(57) Abstract

Methods to display and manage upstream and downstream dependencies of a resource in a group of resources (200), and to define and establish services based on a predetermined group of resources are described. A method to automatically indicate upstream and downstream resource dependencies includes selecting a first resource, determining a second resource (the second resource dependent on the first resource), determining a third resource (the first resource dependent on the third resource), displaying a graphical indication of the first resource, the second resource, and the third resource, and graphically indicating the dependence between the first and second resources and the dependence between the first and third resources. A method to automatically establish (e.g., create, copy, or move or migrate) a service includes obtaining a template identifying resources (200), one or more dependencies between the identified resources, and properties associated with the identified resources, and establishing dependencies between the resources of the resource group as indicated in the template.
FOR THE PURPOSES OF INFORMATION ONLY

Codes used to identify States party to the PCT on the front pages of pamphlets publishing international applications under the PCT.

| AL | Albania | ES | Spain | LS | Lesotho |
| AM | Armenia | FI | Finland | LT | Lithuania |
| AT | Austria | FR | France | LU | Luxembourg |
| AU | Australia | GA | Gabon | LV | Latvia |
| AZ | Azerbaijan | GB | United Kingdom | MC | Monaco |
| BA | Bosnia and Herzegovina | GE | Georgia | MD | Republic of Moldova |
| BB | Barbados | GH | Ghana | MG | Madagascar |
| BE | Belgium | GN | Guinea | MK | The former Yugoslav Republic of Macedonia |
| BF | Burkina Faso | GR | Greece | ML | Mali |
| BG | Bulgaria | HU | Hungary | MN | Mongolia |
| BJ | Benin | IE | Ireland | MR | Mauritania |
| BR | Brazil | IL | Israel | MW | Malawi |
| BY | Belarus | IS | Iceland | MX | Mexico |
| CA | Canada | IT | Italy | NE | Niger |
| CF | Central African Republic | JP | Japan | NL | Netherlands |
| CG | Congo | KE | Kenya | NO | Norway |
| CH | Switzerland | KG | Kyrgyzstan | NZ | New Zealand |
| CI | Côte d'Ivoire | KP | Democratic People's Republic of Korea | PL | Poland |
| CM | Cameroon | KR | Republic of Korea | PT | Portugal |
| CN | China | KZ | Kazakhstan | RO | Romania |
| CU | Cuba | LC | Saint Lucia | RU | Russian Federation |
| CZ | Czech Republic | LI | Liechtenstein | SD | Sudan |
| DE | Germany | LK | Sri Lanka | SE | Sweden |
| DK | Denmark | LR | Liberia | SG | Singapore |
| EE | Estonia | SI | Slovenia | SK | Slovakia |
| SN | Senegal | SZ | Swaziland | TD | Chad |
| TG | Togo | TJ | Tajikistan | TM | Turkmenistan |
| TR | Turkey | TT | Trinidad and Tobago | UA | Ukraine |
|UG | Uganda | US | United States of America | UZ | Uzbekistan |
| VN | Viet Nam | YU | Yugoslavia | ZW | Zimbabwe |
Display and Management of Computer Resource Properties

Background

The invention relates generally to computer networks and, more specifically, to displaying and managing resources in a computer network environment.

Many organizations have a substantial number of computers in operation, often located far apart. For example, a company with many work sites may have a computer at each location to track inventories, monitor productivity, and perform administrative tasks such as payroll. Typically, some or all of an organization's computers may be interconnected to form a network, where two computer systems are interconnected if they are capable of exchanging information. The connection need not be by copper wire; infrared, fiber optics, microwaves, and communication satellites may also be used.

One common computer network model is the client-server model. In a client-server network, a client computer system (a client) sends a request message to a server computer system (a server) asking the server to perform some service (e.g., do some task or provide some information). The server may then perform the service and send a reply back to the client. Thus, clients are generally consumers of services and servers are generally providers of services.

