UNIVERSITY OF SOUTHAMPTON

DEPARTMENT OF ELECTRONICS AND COMPUTER SCIENCE

AN OPEN FRAMEWORK FOR INTEGRATING WIDELY DISTRIBUTED HYPERMEDIA RESOURCES

Stuart Goose, Jonathan Dale, Gary J. Hill,

David C. DeRoure and Wendy Hall

University of Southampton

June 1996

Presented at ICMCS'96

An Open Framework for Integrating Widely Distributed Hypermedia Resources

Stuart Goose, Jonathan Dale, Gary J. Hill, David C. DeRoure and Wendy Hall

Multimedia Research Group

Department of Electronics and Computer Science

University of Southampton

UK

{sg93r, jd94r, gjh, dder, wh}@ecs.soton.ac.uk

The success of the WWW has served as an illustration of how hypermedia functionality can enhance access to large amounts of distributed information. However, the WWW and many other distributed hypermedia systems offer very simple forms of hypermedia functionality which are not easily applied to existing applications and data formats, and cannot easily incorporate alternative functions which would aid hypermedia navigation to and from existing documents that have not been developed with hypermedia access in mind. This paper describes the extension to a distributed environment of the open hypermedia functionality of the Microcosm system, which is designed to support the provision of hypermedia access to a wide range of source material and application, and to offer straightforward extension of the system to incorporate new forms of information access.

1. Introduction

The use of hypermedia-based navigation as the basis of a rich, distributed information environment is now widely accepted, as illustrated by the success of the World Wide Web (WWW) [2]. However, the hypermedia model used by the WWW and other contemporary systems are very simple, offering only limited linking functionality and oriented towards the delivery of information authored specifically for use with in these systems, rather than providing a range of services to people working with distributed information.

The system described in this paper aims to provide a flexible and extensible system for the provision of distributed hypermedia functionality, which will allow users to exercise control over the way in which they choose to share information. The model is a development of that used by the Microcosm system [8,9], an open hypermedia framework developed at the University of Southampton. The Microcosm system is based upon a set of autonomous, communicating processes which together provide complete hypermedia functionality. This modular system allows straightforward introduction of alternative functions and is clearly well suited to distributed operation.

The flexible hypermedia architecture that we have developed extends the current Microcosm architecture to support scaleable, distributed operation, whilst retaining the extensibility of the existing system. Unlike most distributed hypermedia systems, it is designed to provide an underlying hypermedia layer to a user's preferred environment. The system will allow hypermedia functionality to be accessed from a wide range of information resource bases, provide access to shared resources made available by other users and allow users to integrate distributed resources into a single, cohesive architecture. We will show how the model developed will support collaboration and information sharing between users, and describe how the distributed architecture may be extended to support a wide range of multimedia types, from simple text material to network intensive media, such as sound and video.

2. Existing Distributed Hypermedia Systems

Although a number of distributed hypermedia systems have been developed, few combine extensible hypermedia functionality with a flexible approach to managing and sharing widely distributed information. As networks become increasingly important for the dissemination of information and the range of information services available becomes increasingly diverse, there is an developing need for the provision of information management tools which can provide integrated access to a full range of resources [15]. Such systems should allow non-intrusive linking across arbitrary document formats and provide a user community with tools for accessing and structuring information according to their individual requirements. Hypermedia navigation and structuring can provide the basis for such manipulation of information, but existing systems are not suited to widespread use due to the reliance on simple hypermedia functionality and proprietary browsers and formats.

Many existing systems, such as the WWW, although open in terms of the platforms and protocols supported and the ability to handle distributed operation, do not explicitly support a flexible hypermedia model. The WWW in particular suffers due to its reliance on the Hypertext Mark-up Language (HTML) [3] for the provision of linking. This prevents hypermedia navigation from taking place within documents stored in other formats. The WWW hypermedia model provided by HTML is restricted to links marked up in documents and as a result, it is difficult to provide linking as a third party activity, and thus it is very difficult to seamlessly extend the available functionality. In fact, it is possible to implement a variety of more advanced functions for the WWW community through the use of CGI scripts that execute on WWW servers, but this is restricted to dynamic generation of HTML-based documents. The latest generation of WWW browsers promise further advances through the use of executable objects such as Java applets [21], and third party extensions such as the Distributed Link Service [6] offer alternative linking facilities, but this approach is not fully satisfactory as a basis for a seamless information system.

