Int J Adv Manuf Technol (2001) 18:217–234

The International Journal of
Advanced
Manufacturing
Technology

Enterprise Virtualisation: Concept, Methodology, and Implementation

L. Zhang[1], S. C. F. Chan[2], V. T. Y. Ng[2] and K. M. Yu[3]

[1]Department of Computer Science and Technology, Tsinghua University, Beijing, China; [2]Department of Computing, The Hong Kong Polytechnic University, Hong Kong; and [3]Department of Manufacturing Engineering, The Hong Kong Polytechnic University, Hong Kong

Enterprise Integration has been widely studied in the last decade and many of the results have been successfully applied in industry. Virtual enterprise is a newer research field to which researchers are paying more and more attention. However, from the viewpoint of enterprise development, enterprise integration and virtual enterprise are very different in both architecture and concept. In practice, enterprise integration is built in real space and virtual enterprise is built in virtual space. Hence, in the development of enterprises, an intermediate stage between enterprise integration and virtual enterprise must be studied. In this paper, a new concept, enterprise virtualisation is developed. Enterprise virtualisation is the link between enterprise integration and virtual enterprise. This paper discusses the basic architecture, basic activities and global model of enterprise virtualisation and their key elements. The methodologies of enterprise virtualisation are illustrated which involve resource virtualisation, management virtualisation and working environment virtualisation. A prototype of a virtualised enterprise has been successfully developed by the Department of Computing at The Hong Kong Polytechnic University. Through this prototype, Internet-based design and manufacturing has been implemented by resource sharing and reconfiguration.

Keywords: Enterprise integration; Enterprise virutalisation; Resource management; Resource sharing and reconfiguration; Virtual enterprise; Virtual space

1. Introduction

An enterprise is an economic organisation whose objective is to produce certain products. The two basic characteristics of the organisational structure of a traditional enterprise are as follows:

1. An enterprise is generally constituted by three key elements: people, resources, and management. People include managers, designers, engineers, and operators. Resources consist of software systems, hardware devices, tools, product data, and so on. Management of an enterprise includes resource management, process management, personnel management, and project management.
2. A traditional enterprise is constrained by internal and external boundaries. The internal boundaries, such as operation system boundaries, language boundaries, and information boundaries, exist between various branches and components of the enterprise that produce information expression disagreements and difficulties in information exchange. The external boundaries include spatial location boundaries that limit enterprise activity ranges and cause difficulties in resource sharing.

Along with the development of computer networks and information technology, and rapid changes of the market and user requirements, the trend is for manufacturing to become a global undertaking. The production mode, organisational structure and activity range of traditional enterprises faces severe challenges. Many workers have paid close attention to these problems in the past decade and much progress has been made [1–4].

Enterprise integration (EI) and computer integrated manufacturing (CIM) have been widely studied in the last decade and many of the results have been successfully applied in industry. The main purpose of EI is to solve the problems of information exchange and sharing caused by internal boundaries in traditional enterprises. For example, a popular technique of EI is to use the STEP standard [5–7] to define the global product data model over all product life cycles so that product data can be exchanged and shared among different operation platforms and different product development phases. According to the definition by CIMOSA [8], Enterprise Integration provides the right information at the right place and at the right time and thereby enables communication between people, machines,

Correspondence and offprint requests to: L. Zhang, National Research Center for CAD Support Software, Department of Computer Science and Technology, Tsinghua University, Beijing, China. E-mail: lizhang@public.fhnet.cn.net

and computers, and their efficient cooperation and coordination. In general, EI focuses on information expression inside traditional enterprises.

Virtual enterprise (VE) is a newer research field to which workers are paying more and more attention [9,10]. Work is mainly concentrated on the methodologies for building the VEs [11–13]. An authoritative organisation on the VE is the NIIIP (The National Industrial Information Infrastructure Protocols consortium), among whose important achievements on VE is the definition of the reference architecture [14–16]. According to NIIIP, a virtual enterprise is a temporary consortium or alliance of companies formed to share costs and skills and to exploit fast-changing opportunities. VE is a philosophy and methodology. It requires an enterprise to break through both internal and external boundaries, and be created and reconfigured with a world-wide scope in flexible and optimised mode in order to meet the requirements of the rapidly changing market.

2. The Concept of Enterprise Virtualisation

From the viewpoint of enterprise development, enterprise integration and virtual enterprise are very different in both architecture and conception. In practice, enterprise integration is built in real space and virtual enterprise is built in virtual space. Hence, in the development of enterprises, an intermediate stage between enterprise integration and virtual enterprise must be studied. In this paper, a new concept, enterprise virtualisation is developed. Enterprise virtualisation is the link between enterprise integration and virtual enterprise. On the one hand, it is a mapping of enterprise integration in real space. On the other hand, it is a base and condition for building a temporary enterprise consortium (virtual enterprise). Before describing the concept of enterprise virtualisation, two terms must be defined:

Virtual space (VS): VS presents an unbounded information process field which is built on computer networks (Intranet/Internet/Extranet) and corresponding standard communication protocols (TCP/IP, HTTP, HTMP, VRML, CORBA, etc.). In the VS, various information and data can be exchanged, processed, converted, stored, and displayed in security.

Virtualisation: Virtualisation is a mapping process through which the inner structure and implementation of an entity in real space are hidden, and only its outer functional features are mapped to the VS so that this entity becomes visible, operable, manageable, and portable in the VS.

Based on the above terms, we can describe the concept of enterprise virtualisation.

Enterprise virtualisation is a set of mapping processes, called virtualisation, which map key elements of an enterprise from limited real space to unbounded virtual space. The purpose of enterprise virtualisation is to make the enterprise visible, operable, manageable, and re-configurable. An enterprise which has been virtualised is called a virtualised enterprise (VIE). Here, "visible" means that the resources, processes, and product data in the VIE can be queried and displayed in some multimedia mode in the VS. "Operable" means that the functions of the resource in the VIE can be accessed in the VS. "Manageable" means that the status and parameters of the resources in the VIE can be updated, and resource use authority can be reassigned. "Re-configurable" means that organisational structure in the VIE can be rearranged, for example, building a new virtual workspace according to the demands of a project.

Figure 1 shows the relationships among traditional enterprises, integrated enterprise, VIE and VE. From the above discussion, we can conclude that there are three stages during the process of enterprise development. One stage is the enterprise integration. In this stage, the key issue is how to break down the internal boundaries in the traditional enterprise. The second or intermediate stage, is VIE. The main objective in this stage is how to map a traditional enterprise or integrated enterprise from the real space to the VS. VE is a higher stage of enterprise development. The necessary condition to create the VE is enterprise virtualisation. The purpose of studying VE is to create and define a set of mechanisms and protocols to build temporary alliance of the VIE.

3. The Characteristics of a Virtualised Enterprise

3.1 The Architecture of a Virtualised Enterprise

Figure 2 describes the architecture of a virtualised enterprise. A virtualised enterprise is built in virtual space and it divides the virtual space into three subspaces:

1. *Virtualised resource* subspace which is the set of virtualised resources registered in the meta-resource data repository (MRDR).
2. *Virtual working environment* subspace which is the set of admin-spaces and working-spaces.
3. *Virtualised management* subspace that includes the resource management system (RMS) and the MRDR.

These three subspaces, which are also called key elements of the VIE, correspond to the three key elements of a traditional enterprise.

3.2 Basic Activities in a Virtualised Enterprise

Analysing the basic activities in a VIE will contribute to the comprehension of the functional requirements of the various components in the VIE and the relations among them. The activities that are likely to take place in a VIE can be classified into the following five types:

1. *Resource management (RM) activity*: Interactions between the administrators and the RMS, such as registering new resources, querying meta resource data, maintaining resources, supervising resources, etc.
2. *Working-space configuration (WSC) activity*: Interactions between the users and the RMS. For example, a users signs in the RMS by username and password. The RMS sets up

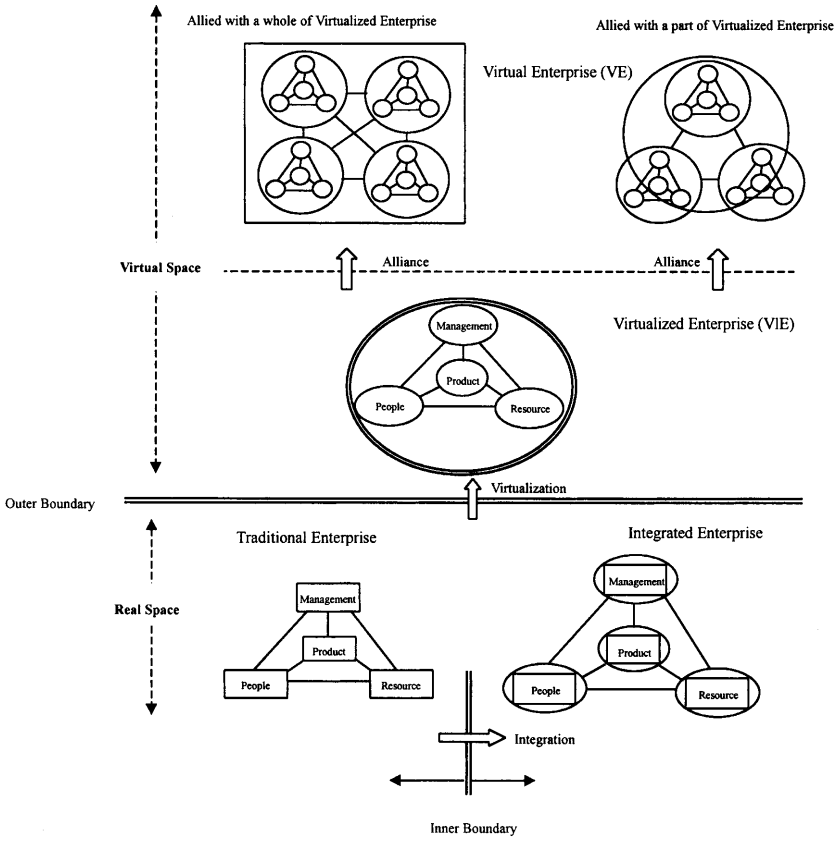

Fig. 1. Relationships among traditional enterprise integrated enterprise, virtualised enterprise, and virtual enterprise.