A popular client-server network is the clustered network organization promoted by Microsoft® for computers running under the Windows NT® operating system. Referring to FIG. 1, current embodiments of cluster 100 include two computer systems (referred to as nodes 102 and 104) interconnected via an optional private communication link (COM-LINK) 106 and one or more shared storage devices 108. Nodes 102 and 104 may also include local disks 110 and 112 respectively. Cluster 100 may be coupled to other computer systems (e.g., 114, 116, and 118) via local or wide area network 120. Any one, or more, of computer systems 114, 116, and 118 may be a stand-alone computer system or a cluster and may operate as clients or servers. In addition, nodes 102 and 104 and computer systems 114, 116, and 118 may be personal computers, workstations, or special purpose processors or multiprocessors.

The rapid and widespread adoption of computer network technology may generally be attributed to the goal of sharing resources. In the present context, a resource may be any entity (hardware or software) residing on a server that may provide a service to a client. Illustrative resources include internet protocol (IP) addresses, network names, physical storage devices, application programs, and data files.
Under some network management schemes, resources may be aggregated into groups and managed as discrete entities. Typically, a group includes all those resources (including their dependencies) needed to provide a service. Illustrative services include dynamic host configuration protocol (DHCP) services, file share services, internet information services (IIS), print spooler services, and time services. Resources are said to be dependent if there exists a relationship of reliance between them that makes it necessary for them to be in a common group to provide a service (e.g., a file sharing service). Thus, groups provide a convenient and intuitive mechanism to manage services. In a cluster for example, dependencies between a group’s resources may determine in what sequence the service’s resources are failed-over or failed-back.

In a cluster managed by Microsoft Cluster Server (MSCS) software, those resources assigned to a group may be determined by selecting the group in the MSCS Administrator’s graphical user interface (GUI). Once selected, the group’s resources are displayed in a list. If an IIS group (service) is selected, for example, the displayed list may include a storage device resource, a FTP service resource, a Gopher service resource, an IP address resource, a network name resource, and a WWW service resource.

Consider an IIS group having the resources listed above and the dependencies listed in Table 1. To determine what resources the FTP service depends on (hereinafter referred to as upstream resources), a user must select (via the MSCS’ Cluster Administration application’s GUI) the FTP service resource, specify presentation of a resource properties dialog box, and then inspect the FTP service’s dependency property. If the user wants to view the upstream resources for another of the group’s resources (e.g., the network name resource), the user must repeat this process. Cluster administration in accordance with MSCS provides a similar mechanism to view those resources which depend upon a specified resource (hereinafter referred to as downstream resources).
Table 1. Example IIS Group Resource Dependencies

<table>
<thead>
<tr>
<th>Physical Storage Resource</th>
<th>Depends on No Resource</th>
</tr>
</thead>
<tbody>
<tr>
<td>IP Address</td>
<td>Depends on no resource</td>
</tr>
<tr>
<td>Storage Device</td>
<td>Depends on no resource</td>
</tr>
<tr>
<td>Network Name</td>
<td>Depends on the physical storage and IP</td>
</tr>
<tr>
<td></td>
<td>address resources</td>
</tr>
<tr>
<td>FTP Service</td>
<td>Depends on network name resource</td>
</tr>
<tr>
<td>Gopher Service</td>
<td>Depends on network name resource</td>
</tr>
<tr>
<td>WWW Service</td>
<td>Depends on network name and storage device resources</td>
</tr>
</tbody>
</table>

The inability of a user to conveniently determine a group’s dependency structure (that is, the upstream and downstream dependencies for each of the group’s resources) may make it difficult to quickly and accurately manage a group’s resources (e.g., modify, add, or delete a dependency relationship). It may also make it difficult for a user (e.g., network manager) to assess what services will be lost in the event a resource is taken off-line. Further, a complete view (e.g., node, cluster, or network wide) of resource dependencies may also allow a user to make better decisions vis à vis preventative maintenance – e.g., knowing what services may be effected by a resource being taken off-line for maintenance may enable a user to plan those actions during times of non-peak usage. In addition, current resource group management techniques do not provide a mechanism to detect and configure those resources needed to provide a service.