The Hyper-G system [1] whilst superficially similar to the WWW, offers some advantages due to the separation of link information into link databases. This facilitates the application of links to a wider range of document formats, for example images or postscript documents, in a common manner. The hypermedia model supported, like that of the WWW, is based on simple point-to-point links, although it is augmented by an integrated search engine. The Hyper-G server design provides some degree of scaleability through the use of replication across servers and its use of separate link management promotes the maintenance of link integrity, but it does not provide general extension facilities to include alternative link models or to incorporate other information systems.

Hyperform [23], goes further by providing a fully extensible hypermedia server, or hyperbase. This provides a set of object classes to allow the hypermedia services provided to be tailored to particular requirements and facilities to integrate external tools into the system. This clearly provides scope for the development of a wide range of hypermedia functionality. The integration facilities suggest that inter-operation between Hyperform servers would also be possible, but this is not explicitly supported.

The Virtual Notebook System (VNS) [5,19], provides an open hypermedia architecture based on the use of an underlying database system, thus allowing the development of custom programs that utilise the hypermedia facilities. The system also supports interaction between different VNS servers, allowing workgroups to be supported by dedicated servers whilst still permitting the sharing of information between individual workgroups. VNS provides less scope for extending the hypermedia model, and integrating other information services than Hyperform, but has better support for controlling shared access to information.

Although all the systems described support some form of distributed operation, none currently provide the degree of flexibility which a distributed version of Microcosm could provide. Initial work on a distributed version of Microcosm [12,13] refined the existing communication architecture to demonstrate the feasibility of fine-grained distribution of hypermedia functionality. This system enabled users to 'publish' their local Microcosm processes (link databases, search engines, etc.) so that they could be incorporated into remote users' configurations. It allowed a wide range of distributed configurations to be supported, but was best suited to small scale, local sharing of information. It lacked the ability to manage logical configurations of processes which would enhance the scaleability of the system and did not have facilities for one user to easily manage processes spread across a range of systems.

It can be seen that the majority of hypermedia systems have failed to address the needs of information sharing and interchange on a wide scale and as part of the general process of working with information. Users are typically forced to access information resources in turn and there is no support for querying a selected array of resources or for augmenting these resources with additional hypermedia structures. As a consequence, information providers utilising current systems have been conditioned to operate in isolation, thus creating an array of information islands.

Popular systems, such as the WWW, have illustrated a demand for global access to information, and for many people the WWW also served as an introduction to the concept of hypermedia; demonstrating its potential in providing an integrated and unifying paradigm for information organisation. As demands on the WWW grow, it is becoming apparent that the simple hypermedia model is not adequate for large scale information management, navigation and discovery. Open hypermedia systems, such as Microcosm, offer more flexible facilities that can overcome these problems. The following sections describe the distributed model for Microcosm that has been developed, the way in which it can facilitate more flexible access to distributed information, and the initial development of a system which implements this model.

3. Elevating Hypermedia to a Distributed Environment

In this section, we describe the introduction of a new hypermedia model that can provide users with access to information resources which can be distributed across wide area networks. This abstract model provides methods for allowing users to define resources that encapsulate information within defined and manageable entities and for allowing users to reuse them in their own information resources.

3.1. A Reflexive Model for the Construction of Distributed Hypermedia Applications

A *hypermedia object* is the smallest element that is managed by a hypermedia framework, that is, a document or a process. This is the model that has been adopted by the WWW; everything is considered to be a document and even processes themselves (apart from the WWW infrastructure) exist to generate other documents. Because the granularity of this system exists at the document level, it is not difficult for users to bind documents through links into sharable information resources, but it is difficult to logically group these individual resources into sharable components that can be associated with larger repositories that are organised by context.