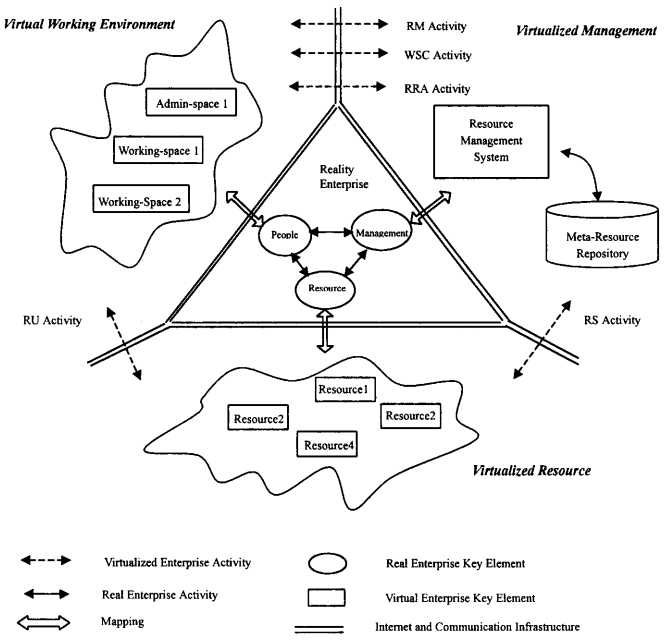

Fig. 2. The architecture of a virtualised enterprise.

the use authority and offers an initial WS. The user enters the initial WS and subscribes virtualised resource from the meta-resource data repository. When the user enters WS next time, he/she will obtain a special working platform which contains the RGUIs of the ordered resources.

3. *Resource request and assignment (RRA) activity*: Interactions between the users and the RMS. When users want to use a resource through the resource graphic user interface (RGUI), they must apply from the RMS first. The RMS will make a decision in terms of the current state of the resource. The decision can be one of the following: approval, rejection, or asking the applicant to wait. If an application is approved, the RMS will notify the applicant and send the operation-handle of the resource server.
4. *Resource use (RU) activity*: These are interactions between the users and the used resources. When users obtain the authority for a resource, they can transparently operate the resource through the RGUI in the WS in client/server mode. The user first launches the resource server by the RGUI. Then the server checks the user's authority of access and operation. Next, the server begins to provide services. Finally, the server returns service results to the user.
5. *Resource supervision (RS) activity*: These are the collaborative operations between virtualised resources and the RMS. For instance, when resource states are changed, the resource server will automatically send a message to the RMS. On

receiving the message, the RMS processes it and launches another resource management event. The RMS can modify the access and operation rules of the resource so as to control the resource server behaviour.

In general, a business process in the VIE is a combination of several basic activities as discussed above. For example, when users want to operate a resource server, they put forward an application to the RMS through *resource request and assignment* activity. If the application is approved, the RMS will launch a *resource supervision* activity to notify that resource server. Finally, users directly interact with the resource server through a *resource use* activity. If the resource states are changed, a *resource supervision* activity will be launched again to send a message to the RMS. The process above is shown in Fig. 3(*a*).

Sometimes, some activities will be executed repeatedly as part of a business process. For instance, if a user wants to use a NC progamming system to generate NC machining tool paths, this process will involve activity RRA, RU, and RS. However, if the format of the input geometry data file cannot be processed directly by this resource server, it will automatically invoke another virtualised resource for data format translation. In this case, activity RRA, RU, and RS will be launched once again, as shown in Fig. 3(*b*).

In some situations, more than one resource is required for one business process. For example, a user wants to transfer a file from resource *A* to resource *B*, the process can be performed only when the user obtains use authority for both resource *A* and resource *B* at the same time. This situation is shown in Fig. 3(*c*).

4. The Enterprise Virtualisation Methodology

The key problem for enterprise virtualisation is how to map an enterprise from real space to the VS. We begin our research from three key elements of the enterprise.

4.1 Resource Virtualisation

Resource virtualisation is a mapping process through which some functions of a resource are mapped to the VS so that the resource becomes visible, operable, manageable, and portable in the VS. Resource virtualisation can be divided into three subprocesses: virtualisation wrapping, resource graphic user interface (RGUI) and meta-resource data mapping, as illustrated in Fig. 4.

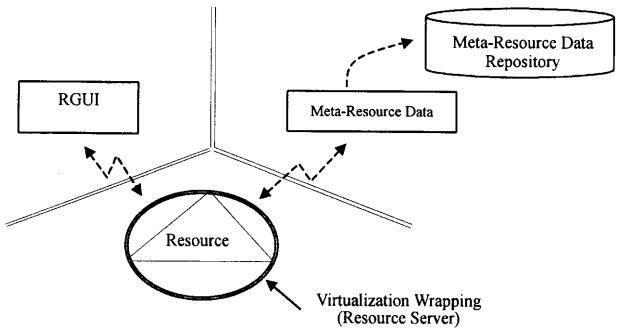

Fig. 4. Resource virtualisation.

4.1.1 Virtualisation Wrapping

The purpose of resource virtualisation wrapping is to make a resource operable and manageable in the VS. It involves two kinds of wrapping technology:

Object-Based or Object-Oriented Wrapping. Generally, the function interfaces provided by various resources may appear in different forms, such as API functions, C++ classees, DOS commands, and so on. In order to make the wrapped resource sharable and interoperable in the VS, a unified function interface should be defined first so that users can apply a common invocation to operate these different resource functions. CORBA-based wrapping technology can meet these requirements. OMG/CORBA 2.0 [17] provides a standard interface definition language (IDL) in which the unified function interfaces can be defined. OMG/CORBA 2.0 also defines a standard ORB/IIOP communication protocol to support object invocation under different computing environments.

Agent-Based Wrapping. Object-oriented wrapping makes the resource a passive object server only. That is to say, a wrapped resource can only accept invocation from users and provide services to them. It is not provided with the capabilities of behaviour autonomy and collaborative works that are necessary for resource management in the VS. "Behaviour autonomy" means that when a wrapped resource is invoked and operated by a user, it can automatically perform security checking, such as checking the access and operation authority. "Collaborative work" means that when the stages of a resource are changed, it can automatically report to the resource management systems. These capabilities can be implemented through agent-based wrapping technology.

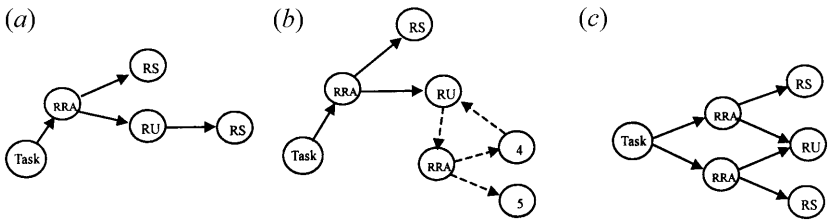

Fig. 3. The basic activity types.

4.1.2 Resource Graphic User Interface (RGUI)

Virtualisation wrapping only implements an operable and manageable object server that is run at the resource server end. When users want to use a resource, they must program a special client program to launch that object server and to operate it. In order to turn the resource into a portable and selectable tool, an RGUI should be developed at the same time as virtualisation wrapping. An RGUI must be a platform-independent software package, which can be subscribed and downloaded to a user platform and can be operated transparently by users. The relations among virtualisation wrapping, RGUI, and a wrapped resource are shown in Fig. 4.

4.1.3 Meta-Resource Data Mapping

From the viewpoint of resource management, it is impossible and unnecessary to store all resource data in the database. Certain resource feature data are needed for resource management, such as the physical location of the resource, the owner of the resource, the role played by the resource, resource states, their usage rules, etc. These resource feature data are called meta-resource data [18–20]. The mapping of meta-resource data is a process for obtaining meta-resource data and storing them in the database. Meta-resource data mapping includes three steps as follows:

1. Defining the meta-resource data model (MRDM). The MRDM should cover all common features of the various resources and meet the function requirements of the resource assignment, use, and management. The MRDM will be discussed in Section 6.
2. Building the Meta-Resource Data Repository (MRDR) according to the MRDM. The MRDR should be visible in the VS through the resource management system.
3. Registering meta-resource data. When a resource must be added to a VIE, the meta-resource data of this resource should be extracted in terms of the MRDM and stored in the MRDR.

4.2 Working Environment Mapping

In an actual enterprise, people can operate resources directly and watch the results of the operation. People can also interact directly with management systems through some multimedia interface and accept instructions from them. However, in a VlE, people are physically separated from the resources and the management system. Hence, a virtual working environment (VWE) must be provided for the user to interact with the resources and the management system. People in a VlE are classified into administrators and users. Accordingly, a VWE is also divided into admini-space (AS) and working-space (WS).

In an AS, an administrator can perform various MRDR maintenance functions, such as registering new resources, querying and editing meta-resource data, and even deleting resources from the MRDR. The WS is a special working platform on which the user can select the required resources from the MRDR and configure his/her special WS. Each selected resource is provided with a RGUI through which the user interacts transparently with virtualised resources and the management system.

4.3 Virtualised Enterprise Management

The management in a VlE is different from that of a traditional enterprise, mainly in the three following ways:

1. In a traditional enterprise, the resource configuration and organisation structures are relatively fixed. However, in a VlE, resources can be reconfigured according to user requirements.
2. The management in a traditional enterprise focuses on the optimisation of the product process during the whole product life cycle. However, the management in a VlE concentrates mainly on the optimisation of resource configuration.
3. In a traditional enterprise, management objects such as people, resources, and processes are many and varied. In a VlE, the types of manageable object are simpler and VlE can be classified into five basic types, which will be discussed in next section.

