A Personal Portal for Individual Resource Control System

Yih-Jiun Lee, Chien-Kuo Technology University, Taiwan; E-mail: yjlee@ctu.edu.tw
Kai-Wen Lien, Chien-Kuo Technology University, Taiwan

INTRODUCTION

A portal aims to make the user more efficient and convenient by providing an entrance to all other coordinators. Variant portals may apply for different purposes, for instance, Yahoo is a portal for all Internet users to connect to sites which are grouped into categories. Except the Internet portal, users might need another kind of portal for their life.

In the real world, users might have different computing devices for use on different occasions, such as a personal computer, a laptop, a Personal Digital Assistant (PDA), or even a workstation with privileged access.

However, users tend to use only one device at a time, for instance a personal computer at home, a workstation accessed on an account provided by an employer, or a PDA to send a file to a co-worker when travelling. In this context, two problems are illustrated: first, when one device is being used, the other (or all others) might be idle; secondly, the user might have to maintain several copies of objects at different places, in order to access them everywhere, causing serious consistency and maintenance problems. IndiGrid, which stands for 'Grid system for Individuals', is designed for a single user solution to those problems. It also acts as a personal portal. By means of IndiGrid, a user is able to access every device he is privileged to access, and perform some authorized actions without bothering about security issues or file version synchronization. IndiGrid is based on web-services technologies and uses communication port 80 for messaging. It also enables asynchronous message transmission to improve transmission and performance. IndiGrid is a personal portal to enable global sharing and resource control to provide efficiency and convenience in the computing world for individuals.

RELATED RESEARCH

1. Grid Computing

Grid computing (Foster 2005) tries to solve the resource sharing problem of boundary crossing. It also refers to the management and integration of distributed resources, conceptual and physical resources, and services across globalized, large-scale, multiple administrative domains (Foster & Kesselman, 2003). Since participants might belong to or follow different rules, as far as "trusting" each other and "being trusted" is concerned, the participants can form a virtual organization, in which every member is mutually trusted. Therefore, they can share resources with each other. Grid computing can be classified as comprising a computational grid or a data grid. However, no matter what the grid system is, it must provide a multi-user and multi-resource enabled environment.

2. Peer-to-Peer

As computational performance on personal computers and network communication infrastructures has improved, Peer-to-Peer (P2P) computing has become another communication model for the environment in which devices (computers, servers, and all other computation devices) link to each other directly. All the devices are "peers". The computers in P2P computing can be both clients and servers, unlike as in server-centric computing (Online, 2006) or asymmetric client-server systems (Foster & Kesselman, 2003). All the participants share their resources (mostly files) equally. The main aim of P2P is for resources to be provided and consumed by each peer, in contrast to the client-server system, in which only servers provide resources. P2P is also comparatively more robust and reliable.

3. WSGrid

WSGrid (Henderson, 2004) is a web services based grid computing environment, proposed by Professor Peter Henderson of the University of Southampton, UK. A WSGrid based grid environment is composed of nodes, where a set of WSGrid web-services is installed. A node is also known as a host, where computation occurs and resources are provided. The virtual organization in WSGrid is constructed around a combination of nodes and users. Unlike most virtual organization solutions, which often have one or more centralized components, the virtual organization of WSGrid's distributed idea is full distributed. Each node in WSGrid has equal position. However, in comparison to Peer-to-Peer computing, WSGrid does provide more controllability and security by allowing users to set up different privileges for different remote accounts.

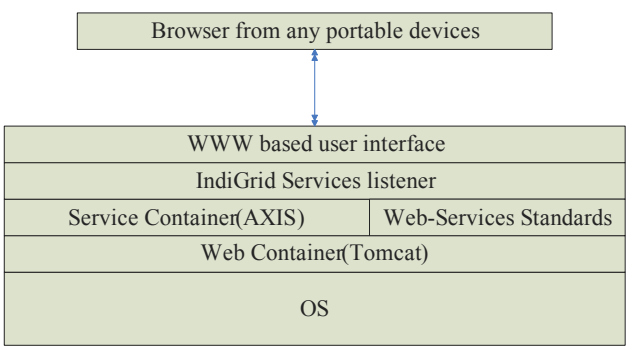
CURRENT RESEARCH STATUS (SYSTEM INTRODUCTION AND ARCHITECTURE)