To provide a model that retains the desirable properties that exist at the document level and that also allows hypermedia objects to be associated with larger information resources that possess context, the concept of a *hypermedia application* was introduced. A hypermedia application is an entity that comprises a resource base of hypermedia objects (documents, link databases and processes). Documents can consist of any media type, while link databases describe the link structure that exists between documents and processes define the set of operations that can be performed on the information. To allow users to share information between each other, applications possess the characteristic of reflexivity, that is, applications can make references to other applications. In this way, users can make references to other applications and associate them with their own hierarchical structures to build information resources which can seamlessly comprise a number of diverse and distributed applications.

Before a hypermedia application can be considered for publishing to a wider environment, a number of distributed issues need to be addressed:

- *Ownership*. Hypermedia applications should be able to be attributed to a particular user, or set of users, who are considered to be its authors. These owners are responsible for managing their applications and making them available to the global community.
- *Sharing*. Applications should be sharable between users, to facilitate information dissemination, reuse and co-operative working.
- *Permissions*. The interactions between users and applications need to be protected by a set of permissions that allow users to control access to the applications that they have made sharable. The richness of this permission set will determine how easy and flexible the system is at sharing information.

Applications can be imported into a *hypermedia session* to allow users to construct larger information resources. Users can have multiple sessions which represent the information resources that they are interested in, either for work or recreational purposes. These collections permit users to apply group operations to logical sets of information, for example, issuing queries to all applications within a session.

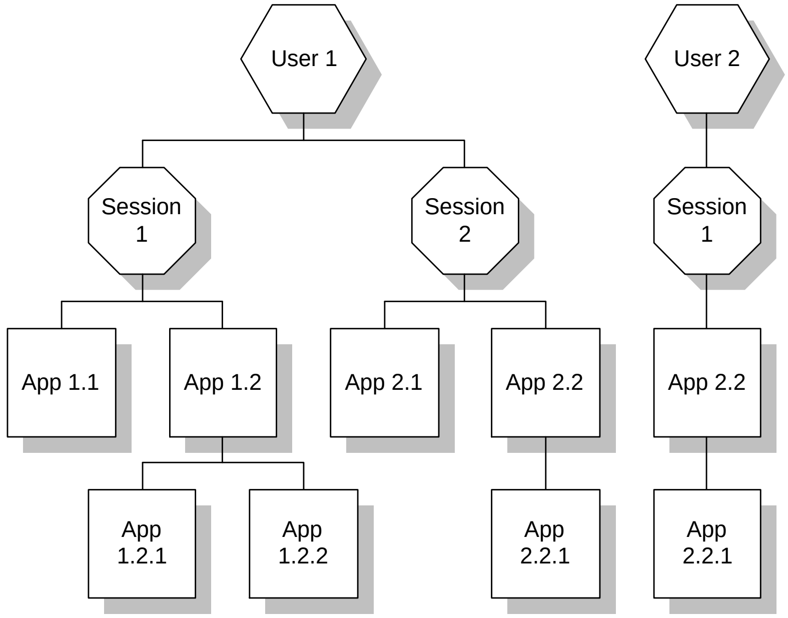

Figure 1 : Application Integration and Sharing in a Distributed Environment

The hypermedia framework proposed is illustrated in figure 1. *User 1* has set-up a number of sessions within their environment and at the top level, *session 1* contains two applications; *application 1.1* is local and has been authored by *user 1* (denoted by a solid box) and *application 1.2* is remote and has been imported from another user (denoted by a dashed box). *Application 1.2*, in turn, contains two more applications; one which is local and one which is remote to this application. Although *user 1* has four applications in *session 1*, *application 1.2.1* and *application 1.2.2* are presented transparently to the user as *application 1.2*. This means that when *user 1* makes use of the resources in either *application 1.2.1* or *application 1.2.2*, they perceive that they are utilising the resources of *application 1.2*. This is advantageous because it abstracts communication and connectivity issues from the user, who can concentrate on building hypermedia applications and creating link structures between them.