5. Function Requirement Analysis of Virtualised Enterprise

On the basis of the discussion of the architecture and basic activity types in a VlE, the common function requirements for each key element in the VlE will be analysed in detail in this section.

5.1 Function Requirement of Virtualised Resource

5.1.1 Function Definition

A virtualised resource must be provided with two types of function: resource operation functions and common management functions.

Resource Operation Functions. Different resources have different operation functions. Although the resource operation functions of the VlE are different from each other, the method of wrapping a resource should be unified so as to make the wrapped function shareable and interoperable. One of the wrapping technologies that can be used for this purpose is CORBA-based wrapping. OMG/CORBA 2.0 provides a standard IDL for defining the unified function interface for different implementations and defines the ORB/IIOP communication protocol for supporting object invocation under different environments.

Common Management Functions. Other than the resource operation functions, each virtualised resource should have the following five common management functions to support *resource supervision* activity. All these common management functions can be wrapped by the CORBA-based method:

State_Notice: Automatically sends a message to the RMS whenever resource states are changed. The format of the

message will be defined according to the meta-resource data model (see Section 6.2).

Rule_Query: Output current rules to control resource behaviour.

Rule_Modification: Modify current rules to control resource behaviour.

Access_Control: Control user's access to resource, such as accept, reject or wait.

Operation_Control: Control user's operation on resource, such as read_only, execute, write, or full_control.

5.1.2 The Function Model of Virtualised Resource

From the viewpoint of wrapping technology, the resources in an enterprise can be divided into two classes: wrappable resources and unwrappable resources. For example, software systems and databases can be wrapped directly, but machines, tools, and people cannot be wrapped. Figure 5 shows the function model of a virtualised resource. The top of the model is the resource server interface and the bottom is the wrapped resource. The event channel receives service requests from the Internet or sends information packages to the Internet. The resource server interface consists of two parts: the resource operation interface and the resource management interface. Accordingly, the information flows are also divided into two branches, one involving the resource operation services, and the second involving the other five common management services.

The resource operation interface contains four modules: *access process module*, *operation process module, translation process module,* and *interaction module.* The role of *access process* is to search for the requested object server in the server repository and to check the user's access authority. The function of *operation process* is to check the user's operating authority. If the resource is "wrappable", a t*ranslation process* is then responsible for translating the standard object invoking operations into specific operation commands of the application resource function interface. If the resource is an "unwrappable", the *interaction module* can provide an interactive window through which the user's operation commands are displayed and the resource states are sent to the management system.

The resource management interface includes two modules: the *object service control module* and the *rule management module*. The *object service control module* is responsible for checking the user's access and operation authority and sending messages to the RMS when resource states are changed. The *rule management module* allows the RMS to query and modify current resource rules in the rule repository.

5.2 Function Requirement of Virtualised Resource Management

The function model of virtualised resource management is shown in Fig. 6. Virtualised resource management consists of two parts: the resource management system (RMS) and the meta-resource data repository (MRDR). The MRDR contains a meta-resource database and an operation interface. According to the characteristic of basic activity, the RMS should be provided with four function modules: *meta-resource management, working-space management, resource request/assignment*, and *resource supervision*.

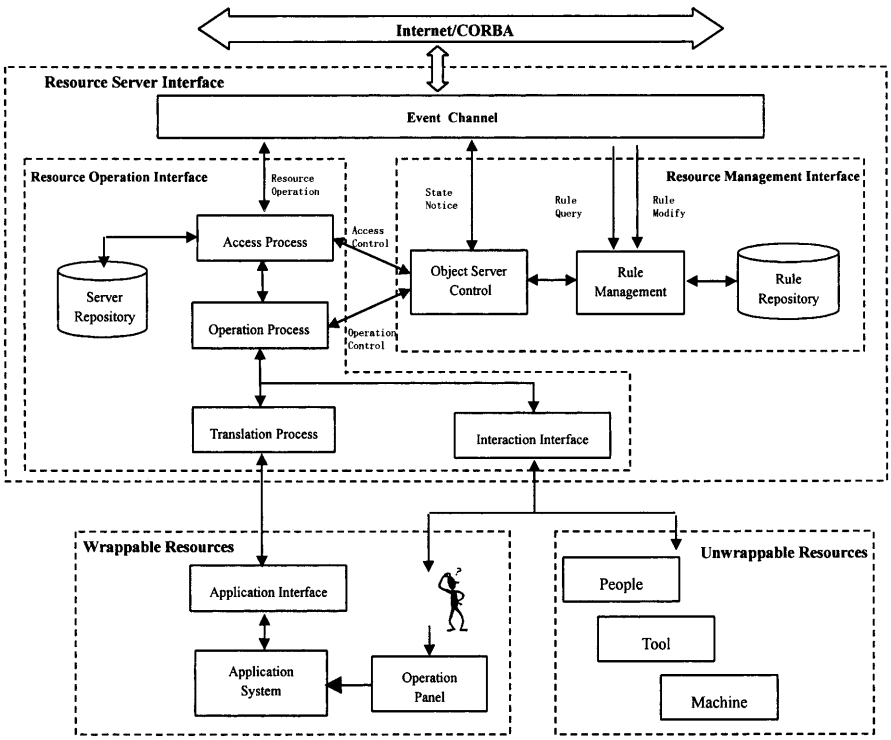

Fig. 5. The virtualised resource function model.

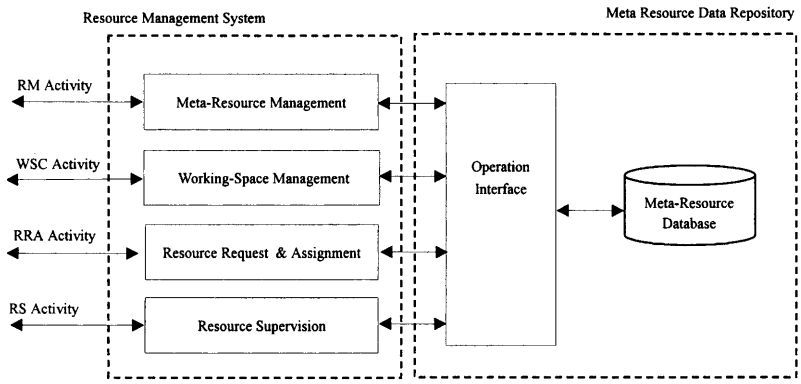

Fig. 6. The function model of the virtualised resource management.

5.2.1 Meta-Resource Management

The meta-resource management module is designed to support resource management activities in the VlE. It helps administrators to extend and maintain the meta-resource data repository. The basic functions of the meta-resource management module are as follows:

Meta-resource registry: Each virtualized resource must be registered in the MRDR. To register a resource involves two processes. First, the RGUI of the registered resource must be copied to the special file directory in the RMS. Secondly, the registration form that is designed according to the meta-resource data model should be filled up and stored in the meta-resource data repository.

MRDR maintenance: This is different from the general database maintenance. Modifying or deleting a registered resource data may affect other resources associated with it. For example, if a resource subscribed by a customised WS is deleted, the configuration of that WS should be modified accordingly.

General functions: These include the general database operation functions, such as meta resource data querying, displaying, printing, and so on.

5.2.2 Working-Space (WS) Management

The working-space management module supports the *Working-Space Configuration* activity in the VlE. It helps the user to register, configure, and operate special WSs.

WS registration: In the VIE, the WS is also a special resource. It is registered by a user, not by an administrator. Each WS belongs to only one user. When a user wants to register a customised WS, a default configuration will be provided first in which there are only some common software tools.

WS configuration: In this default WS, the user can view and subscribe to all the resources that have been registered in the MRDR. The RMS will store user subscription information so that when the user enters the same WS again, the RGUI of the subscribed resource can be downloaded to the WS.

WS download: Each WS must be downloaded to the user's computer platform. A user-specific WS consists of two parts: one contains default common software tools that can be downloaded directly from a special file directory; the other contains the RGUI of the subscribed resource. All the RGUIs are platform-independent and their execution codes are copied to the special file directly when the resource is registered. The RMS can locate these execution codes according to the user subscription information.

5.2.3 Resource Request and Assignment

Sometimes, more than one applicant may apply for the same resource at the same time. In this case, some rules would be specified for deciding which application will be granted. The process of resource assignment is as follows:

Request identification: Extract the information related to the resource applicant, applicant's authority and the requested resource from the resource request message.

Rule query: Obtain the rules related to resource assignment from the MRDR according to the ID of the requested resource.

State query: Obtain current states of the requested resource (working-state and using-state) from the MRDR.

Making decision: Possible outcomes of the decision are to accept, reject, or to ask the applicant to wait.

Request reply: No matter what decision is made, a reply message should be sent to the applicant. If the user's application is accepted, an operation handle to the requested resource server will be sent to the applicant. If the result of the decision is "wait", the information related to the applicant should be added to a waiting list.

5.2.4 Resource Supervision

The resource supervision module controls the collaborative work process between the virtualised resources and the RMS. It has the following funnction requirements:

Control for access: When a resource is assigned to a resource applicant, the RMS should send appropriate messages to notify the assigned resource.

Control for operation: Sometimes a user is permitted to operate only a limited subset of the virtualised resource, and relevant operation rules should be sent to the resource server.

Response to state change: The RMS may need to respond to state changes of the resource. For example, if the working state of a resource is changed from "occupied" to "unoccupied", the resource may be assigned to the first applicant in the waiting list.

5.3 Function Requirements of Virtual Working Environment (VWE)

The virtual working environment is a set of platform-independent software packets. Their main roles are to provide the user with a working platform on which the user can communicate with the RMS and operate the ordered resource. The VWE includes the AS and the WS. The AS is specified for the administrator to maintain the RMS. Its structure and components are fixed.

WS provides a user with a personal working platform. As shown in Fig. 7, WS should be provided with the following characteristics:

A user can download his/her private WS to his/her local computer platform at any time and location.