Thus, it would be beneficial to provide a mechanism to manage services based on predetermined groups of resources, and to display and manage the upstream and downstream dependencies of any resource in a group of resources over a computer network.

**Summary**

In one embodiment, the invention provides a method to automatically indicate upstream and downstream resource dependencies. The method includes selecting a first resource, determining a second resource (the second resource dependent on the first resource), determining a third resource (the first resource dependent on the third resource), displaying a graphical indication of the first resource, the second resource, and the third resource, and
graphically indicating the dependence between the first and second resources and the dependence between the first and third resources.

In another embodiment, the invention provides a method to automatically create and/or migrate a service. The method includes obtaining a template identifying a plurality of resources, one or more dependencies between the identified resources, and properties associated with the identified resources, identifying a node on which the identified resources are available, establishing a resource group on the identified node (the resource group including the identified resources), and establishing dependencies between the resources of the resource group as indicated in the template. Methods in accordance with this embodiment of the invention may also set a property of a resource in the resource group as indicated in the template and/or indicate that the service is available for use.

**Brief Description of the Drawings**

Figure 1 illustrates a prior art clustered computer network.

Figure 2 shows a computer system's software architecture in accordance with one embodiment of the invention.

Figures 3A and 3B show a method to display upstream and downstream dependencies for a resource group in accordance with one embodiment of the invention.

Figures 4A and 4B illustrate the presentation of resource group dependencies in accordance with the method of FIGS. 3A and 3B.

Figure 5 illustrates an alternative presentation of the resource dependencies illustrated in FIGS. 4A and 4B.

Figure 6 illustrates another presentation in accordance with the invention.

Figure 7 shows a method to modify a dependency in accordance with one embodiment of the invention.

Figures 8A and 8B illustrate the presentation of resource group dependencies in accordance with the method of FIG. 7.

Figure 9 shows a method to automatically establish a service in accordance with one embodiment of the invention.

**Detailed Description**

Techniques to manage services based on a predetermined group of resources and their dependencies, and to display and manage the upstream and downstream dependencies of any
resource in a group of resources over a computer network are described. The following embodiments, described in terms of a clustered network operating within a Windows NT® environment, are illustrative only and are not to be considered limiting in any respect.

Referring to FIG. 2, a resource management application 200 in accordance with the invention is shown as one element in a computer system's (e.g., a node's) software architecture. Resource management application 200 may communicate with the node's operating system 204 and hardware 208 via cluster application programming interface (API) module 202. In one embodiment, API module 202 is the cluster API provided by Microsoft® corporation as part of the Microsoft Cluster Server (MSCS) product (e.g., the clusapi.dll file provided as one component of the MSCS version 1.0 product).

As would be known to those of ordinary skill, API module 202 provides a predefined calling convention that applications (such as resource management application 200) may use to communicate with operating system 204. For example, resource management application 200 may invoke a cluster API 202 routine to request a list of resources associated with a specified group. Cluster API 202 may also be used by resource management application 200 to request that operating system 204 modify one or more properties of a specified resource.

Information such as the status of a node or cluster (including the node and/or cluster's resources), may be maintained by operating system 204 in a database, or database-like structure. In a Windows NT® environment, for example, this type of information is stored in NT registry 206. Thus, when resource management application 200 requests information about the upstream dependencies of a specified resource, it may issue a cluster API call to operating system 204. Operating system 204, in turn, may interrogate NT registry 206 and return the requested information to management application 200 via a conventional call-back process.