Application 2.2 in figure 1 is present in two forms; a shared view from the perspective of *user 1* (maybe a co-author) and an owner's view from the perspective of *user 2* (probably the author). *User 1* has developed *application 2.2* and *application 2.2.1* and has made it sharable, through some mechanism, via *application 2.2*. *User 2* has decided that they wish to make use of this application and have incorporated it into *session 2* of their environment. The operations that *user 1* can perform on this application will be determined by the permissions that have been exported with it.

An example of this abstract model in context is give in figure 2. *Encyclopaedia* is a session that is comprised of many applications which describe its behaviour, both in terms of information content and functionality. Such an application is a dynamically extensible entity within which new applications can be plugged or unplugged at anytime. Because the application is not monolithic, different interpretations can be generated from restricted or expanded views on the same sets of applications. For example, a *Junior Encyclopaedia* might reuse some of the constituent applications of *Encyclopaedia*, but restrict other applications that are considered to be of either an adult nature or are too advanced for children to understand.

This hypermedia model provides for a very flexible and extensible model that allows a user to author multiple and different views on the same sets of applications, depending upon the context within which they are being placed. Moreover, these contextual hypermedia applications can be shared across a number users, thus promoting information dissemination and reuse on a global scale.

3.2. A Distributed Framework for Integrating Hypermedia Applications

The mapping between the abstract hypermedia model into a physical representation is achieved through an event-driven process framework. A physical hypermedia application realisation tightly encapsulates a collection of processes (and associated preferences and configurations), an arbitrary

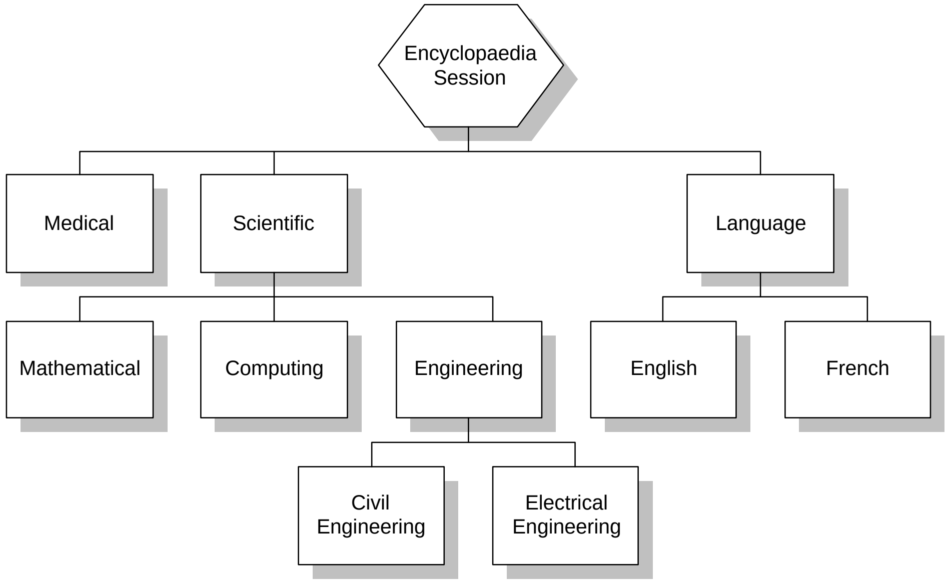

Figure 2 : Application Integration to Form Distributed Hypermedia Resources

amount of data together with the optional characteristic of reflexivity, and can be expressed as:

application ::= {process} x {data} x [{application}]

Such an application can itself be considered as a self-contained, logical entity and the application paradigm provides an abstraction for modelling communities of co-operating processes. Each discrete process publishes a suite of services, usually operates on specified data objects and responds to a designated set of events. As an application comprises executing processes it can be considered as a dynamic object which can respond quickly to those events that occur within the system.

Due to their well defined and modular nature, such processes predispose themselves to being distributed over interconnected networks. To deliver this reflexive model for hypermedia applications, provision for flexible communication between widely distributed asynchronous processes is essential.