WS should be an extensible container. In the default configuration, only common software tools are included. A user can add the special RGUI into his/her own WS.

In the WS, there is a graphic interactive window in which the user can operate the RGUI and view the operating response.

6. Virtualised Enterprise Global Model and Meta-Resource Data Model

Figure 8 shows the virtualised enterprise global model that is defined in unified modelling language [21]. The virtualised enterprise global model constitutes four submodels: virtualised resource, virtualised management, virtual working environment and basic activity. This section will first illustrate the virtualised enterprise global model, and then discusses the meta-resource data model (MRDM). The latter is an important component of the virtualised resource.

6.1 Virtualised Enterprise Global Model

The virtualised enterprise global model describes the relations among the real enterprise, the virtualised enterprise, and the virtual enterprise. One real enterprise is mapped into only one virtualised enterprise. However, one virtualised enterprise can be allied into 0 to *N* virtual enterprises. On the other hand, one virtual enterprise consists of no less than two virtualised enterprises.

One virtualised enterprise must provide 0 to *N* virtualised environments, but one virtualised environment belongs to only one virtualised enterprise. There are people and virtual work environments (VWEs). People are classified into administrators, users, and groups. VWEs are classsified into admini-space (AS) and working-space (WS). One WS contains 0 to *N* RGUI. A person launches a basic activity by operating an AS or WS.

One VIE contains 0 to *N* registered virtualised resources. One virtualised resource is constituted of three parts: resource server, RGUI, and meta-resource data. RGUI is added into WS and user invokes resource server through the RGUI.

All the processes in the VIE consist of basic activities. An activity may be launched by a VWE or by another activity. An activity is controlled by the event server of the virtualised management system and may change the state of the resource.

The kernel of the VIE is the virtualised management system that controls all the processes and activities through a series of rule-based event services. Rule-based decision is supported by the meta-resource data repository.

6.2 Meta-Resource Data Model

Meta-resource data does not describe the resource itself. It describes only some common features of the various different resources. These common features are extracted to meet the demands of resource management. The MRDM is the set of all these common features. Figure 9 illustrates MRDM of VLE that is definedin STEP/EXPRESS Language [5]. MRDM contains four common feature classes: state, association, configuration, and rule.

6.2.1 State

The state class describes dynamic features of a resource. This state information is very important for assignment and supervision. It contains six attribute classes as follows:

1. Executing state: The *executing state* can be queried by administrators and users. The *executing state* or a resource is one of *run, suspended, stopped* OR *terminated*.

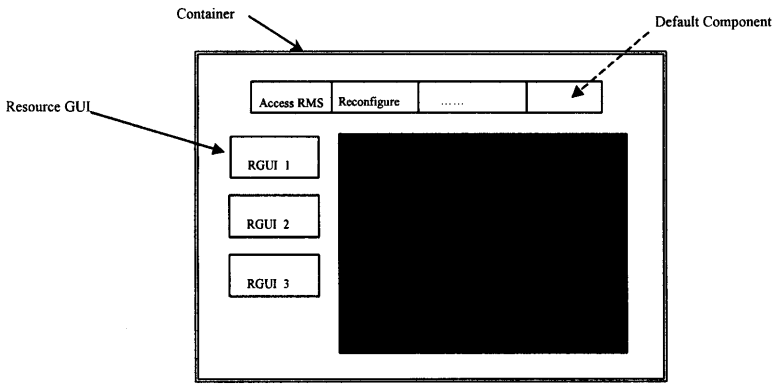

Fig. 7. The structure of working-space.

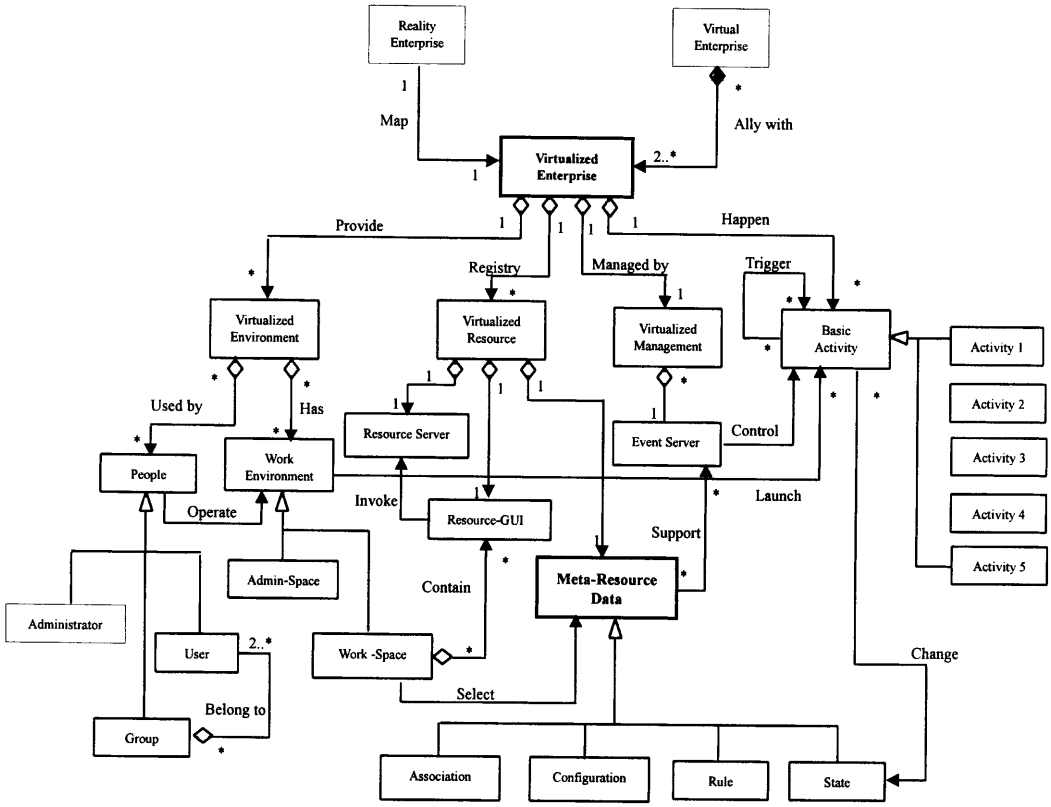

Fig. 8. The virtualised enterprise global model.

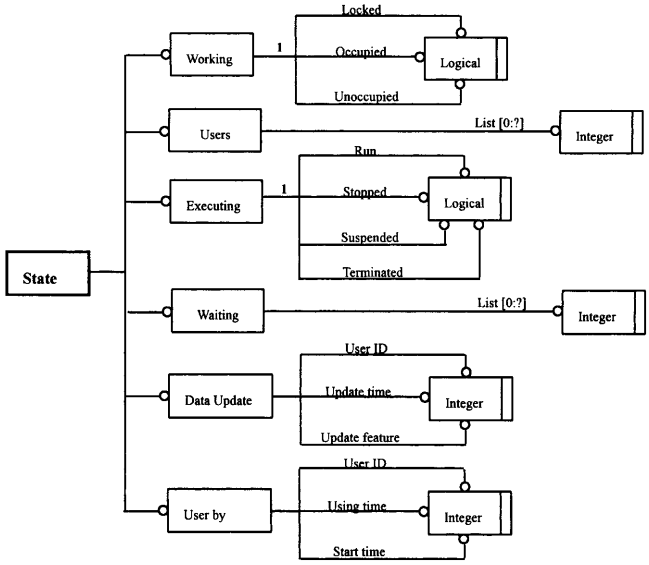

Fig. 9. State model of the meta-resource information model.

2. Working state: The *working state* can be used by the RMS for resource assignment. The *working state* of a resource is one of *locked, occupied* or *unoccupied.*
3. User state: The *user state* is a list of users who are using this resource. It is used for resource assignment.
4. Waiting list: The *waiting list* is a list of users who are waiting to use this resource. When the *working state* of a resource is changed from *occupied* to *unoccupied*, this resource can be assigned to the first user in the *waiting list.*
5. Used by: The *used by user* attribute records current user information, including user ID, start time, and using time.
6. Data update: When a resource is changed by a user, the resource server will send a message to the RMS, which includes the user ID, update time and update features.

6.2.2 Configuration

The configuration class defines static information about a resource that is useful for the user for subscribing to virtualised resources in order to configure his/her specific work-space. The configuration class consists of four attribution classes, as shown in Fig. 10.

6.2.3 Association

In the VIE, one resource may be associated with other resources. These association relations are very important features of the resource, especially for resource management. Three association relations are defined, as shown in Fig. 11.

1. Parent–child: This is a tightly coupled relation between two resources. One resource may have 0 or 1 parent and 0 to *N* children. If one resource is deleted, all its child resources should be deleted accordingly.

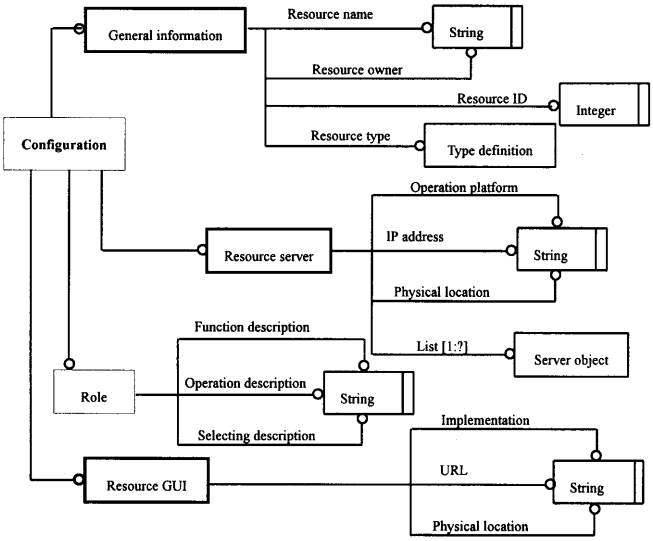

Fig. 10. Configuration class of the meta-resource information model.