1. The Architecture

In order to provide "light-weight" middleware, reduce the risk from malicious users, and maintain network security, IndiGrid's design is based on web-services technologies. All services (the components of IndiGrid) can be hot-plugged and unplugged, so the performance will not be affected. Another benefit of using web-services technologies is that web-technologies and open resources on webs are maturely developed. In addition, configuring a web container is not too difficult, even for a non-professional user. The threshold is low. From the point of view of network security, communication port 80 is turned on as a default. As regards user friendliness convenience, efficiency and performance, web-service architecture is very suited to grid systems. IndiGrid was originally designed for individuals, so it is assumed that only the owner has the privilege of access to the computers in the group. However, through a delegation process, temporary access rights are also provided.

IndiGrid is designed to be a single user grid middleware to provide grid users with a convenient access to their workspaces and resource. Since it is for individual, it is assumed that everyone has security tokens (i.e. username and password, certificates) to be authenticated by their spaces. Furthermore, when a device is connecting to the Internet, remote connection should be restricted to only certain areas. So does IndiGrid. Only opened areas (configured in the configuration)

Figure 1. Middleware structure



are allowed to be remotely accessed. Through the process of authentication and authorization, the security of a host is basically remained.

2. The Functionalities

By referring to the requirements of the Grid (Foster 2005), a grid middleware should provide the following functionalities: discovering and configuring resources, moving data, monitoring and controlling components, and manage credentials. IndiGrid is composed of set web-services for the above functionalities of resource sharing and load balancing.

a. FileService

FileService is used to move objects from one location to another. IndiGrid follows the idea of WSGrid to allow (restrict) remote access to certain locations to prevent possible attack from malicious users. Thus, the owner can move all objects around. And the moved objects can be marked to be removed after use.

b. JobService

JobService is a submission service to allow an owner to submit a job (process, task) to another device. This service aims to balance the computation load and uses some resources only located on a certain computer. The user sends a job along with the job description, which states the conditions for the job (such as delete after execution or result returning to a specific place).

c. LoggingService

LoggingService is used to record the status of the execution and servers. It can also be used as a task-status-query service.

d. MyFavoriteService

Users commonly keep logs of their favourite websites on their computers. However, keeping all web favourites consistent (on different computers) is difficult. MyFavorite- Service is the service to return the favourite web sites as a linkable web page, so users can access them anywhere without bothering synchronization.

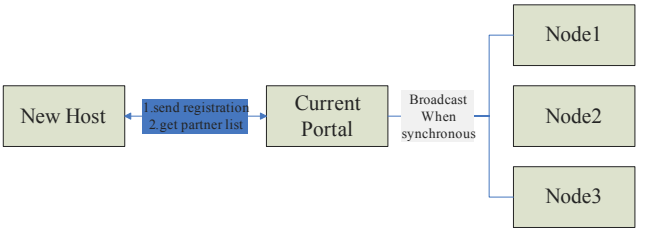
e. DelegationService

DelegationService is an important and existing service to allow the user to "issue" a temporary passport to someone trustworthy. This service follows the idea of GateService (Lee, 2005). This service in IndiGrid is for sharing. For instance, a man needs to share his file(s) to group members. The delegators must specify the rules of delegatee in a simple and clear XML file.

f. NotificationService and StatusRequestService

NotificationService and StatusRequestService are working in pair of co-existent services. When a host joins in the group, the node must firstly invoke the NotificationService (as saying Hello) of any other node (like a registry) to introduce

Figure 2. The registration sequence of IndiGrid



the node itself, ie the WSDL for its services and its capability. The notified node forwards the registration information to all known partners, which then say 'hello' to the new member. Thus, nodes can know each other. Then StatusRequestService can respond any information regarding the status of nodes. They will be illustrated in the next section.

THE FRAMEWORK OF HEALING (NODE STATUS CONTROL)

As a full distributed environment, to aware the availability and capability of nodes is very important. In the earlier version, GateService, the submission can only queue for its execution. However, it is possible that the node might be currently unavailable, but it still appears as online. Therefore, IndiGrid uses a partner list to keep the status of partners.

Partner list is an XML format file to keep the current status. The information of partner list is coming using the NotificationService. Three kinds of NotificationService are currently available. When an environment is built, a Root must be chosen. The root can be specified by the owner or be elected using the token ring method. However, only the nodes who can be servers are candidates.