3.3. The Heterogeneous Communication Model

The Heterogeneous Communication Model (HCM) [10] is an open framework that we have designed to facilitate the tight integration of asynchronous processes so that they can function effectively within a heterogeneous, distributed, multi-user environment.

The key to the success of systems composed in this manner lies in the close co-operation between individual processes or communities of processes. As these processes arbitrate over the work to be performed, the importance of the communication layer connecting them becomes increasingly apparent. Such systems have an inherent reliance upon the underlying communication substrate and it is therefore crucial that this layer exhibits a range of desirable qualities:

- *Scaleability*. A crucial characteristic of a scaleable architecture is that the addition of processes to the system has minimal impact upon the overall performance. Another vital scaleability issue specific to the communication architecture is that of message routing; the number of intermediate steps travelled by a message should remain constant regardless of the number of processes present within the system. A scaleable and reliable communication architecture also affords a high degree of confidence regarding the behaviour of the entire system and can aid in ensuring predictable transmission performance.

- *Openness*. For distributed systems to capitalise upon under utilised resources, the issue of heterogeneity has to be addressed. It is not unusual for contemporary computer networks to connect a multitude of platforms running a variety of operating systems, but by conforming to a core set of standards, using only a common set of services and by observing published protocols, heterogeneity can be achieved. Moreover, openness allows the system to integrate with other third-party applications, thus allowing applications to benefit from improved interconnectivity and interoperability.

- *Flexibility*. By providing an open framework promoting a "plug and play" rationale, the functionality of the system can be dynamically augmented or diminished through the addition or removal of further processes. Because the essential functionality of the system is contained within these modular components, users can develop and introduce new processes to extend the system in directions that meet their requirements. Additionally, this modularity creates future-proofing, since the system can be dynamically reconfigured to incorporate new developments and new protocols.

By incorporating these characteristics into the design of the HCM, we have sought to embody many of the requirements of a distributed communication layer and to provide an open framework above which a powerful distributed hypermedia system can be developed. The HCM architecture is intended to provide a generic solution for message passing across machine boundaries, but the ability for processes to negotiate individual quality of service requirements, for example delivering video over the network, is also accommodated within the layered architecture and is discussed later.

3.4. Process Addressing

For direct communication to take place, one process must be able to uniquely identify or address the process where it wishes to send a message. The addressing scheme used within the HCM is based upon a metaphor found commonly in computer filing systems. Each file in a filing system can be uniquely addressed by using the fully qualified path name, for example, /papers/multimedia/ieeemm96.eps. It is possible to issue a query along with some criteria that returns a list of matching file names. The matching criteria is generally expressed by using *wildcards* and regular expressions. A subset of these techniques have been combined to produce a flexible process registration and addressing mechanism. Although the filing system metaphor may imply that process registrations are hierarchical, the addressing scheme does not impose any rigid structure.

Each participating process providing services can effectively publicise their availability by registering them with the HCM infrastructure. Following this action, any permitted process within the framework can issue a message requesting such a service. The process addressing scheme forms the foundation upon which the support for message passing between processes executing on different machines is based. The first major benefit of the addressing scheme is that it can be customised by the architects of the host system to reflect the context in which it operates. This means that a process can send a message identifying the intended recipients using terms that have meaning within that context. The second key benefit is the wildcard attribute, which aids a process in disseminating a message to a wider audience. By placing a wildcard in one or more positions within the destination process address, a message can be targeted at a specific community of processes.