A method to display upstream and downstream dependencies for a resource group in accordance with one embodiment of the invention is shown in FIGS. 3A and 3B. A group may be selected through, for example, a graphical user interface (GUI) that displays those resource groups associated with a single node (block 300). A list of resources comprising the selected group may then be generated by, for example, executing a cluster API call that returns the desired information (block 302). From the resources determined in block 302, a list of those resources that have no upstream dependencies (i.e., that do not depend on another resource) may be generated (block 304) and indications for each of these resources (e.g., icons) displayed in a first column on a display screen (block 306). In one embodiment, a cluster API call may be made for each resource in a group to determine if upstream resources exist. Next, a
list of those resources that depend only directly on those resources indicated in the immediately prior columns (e.g., column one) is generated (block 308) and indications for each of these resources are displayed in a next (e.g., second) column on the display screen (block 310). If all of the resources determined in block 302 have not been indicated/displayed (the 'yes' prong of diamond 312), processing returns to block 308. Blocks 308 and 310 are repeated until all of the selected group's resources have been displayed – each iteration through blocks 308 and 310 placing resource icons in a subsequent column of the display. If all of the resources determined in block 302 have been indicated (the 'no' prong of diamond 312), processing continues at block 314 of FIG. 3B. If the selected group has resources configured so as to result in a second column (the 'yes' prong of diamond 314), column two is made the current column (block 316). For each resource indicated in the current column (e.g., column two), a directed line to that resource from a resource on which it depends may be indicated on the display (block 318). If there are columns whose resources have not been processed in accordance with block 318 (the 'yes' prong of diamond 320), the next column is made the current column (block 322) and processing returns to block 318. If the selected group does not include resources configured to result in a second column (the 'no' prong of diamond 314), or all resources indicated in all columns have been processed via block 318 (the 'no' prong of diamond 320), processing terminates (block 324). It will be understood that a group dependency display as described above and shown in FIGS. 3A and 3B may comprise more than three columns (layers) of dependencies. It will be further appreciated that a resource displayed in an arbitrary column N may have dependencies to one or more resources displayed in one or more prior columns (e.g., N-1, ..., 1).

By way of illustrating the method of FIGS. 3A and 3B, consider its application to an IIS group having the resources and dependencies listed in Table 1. In this example, block 302 will generate a resource list indicating the following resources: IP address, storage device, network name, FTP service, Gopher service, and WWW service. Block 304 will generate a list indicating the IP address and storage device resources. Referring to FIG. 4A, in one embodiment block 306 will place icons 400 and 402 representing the storage device and IP address resources respectively in a first column 404 of display 406. Block 308 will generate a list indicating the network name resource (in accordance with Table 1, the network name resource depends only directly on those resources indicated in a prior column -- column one 404). Block 310 will place icon 408 representing the network name resource in a second column 410 of display 406. At this time there remain IIS group resources that have not yet
been indicated (the 'yes' prong of block 312) and so processing continues at block 308. Block 308 will now generate a list indicating the FTP, Gopher, and WWW services (in accordance with Table 1, these resources depend only directly on resources associated with a prior column – column one in this example). Block 310 may place icons 412, 414, and 416 representing each of these resources in a third column 418 of display 406.

With all IIS group resources indicated as described above and illustrated in FIG. 4A, upstream and downstream dependencies may be graphically illustrated in accordance with blocks 314 through 322 of FIG. 3B. Referring now to FIG. 4B, block 316 makes column two 412 the current column. Block 318 indicates that the network name resource (indicated by icon 408) is directly dependent on the storage device and IP address resources (indicated by icons 400 and 402) by drawing directed lines 420 and 422 between them. Next, column three 418 is made the current column (block 322) and processing continues at block 318. Block 318 may now indicate that the FTP, Gopher, and WWW services (indicated by icons 412, 414, and 416 respectively) are directly dependent on the network name resource (indicated by icon 408) by drawing directed lines 424, 426, and 428. In addition, block 318 will indicate that the WWW service (indicated by icon 414) is dependent upon the storage device resource (indicated by icon 400) by drawing directed line 430. Thus, FIG. 4B indicates directly that the network name resource depends (i.e., has upstream dependencies) on both the storage device resource and the IP address resource, while also being a resource upon which the FTP, Gopher, and WWW services depend; the FTP, Gopher, and WWW services are downstream resources to the network name resource.