First, when a node (which is not currently on the partner list) joins in the group, the node will say Hello to a node (acting as a registry). Then the notified node (root) can update its partner list and reply the message with the partner list. On occasion, the root might forward the new partner to all other known nodes.

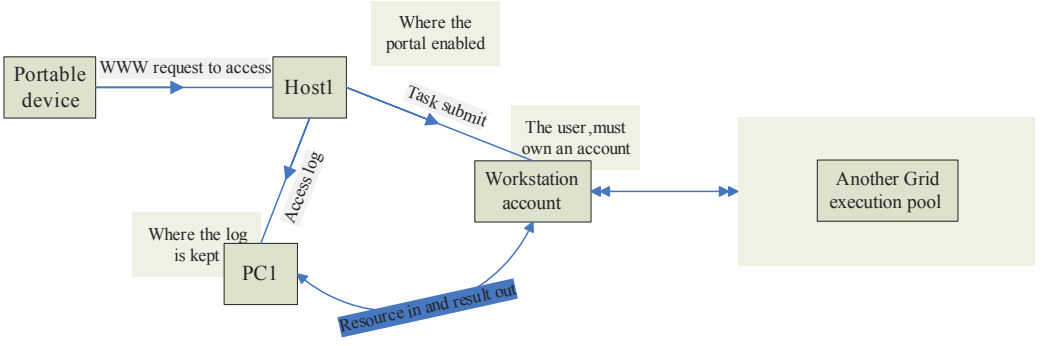
Second, when a node is busy (the job queue is full or nearly full), it can send an "unavailable" message through NotificationService. The corresponding process is to move the node to the end of the partner list.

Finally, when the "unavailable" node is back online, it can also notify its partner about its return.

All these three kinds of notification are using push methods to automatically process.

StatusRequestService is another side of story. It is designed for pull method. For instance, when a node sends a request but does not get the response back in

Figure 3. IndiGrid



reasonable time. The user might want to initiative a StatusRequestService request. When a node initiates the request, a timer is also enabled. When time out happens, a node with no response is moved to "unavailable" state. In order to improve the performance of network, broadcast of StatusRequestService request is not recommended. Other status synchronization methods are still in progress.

CONCLUSION AND FUTURE WORK

At the time being, services 'a' to 'e' listed above have been successfully implemented and tested, but NotificationService and PartnerService are only at version 1. To automatically manage the status of each partner (node) is not too difficult. There are two kinds of notification transmission methods to be provided. The problem is that if the synchronization is processed too frequently, the system performance and network bandwidth might be affected. However, if the synchronization process is seldom performed, the system status is hard to maintain. Thus, both push and pull methods of synchronization are provided. For active information providers, the push method can be used for notification at initiation, busy time and off line. For accidentally system crash or expected system busy, the pull method should be able to help. Besides, a novel equation of automatically reconfiguration is being studied and will be proposed to solve the problem.

IndiGrid is expected to be a fully functioning grid system for individuals. By using web services, it is a portal, able to connect every computer or computing device for the owner. With IndiGrid, users do not need to deal with multiple inconsistent files or wait for a busy device to return the computing result. They can share load and resources among different computers and use them just like at home.

REFERENCES

Foster, I., & Kesselman, C. (2003). The grid 2: Blueprint for a new computing infrastructure. Morgan Kaufmann.

Online (2006) Definition of distributed computing. http://www.microsoft.com/net/basics /glossary.asp

Henderson P., (2004). WSGrid. http://www.ecs.soton.ac.uk/~ph

Lee, Yih-Jiun (2005) A security solution for web-services based virtual organizations in Proceedings of the Information Resources Management Association International Conference (IRMA 2005): Managing Modern Organizations with Information Technology, San Diego, USA, May 2005.

Lee, Yih-Jiun, (2006) A Distributed Grid Service Broker for Web-Services Based Grid Applications, The 10th WMSCI 2006, July 16-19, 2006 Orlando, Florida, USA

Foster, Ian (2005) Globus Toolkit Version 4: Software for Service-Oriented Systems. IFIP International Conference on Network and Parallel Computing, Springer-Verlag LNCS 3779, pp 2-13, 2006.www.globus.org/alliance/publications/papers/IFIP-2005.pdf