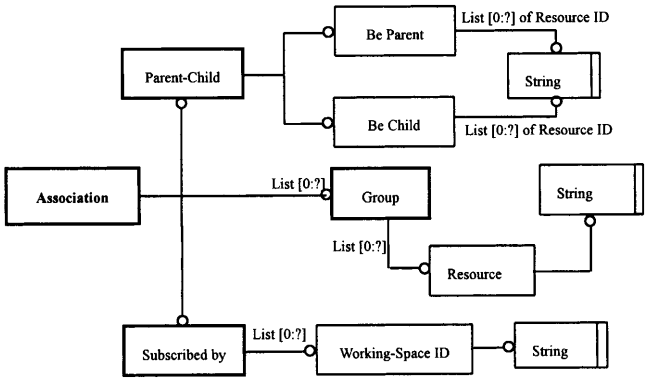

Fig. 11. Association class of the MRIM.

2. Subscribed by: This is a loosely coupled relation between resources and working-spaces. One resource may be subscribed by 0 to *N* WSs. If one resource is deleted, the configuration of the corresponding WSs will be modified.
3. Group: This is a parallel relation between several resources. One resource may belong to 0 to *N* different groups. A group contains at least two resources. If the "data-update" state of one member of a group is changed, appropriate messages should be sent to other members of the group.

6.2.4 Rule

As shown in Fig. 12, the rules of resources include access rules, operation rules, and process rules. Access rules are used by the RMS in order to control resource assignments. Operation rules are used to supervise user operation to resource by resource server. Process rules are used by RMS in order to make decisions on what services should be provided if resource states are changed. Some examples on resource rules are given in Table 1.

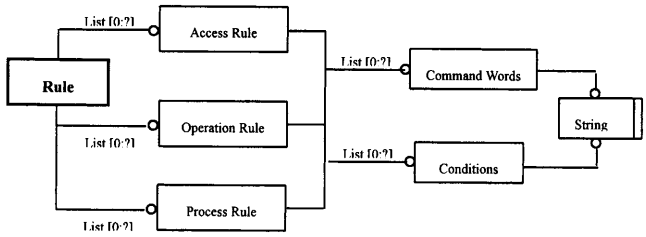

Fig. 12. Rule class of the MRDM.

7. Rule-Based Decision Management

We discussed the basic function requirements of virtualised resource management in Section 5.2. In this section, we will discuss the relevant methods for implementing these management functions. From the viewpoint of resource management, the resource in the VIE has the following characteristics:

1. Its states are constantly changed.
2. Its users are alterable.
3. Its using rules may be different to each other.

In order to adapt to this kind of constant change, RMS must be provided with a dynamic decision capability. In this section, a method of rule-based decision management will be discussed in detail. Figure 13 depicts the action model of the rule-based decision management.

7.1 Event

The whole rule-based decision process begins from receiving an event package. Events are either outer events or inner events. An outer event comes from the RMS. It is launched by one WS or one service object. An inner event is triggered by the RMS according to the rules and conditions during the decision process. An event package is triggered for requesting object service or for publishing a message. The format of an event package must be standardised for the sake of identification. An event package should at least contain the following information: IP address and ID of event launcher, the reference handle of the requested object server, and operation parameters.

7.2 Object Server Repository and Meta-Resource Data Repository

These two repositories will support a rule-based decision. The execution codes and rules related to object services are stored in the object server repository. The conditions and rules related to resource operation are recorded in the meta-resource data repository.

7.3 Rules and Conditions

Rules are classified into two classes:

1. The resource rule that contains resource access rules, resource operation rules, and process rules (see in Section

Table 1. The example of a rule-based decision.

Event				Rule		Decision result	
Launcher	Type	Object	Parameter	Commands	Conditions	Actions	Execution order
User1	Start RPP	Res *A* (RP controller)	File name	Access rule: start	If authority $> n$	S_1: start Res *A*	S_1
User2	Notify	Res *B* (CAD system)	New state value	Process rule: trigger event E1	If state is changed	S_2: modify state value E_1: notify Res *C* (another CAD system)	S_2–E_1
User3	Request object server	Res *D* (RPP system)	No	1. Access rule: access 2. Access rule: request 3. Process rule: trigger event E2	1. If authority $> n$ 2. If using list is null 3. If accept	S_3: accept request E_2: notice Res *D*	E_2–S_3
User4	Delete resource	Res *E* (NCP system)	No	1. Access rule: access 2. Process rule: trigger event E3 3. Process rule: trigger event E4	1. If authority $> n$ 2. If has subresource 3. If has been ordered by working-space	E_3: delete sub-Res *R* S_4: delete Res *E* E_4: modify working-space *W*1	E_3–S_4–E_4

Res = resource, S = service, E = event

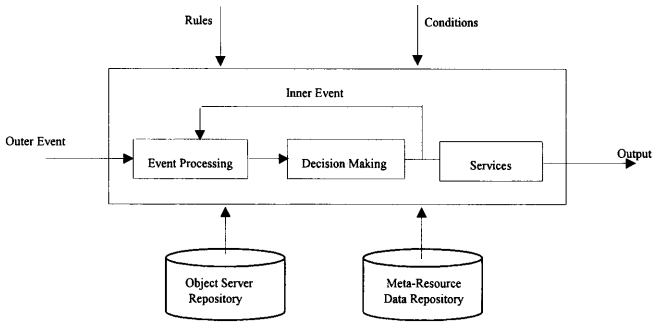

Fig. 13. Rule-based decision action model.

5.2). Resource rules are set initially by the resource owner or administrator during the resource registering process.

2. The object service rule that is fixed and attached to the object service execution codes. Conditions also consist of two parts: Part one is related to the user information. Part two involves the state and association information of the used resource.

7.4 Event Process Module

The function of the event processing module is to prepare for a rule-based decision, such as finding out the object server from the object server repository, obtaining the resource rules from the MRDR, or obtaining the condition related to the resource rules in the MRDR.

7.5 Decision Making

Decision making is a process of analysis and judgement based on rules and conditions. The result of making a decision is a series of action commands of these two types: "to provide some kind of service" or "to trigger a new event". In order to describe different kinds of decision result, the sequence of actions generated are illustrated as "decision trees" in Fig. 14. The root node of the decision tree presents an initial event. Each of its children represents one action command. The order in which the action commands on the same level are executed is from left to right. Figure 14 shows four kinds of probable decision result. Table 1 provides the four typical examples that correspond to the four situations in Fig. 14.

7.6 Recursive Decision

Comparing Fig. 14 and Table 1, it is not difficult to conclude that one decision in response to an outer event may trigger one to two new inner events. In turn, each of these new inner events will bring about one new decision. This new decision may in turn trigger new services and events. In this case, the rule-based decision making is a recursive process. It is clear that if the process of a decision is recursive, only when all the subprocesses of this recursion are accomplished will all the necessary services and their execution order be known. An algorithm has been developed in order to solve this kind of recursive problem. As shown in Fig. 13, the input of the algorithm is one outer event, and its output is a series of services as the final result of the rule-based decision.

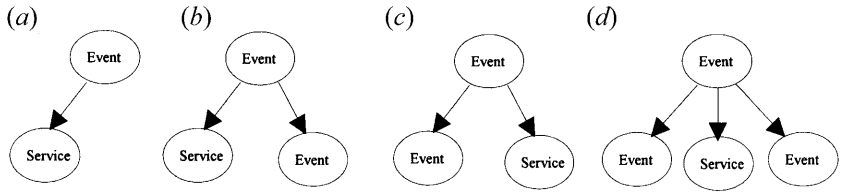

Fig. 14. Four kinds of decision tree.

Algorithm of Recursive Decision

1. Set stack P = null.
2. Make decision in response to initial outer event E_0; and produce an initial decision tree T_0.
3. Search the children of decision tree T_0 from left to right; if a new inner event is found, push it onto stack P.
4. Get a new inner event E_{new} from stack P, if stack P is null, then goto 8, else continue.
5. Make decision in response to inner event E_{new}; and produce a new decision tree T_{new}.
6. Search the children of decision tree T_{new} from left to right; if new internal event is found, push it into stack P.
7. Insert T_{new} to T_0 at the location of the node E_{new} location. Goto 4.
8. Up to this step, the whole recursive decision process has been finished. The final decision tree T_0 contains the results of each recursive decision. The root and non-terminal nodes of the decision tree T_0 represent events and every leaf node represents a service. Hence, a series of services can be obtained by traversing all leaves of decision tree T_0 in-order traversal mode.

Figure 15 gives an example to illustrate the recursive decision processes based on the algorithm above. Supposing resource A contains subresources B and C; working-space W_1 subscribes resource A, and working-space W_2 subscribes resource B. If one outer event is launched by an administrator to delete resource A, the final services order can be computed as S_2–S_3–S_4–S_5–S_1.

8. Implementation of Virtualised Enterprise

Based on the theory and methodology above, a prototype of virtualised enterprise (VIE/1.0) has been successfully developed by the Department of Computing at The Hong Kong Polytechnic University. Through this prototype, Internet-based design and manufacturing has been implemented by resource sharing and reconfiguration. The VIE/1.0 includes four parts: virtual space, resource management system, virtual working environment, and virtualised resource.

8.1 Building Virtual Space

Virtual space is built on the information communication network based on Internet and TCP/IP protocol. Web browsers such as Netscape or Microsoft Internet Explorer can be used as working platforms for users to enter VIE/1.0. In order to support the information exchanges and object operations in the VS, four types of Internet server are used. We use a Microsoft Internet Information Server (IIS) as an Internet WWW server to create a specialised WS dynamically by taking advantage of the Active Service Page (ASP) mechanism. CORBA-based Orbix Web daemons, Orbix Java daemons and Orbix C++ daemons are used as Internet monitors to support distributed object operations based on CORBA/IIOP. As shown in Fig. 16, four types of communication channel are built in the VIE/1.0.