In another embodiment, the icons representing the group’s storage device and IP address resources (400 and 402 respectively) in FIG. 4 may be reordered to reduce, or in some cases avoid, the need to cross resource dependency lines. As shown in FIG. 5, the column-one to column-two dependencies 420 and 422 may be redrawn to 420A and 422A, while the column-one to column-three dependency 430 may be redrawn to 430A. A benefit of rearranging the icons in this manner is to reduce the amount of overlap in dependency lines so as to improve their clearness to a user.

In yet another embodiment, a method in accordance with FIG. 3 may not use directed lines. Rather, upstream and downstream dependency relationships may be displayed implicitly. For example, resources to the left of a given resource may implicitly be upstream resources, while those to the right may implicitly be downstream resources. Another way in which upstream and downstream relationships may be displayed is through a dependency tree as
shown in FIG. 6. Again, upstream and downstream dependencies are immediately and intuitively observable from the displayed information.

In still another embodiment, a method in accordance with FIGS. 3A and 3B may be applied to all groups within a node, all nodes within a cluster, all clusters within a domain, or across multiple domains. If upstream and downstream dependencies for all groups within a node are to be displayed, a user may select a node through a GUI. A method in accordance with the invention may then issue API calls to determine all groups associated with the node and, for each of these groups, a display in accordance with FIGS. 3A, 3B, and 6 may be generated. If upstream and downstream dependencies for all groups on all nodes within a cluster are to be displayed, a user may select a cluster through a GUI, and a method in accordance with the invention may then issue API calls to determine all of the cluster’s nodes and, subsequently, issue additional API calls to determine which groups are associated with each node and, for each of these groups, generate a display in accordance with FIGS. 3A, 3B, or 6. Similar techniques may be used to determine which clusters exist within a domain, and what domains are available in a network.

The ability to simultaneously display upstream and downstream dependencies within a single group, across nodes, clusters, and domains may provide important network management capability that is not currently available. For example, the simultaneous display of upstream and downstream dependencies may make it easier for a user (e.g., network manager) to assess what services will be lost in the event a resource is taken off-line, e.g., during maintenance. A complete view of a service’s dependencies may also assist the user in determining what resources must be brought back on-line, and in which order, to reestablish a service.

A graphical display of intra-group dependencies such as that shown in FIG. 5 may also be used to modify the dependencies between group resources. Referring to FIG. 7, a user may select a first resource such as by single clicking on the resource’s icon with a mouse pointer device (block 700), and dragging the selected resource onto an icon associated with a second resource (block 702). The first selected resource may then be made dependent on the second resource (block 704). A graphical representation of the newly created dependency may then be generated by a method in accordance with FIGS. FIGS. 3A and 3B (block 706).

By way of illustrating the method of FIG. 7, consider making the FTP service dependent upon the storage device resource as shown in FIGS. 8A and 8B. First, icon 400 is selected as represented by box 800 (block 700). Next, icon 400 is moved (represented by arrow 802) onto the FTP service icon 412 as represented by box 804 (block 702). Resource
management application 200 may then issue one or more API calls to operating system 204 to modify the dependency properties of the FTP service (block 704). For example, in a computer system such as that illustrated in FIG. 2, operating system 204 may modify NT registry 206. Once the FTP resource’s dependency has been modified in accordance with the user’s select-and-drag operation, a graphical representation of the group’s new dependencies may be generated by a method in accordance with FIGS. 3A and 3B, the result of which may appear as shown in FIG. 8B – the new dependency between the storage device resource and FTP service is shown by directed line 806.

In one embodiment cyclic dependencies are not allowed to form. For instance, if resource A is dependent upon resource B and resource B is dependent upon resource C, a method in accordance with the invention will not form a dependency such that resource A depends on resource C. Thus, if a user selects and drags an icon representing resource A onto an icon representing resource C, no dependency is created and the group’s graphical display is not modified.