For example, a host system may require all process registrations to conform to the following custom addressing template:

/User/SessionId/HypermediaApplicationName/ProcessName/ProcessId/DocumentObject/Service

Using this template, two link database processes could each register, or advertise, two service entries in the following style:

/user1/152.78.64.128.11/projects/linkbase/152.78.64.64.22/progress.links/follow.link
/user1/152.78.64.128.11/projects/linkbase/152.78.64.64.22/progress.links/create.link

/user2/152.78.64.128.12/personnel/linkbase/152.78.64.64.25/staff.links/follow.link
/user2/152.78.64.128.12/personnel/linkbase/152.78.64.64.25/staff.links/create.link

In order to submit registration entries to the HCM, these two link database processes have specialised each field of the custom template outlined above. The registration entries disclose that there are two users each with a session. User 1 has a hypermedia application called *projects*, within which is a linkbase process operating upon the link database progress.links and offering two services follow.link and create.link. User 2 has a hypermedia application called *personnel*, with a linkbase process operating upon the link database staff.links and advertising the same services.

Consider if user 1 is a project manager and user 2 is a personnel manager, and the project manager wishes to requisition information regarding the particular skills of their staff matched against the skills required for future projects. From a document viewer, the project manager could select a list of skills and apply the search to both the *projects* application and the *personnel* application. This message request could be sent to all applications within the project manager's environment and onto every link database processes that services follow.link requests. The following destination process address (where * represents a wildcard) achieves this:

/*/*/*/linkbase/*/*/follow.link

It has been shown how the generic process addressing scheme can be tailored to model the domain in which the target system is to operate. This affords the system architects the total freedom to assign simple or complex semantics to a custom template. This means that the granularity at which process may interact is completely configurable. The added flexibility provided by the wildcard feature allows a process to accurately define the scope of the audience over which a message is to be directed. This facilitates conversations between processes to be conducted on a variety of levels, dependent upon the richness of information incorporated within the addressing template. As can be appreciated from the above example, messages could be exchanged between processes on a process name basis, a document basis or a service basis etc., or more usefully, combinations of all the fields.

For inter-process communication to occur between two distributed processes, one process must initiate the request by contacting the other. This requires that the initiating process is aware of the location of the remote process. Considering this scenario within the context of TCP/IP, the initiating process would have to know both the Internet address of the machine where the remote process is executing and the port number on which it is listening for network activity. The process addressing mechanism introduced above affords a high degree of process location transparency. By using the HCM as a communication layer, an initiating process can identify and target the services of a remote process in a meaningful context, as opposed to first determining and then supplying the network addresses of remote processes.

3.5. An Architecture to Support Widely Distributed Process Interaction

One of the key principles of distributed computing is that any processing need not be performed locally, but on the most appropriate machine available. It was decided that the design of the HCM would embody this principle from the outset and the infrastructure to support this activity required three core components.

A central module, called a *router*, is responsible for co-ordinating all communication between participating processes wishing to pass messages across machine boundaries. The router acts as a bureau where processes, using the addressing scheme as the vehicle, can dynamically advertise and withdraw their services and also post messages to other registered service providers.

A *process manager* is also started with each session to govern load balancing across the network and the distributed invocation of processes. Via its user interface, processes can be remotely configured and managed.

A single *daemon* process exists to serve the local network *domain* (an administrative term that is used to describe a set of logical machines where HCM processes can execute). Due to the fact that processes can execute anywhere within the local network, routers are required to register their network location with the daemon. As the daemon provides the sole, fixed point of contact within the domain, processes can query it to determine the network location of routers. The dynamic allocation and registration of network connections removes any explicit network location dependencies from processes within the HCM infrastructure.

The HCM supports a variety of alternative configurations:

- *Single user environment*. Where processes only communicate with processes that belong to the same user.
- *Multi-user environment*. Where processes belonging to one user may communicate with the processes of another user within a single domain.
- *Distributed, multi-user environment*. Where processes belonging to one user may communicate with the processes of another user across widely distributed domain boundaries.

