The Prospero Distributed Operating System

Main Goal

As systems grow larger, it becomes harder for the user to organize, identify, and locate the information and services that are of interest. The Virtual System Model is an approach to organizing large systems that makes it easier for users to keep track of and organize such objects. Prospero is a distributed operating system based on the Virtual System Model.

Advantages

Prospero allows users to build their own "virtual systems" from the files and services that are available to them. It is easier for users to keep track of objects in the resulting name space. Tools are provided that allow a name space to be specified as a function of other name spaces. This relieves part of the burden of keeping name spaces up-to-date. Additionally, the tools that are provided allow information to be organized in ways that make it easier for other users to identify new information of interest to them, but which they have not yet seen.

A disadvantage of supporting multiple name spaces is the lack of name transparency: the same name may refer different objects when used by different people. Prospero addresses this problem by supporting closure.

Description

A *virtual system* defines a view of the world centered around the user. Those files and services of interest to the user have short names, while the names of objects that the user is less likely to access are much longer. Users can specify parts of their name space as a function of one or more other name spaces. This is accomplished using two tools: a filter is a user defined program, associated with a link, which changes the way one views objects seen through that link; a union link takes the contents of a linked subdirectory and makes them appear as if they are part of the directory containing the link.

Naming in Prospero is "object-centered". In object-centered naming, multiple name spaces are supported, and each object has an associated name space (forming a closure). By supporting closure, it is always clear which name space is to be used when resolving a name.

The global naming network forms a generalized directed-graph, but users don't see it as such. Instead, they see the name space associated with the active virtual system. Each virtual system specifies the node in the directed graph that is to be the root of its associated name space. When projected from that root, the name space appears to be hierarchical.

A prototype is presently implemented on top the Unix operating system. The prototype focuses on the file system and allows users to organize references to information scattered across Internet archive sites. Implementations are planned for other systems.

Objects that are accessible through Prospero may reside on systems of different types. The Prospero protocol makes extensive use of types, allowing full support for heterogeneity. Because Prospero is intended to scale across administrative boundaries, support for heterogeneity was an important requirement in its design.

Prospero runs as a network service. Each system that contains objects that are to be accessed through Prospero must run the server. The servers collectively implement a distributed directory service, independent of the mechanism used to actually store the objects. In addition to directory information, the directory service also stores information about individual objects. This information includes replication, closure, and access control information, a list of cross references, and a (possibly incomplete) list of back-links to the directories which name the object.

Client applications can run on any system, not just those running the server. No kernel changes are required. Several applications supporting the Prospero name space are provided and a library redefining the open system call allows existing applications to use the Prospero name space. A kernel implementation is planned. Such an implementation would avoid the need to relink applications, and would thus reduce the size of the resulting application binaries. The kernel and non-kernel implementations will interoperate, so users can choose whichever implementation is most suitable for their environment.

The Prospero protocol is presently implemented on top of UDP (in the TCP/IP suite of protocols). The protocol could be easily implemented on top of other protocols, and simultaneous support for multiple protocols has been made possible by the extensive use of types, especially in passing network host names and addresses.

Access to individual objects in Prospero is mediated by existing access mechanisms, and the security for such access is defined by those mechanisms. Access to read or write directory information may be specified by access control lists associated with directories, or with individual links within a directory. Prospero uses interchangeable authentication mechanisms. Authentication information is passed as an authentication type field, and a field for data specific to that type. The prototype uses an authentication type of "assertion", a future release will incorporate Kerberos authentication, and other authentication mechanisms may be added as needed.

Objects in Prospero may move around, and the links to the object will continue to work. This is accomplished through a combination of forwarding pointers, callbacks, and timeouts.

Scalability was a key consideration in the design of Prospero. Prospero is intended to tie together pieces from numerous systems that are geographically distributed and which cross organizational boundaries. In addressing scale, Prospero is not just concerned with the demands placed on the system. The demands placed on the user are just as important. The user-centered view supported by Prospero is one way of limiting those demands.

Virtual systems may be set up by users or organizations. They may be created for individuals, projects, or any other purpose for which a separate view of the world might be useful. It is expected that organizations will maintain directories organizing objects in different ways, and that these directories can be incorporated into the virtual systems of those that need the information. Among these directories might be indices by author, project, subject, or any other key fields. Users can find objects by looking for them in the appropriate index, or by browsing through related virtual systems.

Miscellaneous

STATUS: A prototype is running on 40+ systems in 7 countries on three continents. A beta release is available. Those interested in being a beta test site should contact INFO-PROSPERO.

MAILING LIST: INFO-PROSPERO@CS.WASHINGTON.EDU is a mailing list on which announcements will be made as new pieces of the system become available. To be added to the list, send a request to: INFO-PROSPERO-REQUEST.

CONTACT: B. Clifford Neuman
Department of Computer Science and Engineering
University of Washington, FR-35
Seattle, Washington 98195

bcn@cs.washington.edu
+1 (206) 543-7798

References

[1] B. Clifford Neuman. The need for closure in large distributed systems. *Operating Systems Review*, 23(4):28–30, October 1989.

[2] B. Clifford Neuman. The Virtual System Model for large distributed operating systems. Technical Report 89-01-07, Department of Computer Science, University of Washington, April 1989.

[3] B. Clifford Neuman. Workstations and the Virtual System Model. In *Proceeding of the 2nd IEEE Workshop on Workstation Operating Systems*, pages 91–95, September 1989. Also available as University of Washington Technical Report 89-10-10. Also appears in the *Newsletter of the IEEE Technical Committee on Operating Systems*, Volume 3, Number 3, Fall 1989.

[4] B. Clifford Neuman. Managing replicated data within the virtual system model. Department of Computer Science and Engineering, University of Washington, May 1990.

[5] B. Clifford Neuman. The Virtual System Model: A scalable approach to organizing large systems (a thesis proposal). Technical Report 90-05-01, Department of Computer Science, University of Washington, May 1990.