It will be understood, that the method of FIG. 7 is not limited to adding dependencies. For instance, an existing dependency (represented by a directed line such as line 430A in FIG. 5) may be deleted by selecting it (e.g., by single clicking on 430A with a pointer device in a GUI) and then issuing a delete command via, for example, a key stroke command (e.g., delete) or a pop-up menu. The resulting action may be relayed to operating system 204 by resource management application 200 via calls to cluster API 202 and a graphical representation of the group’s modified dependencies generated by a method in accordance with FIGS. 3A and 3B.

In another aspect of the invention, a template may be defined to facilitate the detection and configuration of a network or cluster service. In this embodiment, a service template identifies those resources needed to support the service, a description of the dependencies between the service’s resources, standard service management details such as an alias name, fail-over and fail-back properties, and standard resource properties (which vary from resource to resource). Templates provide a means, and the necessary information, to automatically detect, aggregate, configure, and migrate (e.g., copy or move) resources so as to provide a service.

Referring now to FIG. 9, a method to automatically detect, group, and configure resources to provide a service is shown. First, a template describing a target service’s required resources, their properties and dependencies is acquired (block 900) – see discussion below -- and a network location is specified (902). In one embodiment, a user may specify a specific
node they want to establish the service on. In another embodiment, the user may specify a specific cluster and, in yet another embodiment, the user may specify a domain or network in which they want to establish the target service. In addition, the user may specify the search environment interactively (through a GUI, for example), or it may be specified by the template, or it may default to a standard location (the cluster executing the method, for example).

Once specified, the location is searched to determine if the specified resources exist and are available. In one embodiment, one or more API calls to the target location’s operating system may be made to determine a node’s available resources. If the resources identified in the acquired template exist and are available at the specified location (the ‘yes’ prong of diamond 904), the identified resources are aggregated into a group (block 906), intra-group dependencies are established in accordance with the template (block 908), properties of the individual resources within the group are set in accordance with the template (block 908), and the newly formed group (i.e., service) is indicated on the cluster in which it has been formed by, for example, an icon identifying the target service. The acts of aggregating and establishing resource dependencies and properties may be performed through API calls to the operating system hosting the resources. Following the act of indicating the newly formed service, or in the event that the resources identified in the acquired template do not exist or are not available (the ‘no’ prong of diamond 904), processing in accordance with the method of FIG. 9 terminates (block 914).

In one embodiment, templates are embodied in a text file that may be processed by resource management application 200. A template description file may, for example, provide a list of those resources to be included in a group to provide the target service, property settings for each of the identified resources, dependencies between the identified resources, and properties of the target service (e.g., the name of the service, fail-over and fail-back sequences, and target owner).

One benefit of this aspect of the invention is that a service defined by a template may be used to automatically establish services across multiple networks without the user (e.g., network manager) having to manually define the service on each node or cluster. This, in turn, may reduce the time, effort, and number of problems associated with installing and/or defining network services. For example, a method in accordance with FIG. 9 may be executed when a cluster is initially created to automate the service setup process generally performed by a network administrator. Another benefit of this aspect of the invention is that templates may be
used to copy or move a service (e.g., a group of resources having established dependencies and properties) from one node or cluster to another node or cluster.

Acts in accordance with FIGS. 3A, 3B, 7, and 9 may be performed by a programmable control device executing instructions organized into a program module. A programmable control device may be a single computer processor, a plurality of computer processors, one or more special purpose processors, or one or more custom designed state machines. Storage devices suitable for tangibly embodying program instructions include all forms of non-volatile memory including, but not limited to: semiconductor memory devices such as EPROM, EEPROM, and flash devices; magnetic disks (fixed, floppy, and removable); other magnetic media such as tape; and optical media such as CD-ROM disks.

While the invention has been disclosed with respect to a limited number of embodiments, numerous modifications and variations will be appreciated by those skilled in the art. It is intended, therefore, that the following claims cover all such modifications and variations that may fall within the true spirit and scope of the invention.
What is claimed is:

1. A computer executable method to indicate computer resource dependencies, comprising:
   selecting a first resource;
   determining a second resource, the second resource dependent on the first resource;
   determining a third resource, the first resource dependent on the third resource;
   displaying a graphical indication of the first resource, the second resource, and the third resource; and
   graphically indicating the dependence between the first and second resources and the dependence between the first and third resources.