A conceptual view of the third configuration is presented in figure 3. Here, *user 1* in *domain 1* initiates an action causing the daemon of *domain 2* to be contacted, requesting a communication transaction between

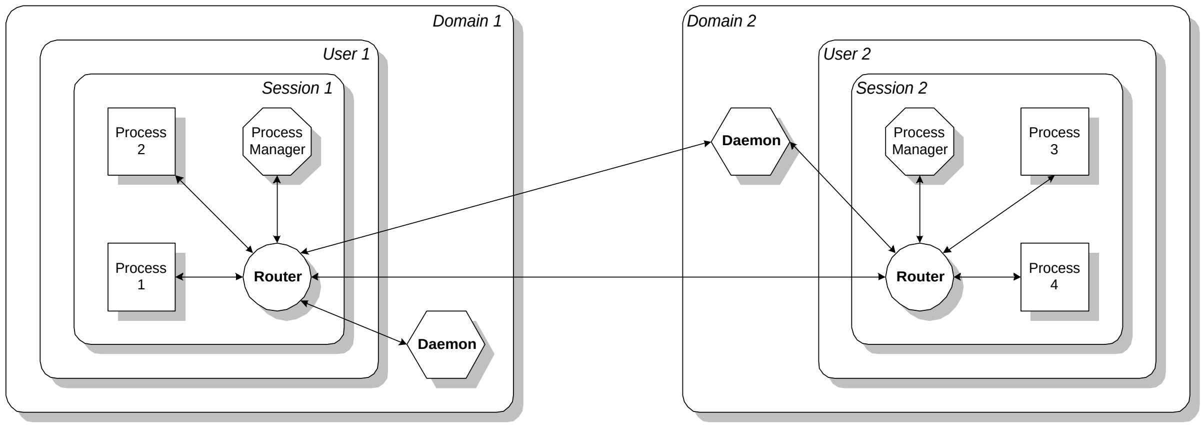

Figure 3 : Interaction Between Processes in a Distributed, Multi-User Environment

the two routers; once a virtual connection has been established, negotiation and information transfer can occur. The process registration entries declared as *published* (that is, services that are available to other users) are then exchanged between the two routers. This activity alleviates superfluous message passing between routers. The scaleable framework for providing widely distributed, peer-to-peer communication can now be exploited.

3.6. Microcosm: The Next Generation

A successful open distributed hypermedia system can be characterised by a scaleable architecture which is inherently distributed. While the architects of distributed hypermedia systems have addressed the issues of providing and retrieving distributed resources, they have often neglected to design systems with the inherent capability to exploit the distributed processing of this information. This section describes the construction and use of an open hypermedia system concerned equally with both of these facets.

Microcosm: The Next Generation (Microcosm TNG) is a prototype that retains the core philosophy of the Microcosm system and has been developed in response to the driving requirement for extending the principles across a distributed environment. Within the Microcosm hypermedia model there are a number of core elements that work together to provide a minimum level of hypermedia functionality:

- *External link storage*. The concept of link separation is well documented and its advantages clear when applied to open hypermedia [1,9,20]. A *linkbase* process maintains the link database and handles all link queries and creations. Multiple linkbases can exist to break document link connectivity into logical groupings.
- *Presentation and browsing*. Displaying links and allowing users to browse through links is handled by a range of *viewers*, primarily one for each class of media (text, bitmap, sound, etc.). Viewers provide the first-level interface to the user and let the user follow links and submit link queries.
- *Link selection*. The results of link queries that have been submitted by a user are handled by the *available links* process. This presents the links to the user for selection and then invokes the appropriate viewer to render the document.
- *Link authoring*. To allow links to be created within a link database, an intermediary process (the *linker*) performs pre-processing on the new link to determine its details (for example, the start and end anchor positions, the type of the link, the description of the link, etc.). Once this information has been collected, the completed link is forwarded to the relevant linkbases for creation.
- *Document organisation*. The document database (or *docuverse*[1]) was introduced to provide a layer of abstraction between the documents themselves and the filing systems upon which they reside.

The Microcosm TNG prototype has been developed using the HCM and has taken the core concepts identified previously and modelled them as discrete, distributed processes. These processes (linkbases,

[1] This term was first used by Ted Nelson when describing the Xanadu system [17].

