic policy handling in a large distributed system. Policy based approach gives us a flexible management of the system. Different sets of policies are written for different sets of customers, users and systems. The event is a trigger for a policy. An event may be generated either directly by an XML-aware component or by a special monitoring component which obtains information using some other mechanism (e.g. SNMP) and translates it into XML.

The main task of the Management Infrastructure is to co-ordinate the monitoring and the control of the application. All the messages to and from different parts of the systems are defined in terms of XML events. The application server gets two kinds of events: Application related events and User related events. The Application related events might contain resource monitoring information (e.g. Monitor_info). In this case the information is stored in a database for later analysis. The other Application related events include those indicating node failure, resource changed etc. The User events need to be validated against the role/activity database before policies can be executed. Policies, set by users with the appropriate capabilities, are retrieved from a policy store. Users can also retrieve monitoring information from the monitor info store. Users can also send requests to change the application performance. The policies and events that are used to manage the system are divided in four classes.

- Access Control: These policies are triggered by events generated as the user accesses the system or requests a controlled action (e.g. initial login, change of resources). Some of these policies are defined by the hosting Super Admin and, within the constraints that these set, others can be defined by the Admin user of the outsourced application.
- Monitoring: These policies and events are used when the user monitors application performance and computing resources. There are two kinds of performance view available to users. The system view will allow the user to drill down to an individual component of the application infrastructure (e.g. a server) and monitor its usage (e.g. free memory, free disk space etc.). The business view allows the user to monitor the overall performance of his application. We anticipate that each user will be able to define a library of business performance views for his applications which can then be selected and customised through the use of policies.
- Resource Assignment: These policies and events are used when the resources are allocated or de-allocated to the application or when new application is created.
- System Administration: System may generate several events which are useful for system administration. The users (e.g. admin user of the outsourced application) need to listen to the event and as the events come to the system, they are forwarded to the user to take further action. For example if an overload event comes

to the system, the user may want to improve the performance by adding extra servers to the application. System events can also trigger self-healing policies for the system, for example a node failure event will trigger a policy which will replace the dead node for the system.

Application deployment and control

This actually looks after the individual application, it sends monitoring information to the application servers and controls the applications according to user preferences and actions (e.g. add new resources. Automated provisioning and configuration of resources is an essential element of our system. For the prototype we have chosen a load balanced web service as the application. This consists of several working web servers and a load balancer. The initial deployment can be done via the portal, using scripts written in Perl. We have a database containing all the resources that can be used to build the application.

- We have a pool of machines that can be used to build the application. At the initial stage these systems are idle, i.e. the systems are powered on and can be contacted and rebooted.
- PXE (Pre-boot Execution Environment) [6] and DHCP are used to boot the machines. DHCP is used for IP number allocation for the resources. The PXE capability gives PC manufacturers and system administrators a way to boot a system without having to know anything about its operating environment. A set of scripts (written in Perl) is used to create and control the resources.
- The System Image database contains the images of standard application resources, e.g. the web servers. System images and web content are loaded as part of the system build.
- For the load balancer we are using Linux Virtual Server [7]. When a load balancer is up and running, it starts several processes to gather standard resource (monitoring) information and send it to the communication bus (the MID).

Monitoring

To allow users to access performance data, there needs to be a mechanism for monitoring the performance of the resources associated with a specific application instance and also some way of relating the resource performance to that of the application. For individual resource monitoring, we have only looked in the prototype at monitoring of a Linux based system. Most of the system related monitoring information came from the /proc file system, which acts as an interface to internal data structures in the kernel. For monitoring application performance we use an approach based on a client proxy running at the front end of the application. This approach gives a flexible way of introducing customer-specific monitoring of performance closely related to end user experience