2. The method of claim 1, wherein the method comprises a method to indicate resource dependencies in a clustered network environment.

3. The method of claim 1, further comprising:
   determining the second resource is dependent on the third resource; and
   graphically indicating the dependence between the second resource and the third resource.

4. The method of claim 1, further comprising:
   selecting one of the displayed graphical resource indications;
   modifying a property of the resource associated with the selected graphical indication;
   and
   displaying a graphical indication of the modified property.

5. The method of claim 4, wherein the act of modifying a property of the resource associated with the selected graphical indication comprises invoking an application program interface call to change the property of the resource.

6. The method of claim 4, wherein the method is executed by a first node in a first domain and the first, second, and third resources are associated with a second node in a second domain.
7. The method of claim 1, wherein the act of graphically indicating the dependence between the first and second resources and the dependence between the first and third resources uses a dependence tree.

8. A program storage device, readable by a programmable control device, comprising:
instructions stored on the program storage device for causing the programmable control device to
  select a first resource,
  determine a second resource, the second resource dependent on the first resource,
  determine a third resource, the first resource dependent on the third resource,
  display a graphical indication of the first resource, the second resource, and the third resource, and
  indicate graphically the dependence between the first and second resources and the dependence between the first and third resources.

9. The program storage device of claim 8, wherein the instructions to indicate graphically the dependence between the first and second resources and the dependence between the first and third resources comprise instructions to generate a columnar layout on a display, wherein the dependencies are represented by directed lines.
10. A computer executable method to indicate computer resource dependencies, comprising:
   selecting a resource group;
   determining resources associated with the selected resource group;
   generating a first list of resources from the resources associated with the selected resource group that have no upstream dependencies;
   generating a second list of resources from the resources associated with the selected resource group that depend only directly on those resources in the first list of resources;
   generating a third list of resources from the resources associated with the selected resource group that depend directly on those resources in the second list of resources;
   indicating each of the resources in the first, second, and third lists; and
   indicating direct upstream dependencies between the indications of the resources in the first and second lists, and between the indications of the resources in the second and third lists.

11. The method of claim 10, wherein the act of generating the first list of resources comprises:
   selecting a resource from the resources associated with the selected resource group; and
   issuing an application program interface call to an operating system, the application program interface call returning a list of resources that the selected resource depends upon.

12. The method of claim 10, wherein the act of indicating each of the resources in the first, second, and third lists comprises displaying graphical icons representing each of the resources.

13. The method of claim 12, wherein the act of indicating direct upstream dependencies between the indications of the resources in the first and second lists comprise displaying a directed line from a graphical icon of a resource in the first list to a graphical icon of a resource in the second list.

14. The method of claim 13, wherein the act of indicating direct upstream dependencies between the indications of the resources in the second and third lists comprise displaying a directed line from a graphical icon of a resource in the second list to a graphical icon of a resource in the third list.
15. The method of claim 14 further comprising displaying a directed line from a graphical icon of a resource in the first list to a graphical icon of a resource in the third list.

16. The method of claim 10, wherein the act of indicating each of the resources in the first, second, and third lists comprises displaying a textual name associated with each of the resources.

17. A method to establish a service, comprising:
   obtaining a template identifying resources, one or more dependencies between the identified resources, and properties associated with the identified resources;
   automatically identifying a node on which the identified resources are available;
   automatically establishing a resource group on the identified node, the resource group including the identified resources; and
   automatically establishing dependencies between the resources of the resource group as indicated in the template.

18. The method of claim 17, further comprising setting a property of a resource in the resource group as indicated in the template.