Channel 1 is based on Internet/HTTP protocol and the HTML/ASP mechanism. It is used in the communication

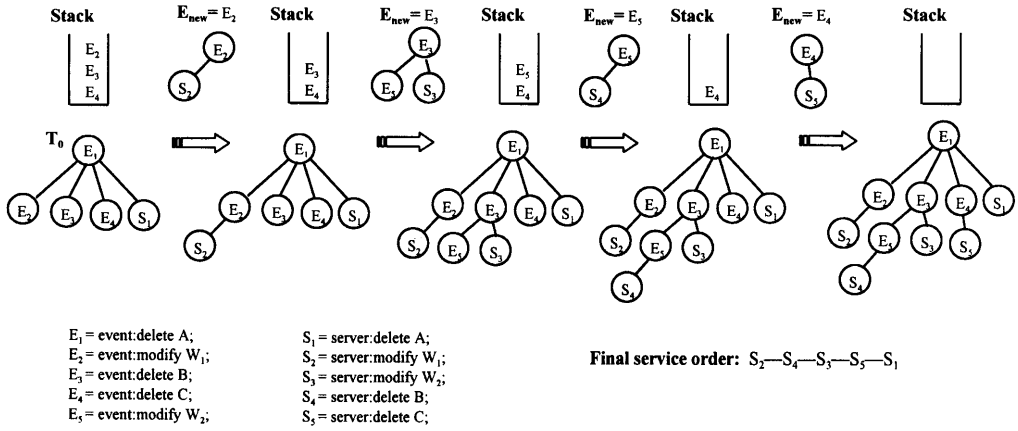

Fig. 15. An example of rule-based recursive decision.

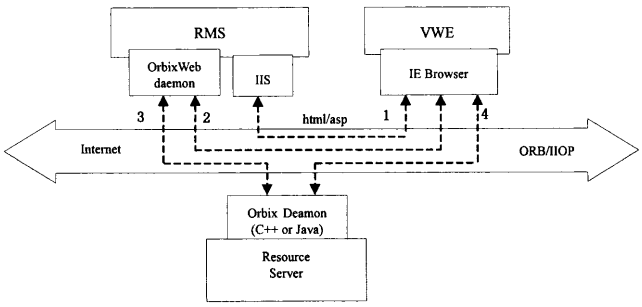

Fig. 16. The four types of communication channel in the VIE/1.0.

between the IE browser and IIS in order to implement some resource management functions, such as resource registry, resource query, WS subscription, and WS download.

Channel 2 is also built between the VWE and the RMS, and is used for communication between the IE browser and the Orbix Web daemon for the other resource management activities related to CORBA-based object operation, such as resource request and assignment.

Channel 3 acts as a connection between the RMS and the resource server for carrying out resource control and supervision.

The final resource operations between the VWE and resource servers are established through Channel 4. Channels 2, 3 and 4 are all based on CORBA/IIOP.

8.2 CORBA-Based Resource Wrapping and Object Operation

We use a practical example to explain the method of CORBA-based resource wrapping and distributed object operation. As shown in Fig. 17, the resource to be wrapped is a controller of a rapid prototyping machine (FDM1600). One of the functions of this controller is to send a machining program (smlfile) to the FDM1600 through RS232 port (A) [22]. This function is operated by a DOS command: "Ssend smlfile portname". In order to wrap this function by the CORBA-based method, at least the following five steps must be performed.

1. Define a standard interface in CORBA/IDL according to the function to be wrapped.
2. Compile this IDL interface file using the IDL compiler in order to generate four kinds of progam code (Java or C++): Stub Head files, Stub files, Skeleton Head files, and Skeleton files.
3. Write a server program that creates a CORBA server ("SendServer") and implements the server function of the resource in it.
4. Write a client application program to bind this CORBA Server ("SendServer") and operate the implementation object through IDL interface.
5. Register this CORBA server ("SendServer") in the server repository so that the Orbix daemon can launch this server automatically when it is invoked.

Before invoking practical operations on a CORBA server object, an Orbix daemon should be run. The operation process will have two phases. First, the client program invokes the server object through the ORB stub, which will send a command "bind (ServerName, HostName) to the Internet. This message will be received by the Orbix daemon, which will search for the specific server (SendServer) in the server repository and launch it automatically if it exists. Then, the Orbix daemon will reply to the client with the message: "CORBA_Orbix_imp_is_ready". Finally, actual interoperation will begin through the ORB stub and the ORB skeleton.

8.3 Resource Management System and Virtual Working Environment

The architecture of the RMS and the VWE is shown in Fig. 18. The kernel of the VIE/1.0 is the RMS, which was built on the Windows NT platform. Two Internet monitor systems are run simultaneously. One is the Microsoft IIS server that is in charge of listening for and processing information exchange between the RMS and the VWE. Another Internet server is the OrbixWeb daemon responsible for supervising CORBA-based object service requestss which come from the VWE. The RMS contains three types of repository: meta-resource data repository, object server repository, and ASP file repository. The principle and operation processes between the RMS and the VWE are as follows.

First, a person enters the homepage of the VIE/1.0 via an URL addresss (http://158.132.10.251/work/index.asp). This log-in message is received by the IIS server. The IIS server will locate the file index.asp from the ASP file repository and produce a corresponding homepage.html file, and then sends it back to the user. The browser at the user end will display this homepage.

Secondly, the person can enter VWE by submitting username and password. The IIS server automatically distinguishes between administrators and users and sends back the corresponding admini-space.html or working-space.html file to the client end.

Thirdly, if the person is an administrator, the admini-space page will be displayed on the client platform. An administrator can execute management operations, such as resource registration, resource query, and resource maintenance. Figure 19 shows a Web page for resource registration. Figure 20 depicts a Web page for resource query. Since resource registration and query operations only involve general database operations, they can be processed by the IIS server through directly accessing and operating the meta-resource repository. However, other resource operations, such as updating and management, involve rule-based decision management, they must be processed by the OrbixWeb daemon. When the OrbixWeb daemon receives an object request event, it will search in the object server repository to find the matched object server. Once found, the object server will be launched by the daemon. This object server must also access the meta resource data repository throught the ODBC driver. Based on the rules and states of the resource, the object server will make a decision and then perform the related resource maintenance operations.

If the client is a general user, a special working-space page will be downloaded and displayed on the user's platform. This

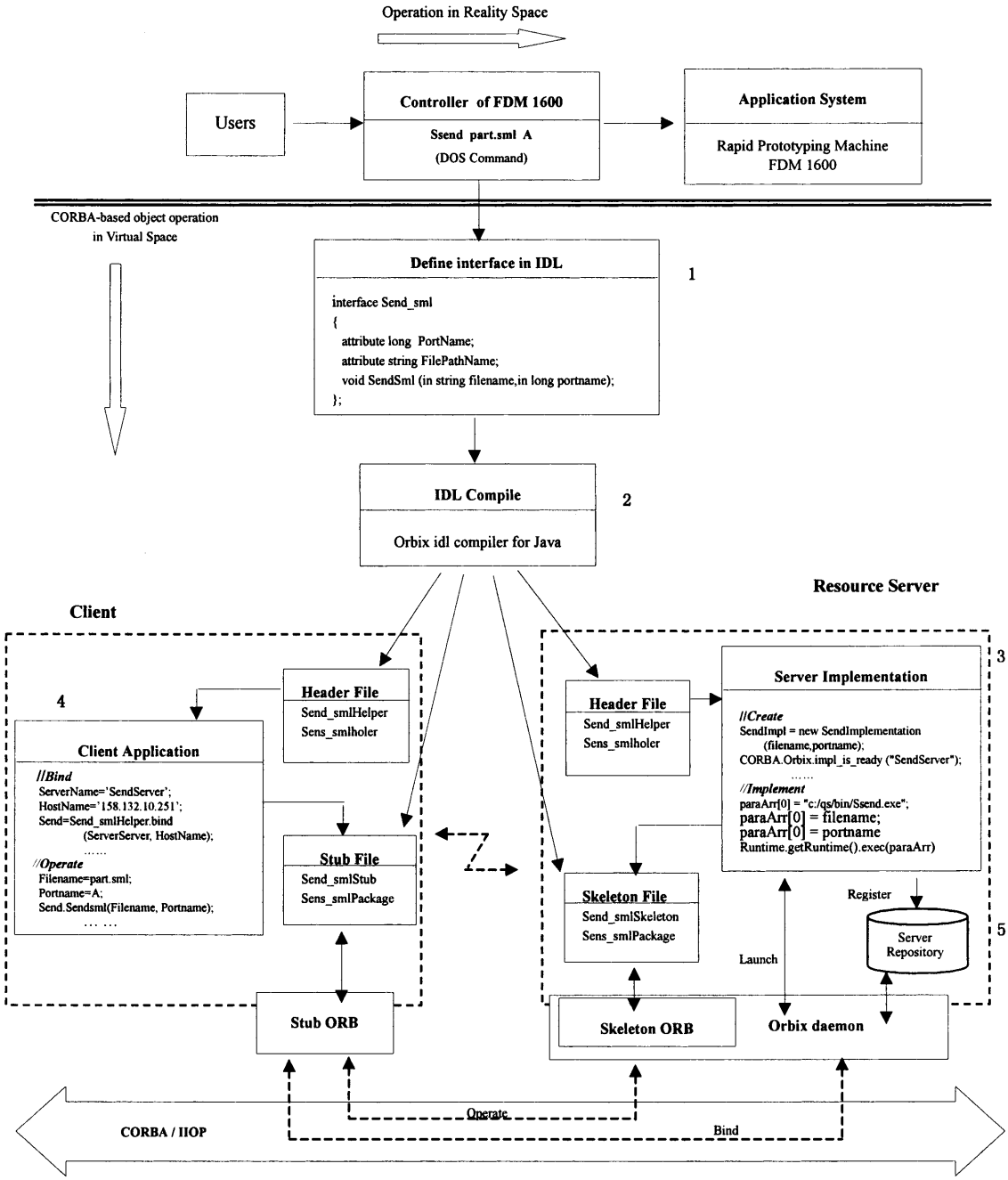

Fig. 17. The CORBA-based resource wrapping and object operation.

working-space page contains some common software tools and user-ordered RGUIs. Requests from common software tools will be processed by the IIS server. Requests from RGUIs, however, will be processed by the OrbixWeb daemon. If the request is accepted, real interactive operation between the user and the virtualised resource will begin.