3. How it works

At the initial stage the Super Admin (role of the user) can send a request (via the web portal) to configure a new application. This application will contain one load balancer and several web server nodes. The system first checks the resource database for free resources. If free resources are available, a script is executed to build the application, i.e. configure the load balancer and web server nodes. Once the application is up and running, the client administrator (Admin) can log into the system with the details he receives in the email. After logging in, the User Admin can create some new users, and alter default policies to change monitoring views. When a new user is created such as a Super User, they can log in and monitor the application. The Super User can then decide to alter the performance by pressing one of the control buttons. This will eventually lead to more resources being added or resources being removed. For the prototype, we have used blade servers with Linux based operating system. Blades can provide high density of servers with fine granularity – appropriate for virtualisation and dynamic management of utilisation.

4. System Testing

The functionality of the system has been tested in a black box style. These tests were carried out to make sure that the system functioned correctly when matched against the customer requirements. Performance tests were then carried out on the front-end web portal. This was to establish how many concurrent users could be logged on in a specific instance. These tests are vital to show how the system performed under high stress conditions.

Application creation was tested which enabled the customer's application to be hosted. The details required were the users credentials and the starting number of resources required. When the system was fully loaded, an email was generated and sent to the client successfully. The client was then able to log into the created application. It was found that to build an application resource from its idle sate to running state takes about 3 minutes. This

involves loading operating system and content and rebooting the system. But this can easily be speeded up by having a pool of machine with operating system pre loaded. With this approach a new resource can be introduced within 30 seconds. The other tests were, role based access to see different users have different authority to manage the hosted application, resource alteration to see whether it is possible to add or remove resources. The altering of a policy was tested within this application. The user selected the appropriate choices presented through the policy editing function. This resulted in a policy in the policy management system being altered. This alteration affected the state of the system and this was reflected successfully in the clients monitoring view. For the application, we have decided to have at least one loadbalancer and two servers running for the application. Hence we did a test to fail/crash one of the two servers, and found that the system recovers it self by adding a new server to the application.

5. Conclusion

Outsourcing is a popular business strategy that allows companies to use external expertise to deal with non-core parts of their business such as IT management. In today's highly competitive marketplace, more businesses are considering outsourcing as a way to reduce costs, streamline internal operation and boost performance. In this paper we have discussed a way to manage out sourced IT application infrastructure using policies and events. Our approach allows the application provider to monitor and control the performance of their application which is hosted in a third party domain. As this is only a prototype, we have only looked at monitoring Linux based system. From the functional tests it has been established that the system works correctly and performs well. It was found that the web service can handle about 80 simultaneous request, which is satisfactory for management and control. The controlling (e.g. starting or shutting down servers) has also done in the Linux domain but similar actions can also be taken in the Windows domain.

References.

[1] Java Server Pages http://java.sun.com/products/jsp/

[2] JFreeChart http://www.jfree.org/jfreechart/index.html

[3] Lionel S., Prnjat O., Liabotis I., Olukemi T., Ching A., Fisher M., Mckee P., Georgalas N., Yoshii H., Active Robust Resource Management in Cluster Computing Using Policies, Journal of Network and Systems Management (Special Issue on "Policy Based Management of Networks and Services"), Volume 11, Issue 3, September 2003, Pages: 329 – 350, ISSN: 1064-7570, Plenum Press, New York, NY, USA

[4] O. Prnjat, I. Liabotis, T. Olukemi, L. Sacks, M. Fisher, P. McKee, K. Carlberg, G. Martinez , "Policy-Based Management for ALAN-Enabled Networks",3rd International Workshop on Policies for Distributed Systems and Networks (POLICY'02) , June 05 - 07, 2002, Monterey, California

[5] Damianou N., Dulay N., Lupu E., Sloman M., Ponder: A Language for Specifying Security and Management Policies for Distributed Systems, Imperial College Research Report DoC 2001, Jan. 2000

[6] PXE information http://clic.mandrakesoft.com/documentation/pxe/

[7] Linux Virtual Server:- http://www.linuxvirtualserver.org