19. A program storage device, readable by a programmable control device, comprising:
   instructions stored on the program storage device for causing the programmable control device to
   obtain a template identifying resources, one or more dependencies between the identified network resources, and properties associated with the identified network resources,
   identify a node on which the identified resources are available,
   establish a resource group on the identified node, the resource group including the identified resources, and
   establish dependencies between the resources of the resource group as indicated in the template.

20. The program storage device of claim 19, further comprising instructions to set a property of a resource in the resource group as indicated in the template.
FIG. 1
(Prior Art)

FIG. 2
300
SELECT A RESOURCE GROUP

302
DETERMINE WHICH RESOURCES ARE ASSOCIATED WITH SELECTED GROUP

304
GENERATE LIST OF RESOURCES THAT HAVE NO UPSTREAM DEPENDENCIES

306
INDICATE EACH LISTED RESOURCE IN A FIRST COLUMN

308
GENERATE LIST OF RESOURCES THAT DEPEND ONLY DIRECTLY ON THOSE RESOURCES INDICATED IN IMMEDIATELY PRECEDING COLUMN

310
INDICATE EACH LISTED RESOURCE IN A NEXT COLUMN

312
MORE RESOURCES?

YES

NO

314
GOTO

FIG. 3A
FROM 312

314
COL. TWO
EXIST?

316
MAKE COLUMN TWO THE CURRENT COLUMN

318
INDICATE DIRECT UPSTREAM DEPENDENCIES FOR ALL RESOURCES IN THE CURRENT COLUMN

320
MORE
COLUMNS?

322
MAKE NEXT COLUMN THE CURRENT COLUMN

324
END

FIG. 3B
FIG. 6

FIG. 7

700

SELECT A FIRST RESOURCE

702

DRAG THE SELECTED RESOURCE ONTO A SECOND RESOURCE

704

MAKE SECOND RESOURCE DEPENDENT UPON THE FIRST RESOURCE

706

DISPLAY NEW DEPENDENCY
FIG. 8B
ACQUIRE TEMPLATE

DETERMINE SEARCH ENVIRONMENT

RESOURCES AVAILABLE?

AGREGATE RESOURCES INTO GROUP

ESTABLISH INTRA-GROUP DEPENDENCIES

ESTABLISH RESOURCE SETTINGS

INDICATE SERVICE AVAILABILITY

END

FIG. 9
## INTERNATIONAL SEARCH REPORT
### A. CLASSIFICATION OF SUBJECT MATTER
- **IPC(6):** G06F 3/00
- **US CL:** 345/356

According to International Patent Classification (IPC) or to both national classification and IPC.

### B. FIELDS SEARCHED

**Minimum documentation searched (classification system followed by classification symbols)**
- **U.S.:** 345/356, 339, 969, 970, 965, 966, 967
  - 345/223, 221, and 224

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched.

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used).

### C. DOCUMENTS CONSIDERED TO BE RELEVANT

<table>
<thead>
<tr>
<th>Category</th>
<th>Citation of document, with indication, where appropriate, of the relevant passages</th>
<th>Relevant to claim No.</th>
</tr>
</thead>
<tbody>
<tr>
<td>X, E</td>
<td>US 5,999,179 A (KEKIC et al) 07 DECEMBER 1999, abstract and fig. 37f</td>
<td>1-20</td>
</tr>
<tr>
<td>A, P</td>
<td>US 5,910,803 A (GRAU et al) 08 JUNE 1999</td>
<td>1-20</td>
</tr>
<tr>
<td>A</td>
<td>US 5,295,243 (ROBERTSON et al) 15 MARCH 1994</td>
<td>1-20</td>
</tr>
</tbody>
</table>

Further documents are listed in the continuation of Box C.

See patent family annex.

Date of the actual completion of the international search: 24 FEBRUARY 2000

Date of mailing of the international search report: 20 MAR 2000

Name and mailing address of the ISA/US Commissioner of Patents and Trademarks:
- Box PCT
- Washington, D.C. 20231
- Facsimile No. (703) 305-3230

Authorized officer: [Signature]

Telephone No. (703) 305-9600