One customised working-space has subscribed to five virtualised resources: Lonicera-NCP, Lonicera-MDA, FDM1600-RPM; QuickSlice-RPP, and CSM Session Manager.

Lonicera-NCP is a numerical control (NC) programming system that generates NC milling machine tool paths.

Lonicera-MDA is a mechanical design assistant that includes a number of basic computer-aided design (CAD) modules, such as 2D engineering drawing, 3D parametric model, advanced surface model, and geometric model data conversion, and so on.

Quickslice is a powerful slicing program that creates Stratasys modelling language (SML) codes to drive the fused deposition modelling (FMD) system.

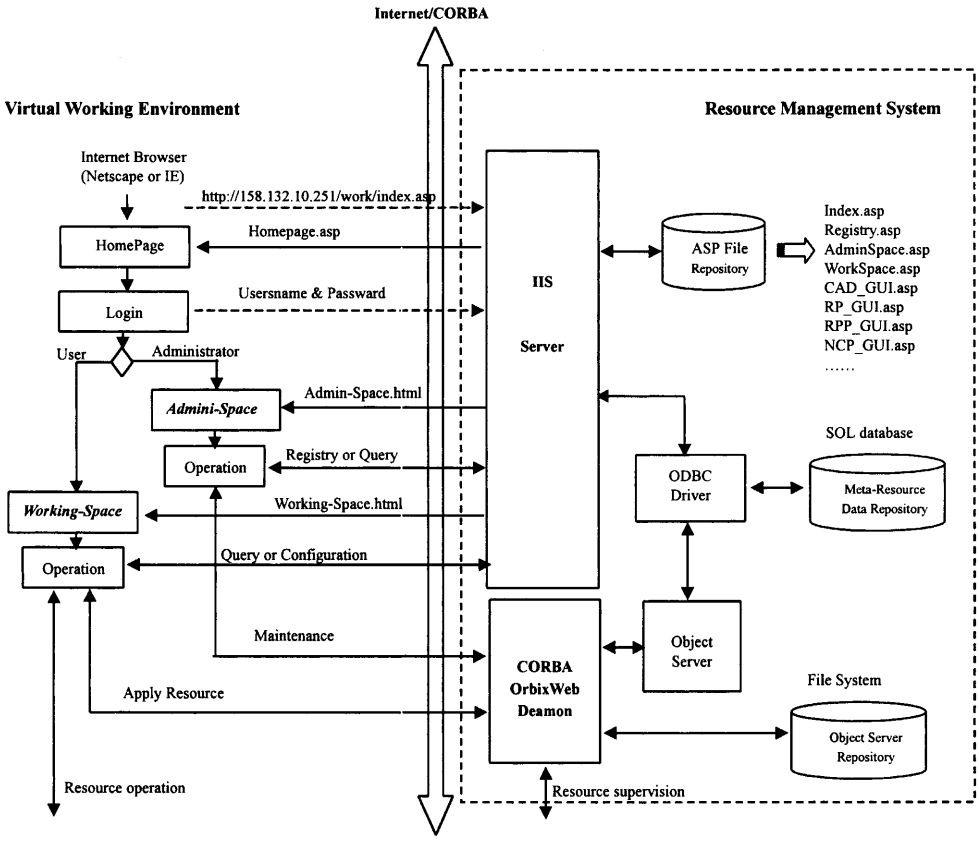

Fig. 18. The architecture of RMS and VWE in VE/1.0.

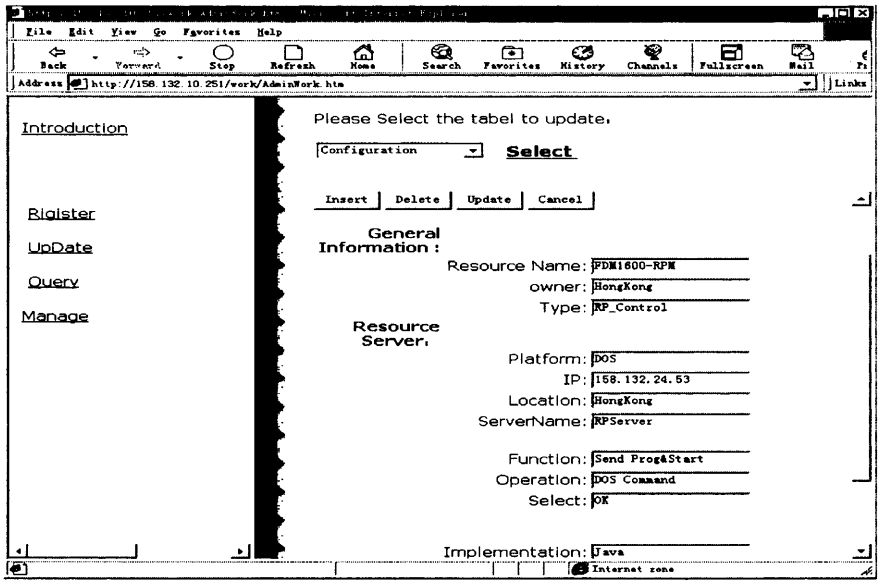

Fig. 19. Resource registry in VIE/1.0.

FDM1600 is a rapid prototyping machine made by Stratasys Inc.

CSM is an experimental collaborative solid modelling system that allows multiple designers to collaborate in real-time on the Web [23].

In Fig. 21, QuickSlice-RPP's RGUI is opened, through which the user can undertake remote programming for a rapid prototyping machine, such as uploading an STL file from another resource server to the QuickSlice-RPP server, setting control parameters, checking operation processes, viewing and

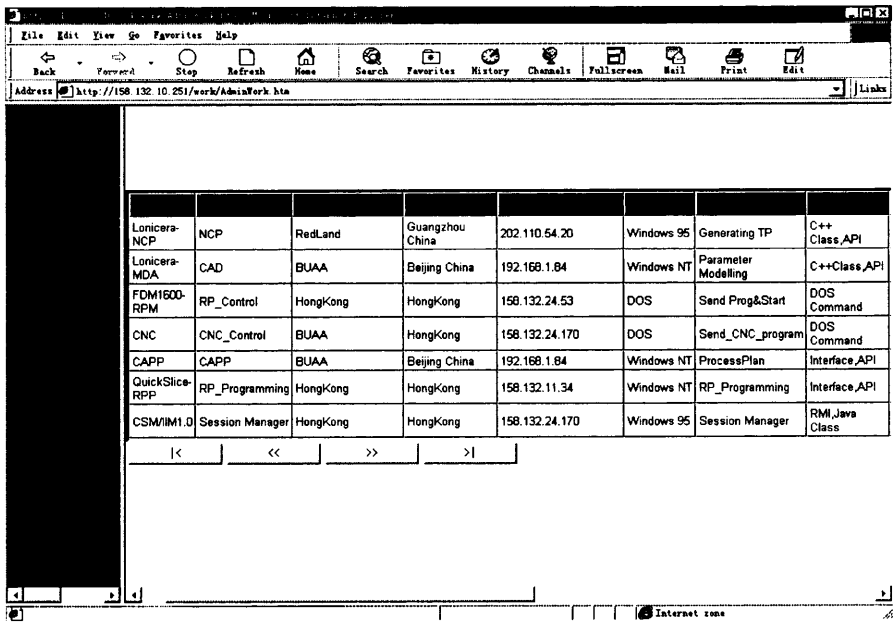

Fig. 20. Resource query in VIE/1.0.

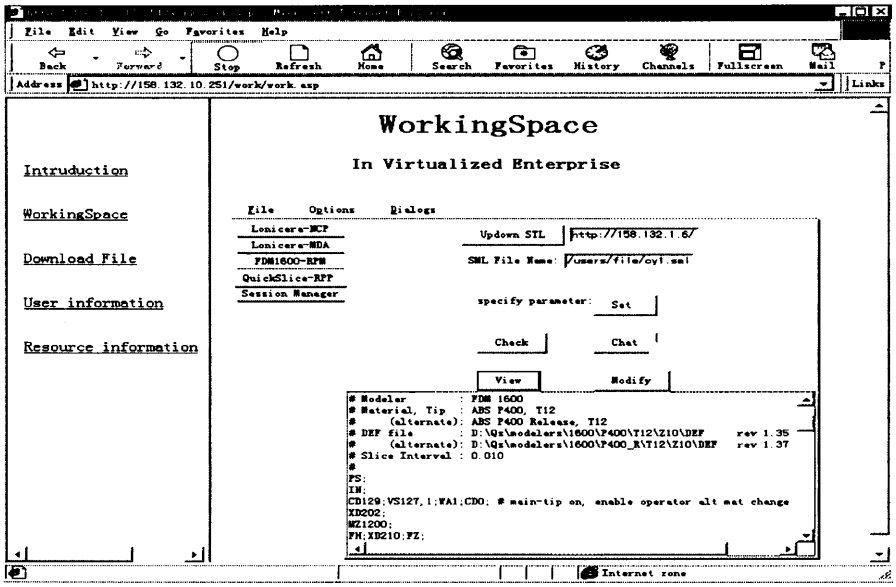

Fig. 21. A customised working-space: RGUI – QuickSlice-RPP.

modifying the outputs, and so on. Another RGUI, FDM1600-PRM, is illustrated in Fig. 22. Some remote machining controls can be implemented through this RGUI. For example, the user can upload a machining program from any other resource server to the FDM1600-PRM server, send it to the RP machine and start the machining process.

8.4 Resource Wrapping and Resource Registry

Five resources have been wrapped and registered in the RMS. The characteristics and relevant wrapping information are summarised in Table 2. The whole processes of resource wrapping is roughly divided into four steps:

1. Analyse the basic characteristics of resources, such as resource location, platform, language, role and main functions.
2. Analyse the operation interfaces provided by the resource; select the wrapping method; define the unified function interface.
3. Finally, implement resource wrapping: define and implement the RGUI of the wrapped resource and implement the management functions of the wrapped resource which are

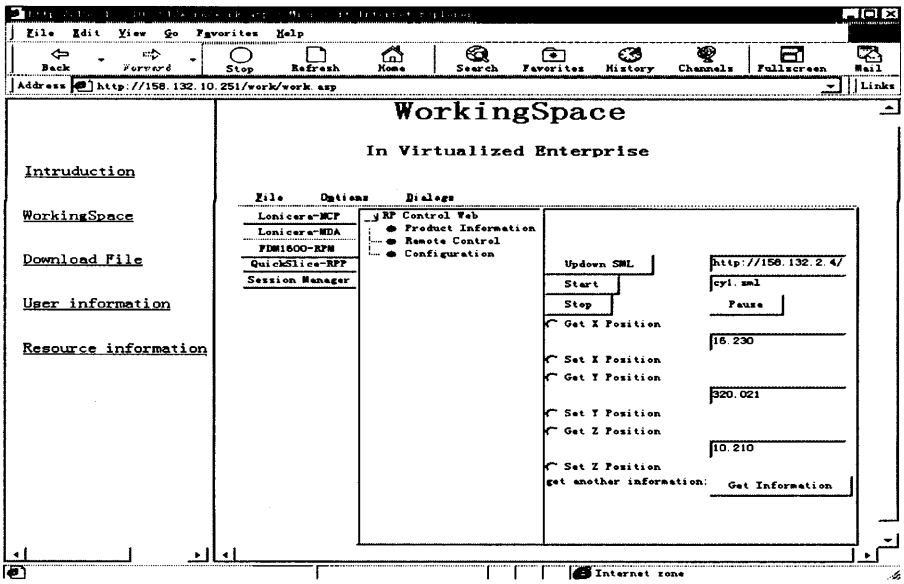

Fig. 22. A customised working-space: RGUI – FDM1600.

Table 2. Five different type of resource wrapping.

Processes	Name	Lonicera-MDA	Lonicera-NCP	QuickSlice	FDM1600 Controller	CSM/IIM1.0
1	Location	Beijing, China	Guanzhou, China	Beijing, China	Hong Kong	Hong Kong
	Platform	Windows/95/NT	Windows/95/NT	Windows NT	DOS	Windows/95/NT
	Language	C++	C++	Unknown	Unknown	Java
	Role	Product design	NC programming	RP programming	RP machine	Session system collaborative design
	Main functions	2D draft 3D parameter model Assemble PDM Product data conversion	2.5D TP generation Surface TP generation TP simulate Post-process	Produce SSL Produce SML	Send SML to RP machine Machining process Supervise RP machine configuration	Session manager Solid modelling STEP to UG RP request service
2	Operation mode	Interaction C++ class library API	Interaction C++ class library API	Interaction DOS	Interaction DOS	Interaction Java class RMI
	Wrapping mode	CORBA/C++ Through API	CORBA/C++ Through API	CORBA/J++ Through file and interaction	CORBA/J++ Through DOS common	CORBA/J++ Through RMI
	Interoperable object (interface)	3D-Model STEP-SAT SAT-STL Assemble	2.5D-TP Surface Post-process	file-modify file-view	Ssend Runacl	Session manager STEP-UG RP-service
3	Virtualised resource GUI	3D-Model Data conversion Assemble	2.5D-TP surface TP Post-process	Command Window Supervise Window	RP control Supervise Program Manager	Session manager Data conversion
4	Manageable object (interface)	State-Notice Using-Notice Access-control Operate-control Rule-modify Rule-query	Same as left	Same as left	Same as left	Same as left

common to all the virtualised resources.

4. Register the wrapped resource into the RMS.

9. Conclusion

We believe that virtualised enterprise is an important and necessary developmental stage to progress from integrated enterprise to virtual enterprise during the whole process of enterprise development. This paper analyses and explains some concepts related to the virtualised enterprise in detail, including the architecture, key elements, basic activities, and the global model.

The key technology in the implementation of virtualised enterprise is "virtualisation". Virtualisation of an enterprise can be divided into three subprocesses: resource virtualisation, environment virtualisation, and management virtualisation. The main research achievements on virtualisation technology include object-agent-oriented wrapping, platform-independent RGUI, meta-resource data model, meta-resource data repository, rule-based resource management and the reconfiguration of working-space. A prototype of an virtualised enterprise has been developed. The relevant theory and methodology presented in this paper have been partly tested and verified.

This paper has concentrated on virtualisation of an enterprise. Because the virtualised enterprise is built in virtual space, more work on how to build the virtual space to support the virtualised enterprise is required. This will involve the methods and technologies of the communication infrastructure, standard protocols, security, real-time operations, and so on. How to combine multiple virtualised enterprises to constitute a virtual enterprise is another topic for future work.

Acknowledgement

The work described in this paper was partially supported by Hong Kong UGC/RGC/CERG Research Project PolyU5102/97E.

References

1. S Chan, V. Ng, K. M. Yu and A. Au, "An internet-integrated manufacturing system prototype", submitted to 1999 International Computer Science Congress, Hong Kong, 13–15 December 1999.
2. S Rajagopalan, J. M. Pinilla and P. Losleben, "Integrated design and rapid manufacturing over the Internet", in Proceedings DETC98 1998 ASME Design Engineering Technical Conference, Atlanta, GA, 13–16 September 1998.
3. S. Zhang and B. Chen, "A concept of virtual global manufacturing based on autonomous manufacturing islands", in CIRP International Symposium – Advanced Design and Manufacturing in Global Manufacturing Era, Hong Kong, 21–22 August 1997.
4. C. K. Yung, C. Grier and I. Lin, "Development of collaborative CAD/CAM system", Robotics and Computer-Integrated Manufacturing, 14, pp. 55–68, 1998.
5. ISO/IS 10303-21, Product Data Representation and Exchange – Part 21: Clear Text Encoding of the Exchange Structure, 1994.
6. ISO/IS 10303-1, Industrial Automation Systems – Product Data Representation and Exchange: Part I, Overview and Fundamental Principles, 1994.
7. Part 213, Application Protocol: Numerical Control (NC) Process Plans for Machined Parts. ANSI USPRO/IPO, 1200-213-DIS, 1997.
8. K. Kosanke, F. Vernadat and M. Zelm, "CIMOSA: enterprise engineering and integration", Computers in Industry, 40(2–3), pp. 83–97, 1999.
9. J. W. Erkes, K. B. Kenny, J. W. Lewis, B. D. Sarachan, M. W. Sobolewski and R. N. Sum, "Implementing shared manufacturing services on the World Wide Web", Communications of the ACM, 39(2), pp. 34–45, 1996.
10. A. Dewey, "The impact of NIIIP virtual enterprise technology on next generation manufacturing", in Proceeding of Conference on Agile and Intelligent Manufacturing Systems, Troy, NY, 1996.
11. Y. M. Chen, C. C. Liao and B. Prasad, "A system approach of virtual enterprising through knowledge management techniques", Concurrent Engineering: Research and Applications, 6(3), pp. 225–244, 1998.
12. C. Gilman, M. Aparicio, J. Barry, T. Durniak, H. Lam and R. Ramnath, "Integration of design and manufacturing in a virtual enterprise using enter rules, intelligent agents, STEP, and workflow", Architecture, Network and Intelligent System for Manufacturing, Integrated Proceeding, Society of Photo-Optical Instrumentation Engineers (SPIE), http://smart.npo.org/public-forum/, 1997.
13. J. Barry, M. Aparicio, T. Durniak, C. Gilman and R. Ramnath, "NIIIP-SMART: an investigation of distributed object approaches to support MES development in a virtual enterprise", The Second International Enterprise Distributed Computing Workshop (EDOC98), http://smart.npo.org/public-forum/, 1998.
14. NIIIP, Introduction to NIIIP Concepts. NIIIP Reference Architecture, Book 0, 1998.
15. NIIP, Guide to the NIIIP Reference Architecture Model. NIIIP Reference Architecture, Book 1, 1998.
16. NIIIP, NIIIP Planner's Guide, NIIIP Reference Architecture, Book 2, 1998.
17. T. J. Mowbray and R. Zahvi, The Essential CORBA: System Integration Using Distributed Objects, John Wiley, New York, 1995.
18. C. Hsu, M. Bouziane, L. Rattner and L. Yee, "Information resource management in heterogeneous, distributed environments: a metadatabase approach", IEEE Transaction on Software Engineering, 17(6), pp. 604–624, 1991.
19. C. Hsu, A. Rubenstein, L. Yee, G. Babin, N. Lawson and W. Hofmann, "What is Rensselaer's metadatabase system?", Proc. 3rd International Conference on computer integrated manufacturing, IEEE Computer Society, http://vill.eng.rpi.edu/publication.html, 1992.
20. C. Hsu, G. Babin, M. Bouziane, W. Cheung, L. Rattner, A. Rubenstein and L. Yee, "The metadatabase approach to integrating and managing manufacturing information systems", Journal of Intelligent Manufacturing, 5(5), pp. 333–349, 1994.
21. G. Booch, J. Rumbaugh and I. Jacobson, The Unified Modeling Language User Guide, Addison-Wesley, 1999.
22. Stratasys, QuickSlice Manual Release 6.0, 1998.
23. S. Chan, M. Wong and V. Ng, "Collaborative solid modeling on the WWW", 1999 ACM Symposium on Applied Computing – Special Track on World Wide Web Applications, 1999.