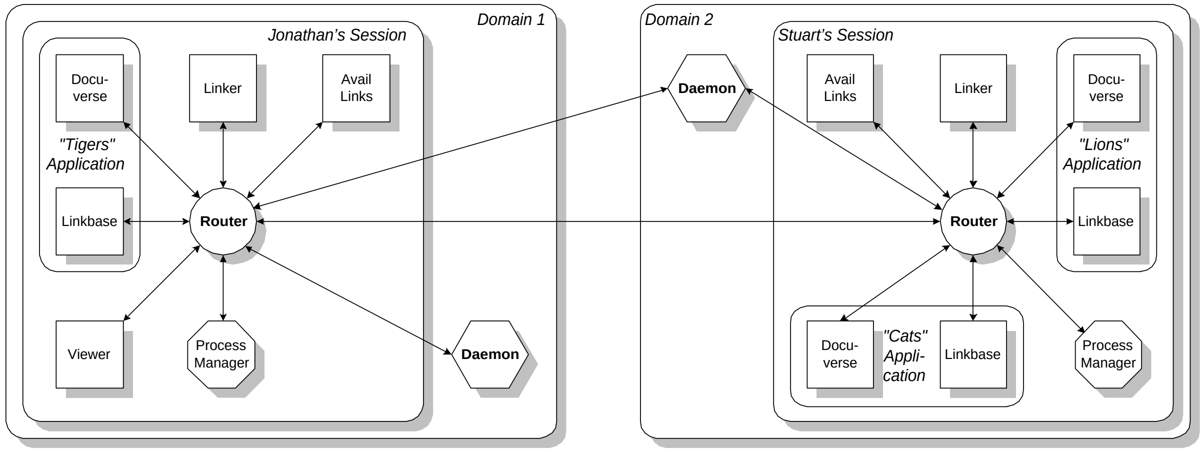

Figure 4 : Microcosm TNG Users Sharing Applications Across a Distributed Environment

viewers, available links, linkers, docuverses, etc.) were described in terms of the messages that they send and receive and the processing that they perform. Processes are considered to be peer entities due to the fact that they can communicate with other processes asynchronously. The distribution of both processes and messages is handled by the HCM subsystem. In this way, the naming of a process is kept independent of its location.

Resources are distributed through the application entity to which they belong. Once a user has authored an application, they can publish it to the world at large. The Microcosm TNG system provides a mechanism for allowing remote applications to be transparently included within a user's session. Also, since an application can comprise both data and processes, this inclusion not only enhances the functionality of the user's session, but also increases their available dataspace; all actions (for example, link queries) will be automatically propagated to all of the user's local *and* remote applications.

In figure 4, for example, the two users, Jonathan and Stuart, both have sessions running in their respective domains, which could be distributed across the Internet. In a strict local session, all link queries and link creations only apply to the applications that are bound to a particular session (for Jonathan, this is the *Tigers* application). Jonathan and Stuart can create their applications in isolation, importing local documents and media, and making links between them.

However, imagine that Stuart wishes to create a new application, called *Cats*, which comprises both the *Tigers* and *Lions* applications. This new application is made in his local session and he forms a connection to the *Tigers* application through domain the address of Jonathan, which he obtained previously via a resource discovery agent. Once connected, through the HCM subsystem, Stuart has access to both applications and he can create links between them and import new documents as appropriate.

Upon its completion, the *Cats* application can then be published for other people to connect to and can be reused in a similar fashion as indicated previously. It is important to note that when users access this application, connection to the *Tigers* and *Lions* applications, subsequent communication and the manipulation of hypermedia links are handled transparently.

From the work that we have undertaken so far, the distributed open hypermedia system that has been developed using our abstract hypermedia model above the HCM shows great promise for the future. The prototype has illustrated that the distribution of users, data and processing is possible and advantageous, and that future work lies in extending the hypermedia functionality into information sharing and reuse of applications on a global scale.

4. Service Layers

The provision of integrated and extensible multimedia handling within Microcosm TNG is provided through two support service layers (figure 5); a *presentation layer* and a *quality of service layer*. The presentation layer is responsible for determining how different media formats are managed and presented dynamically and consistently to the user across multiple platforms, and the quality of service layer is responsible for negotiating bandwidths with the underlying network layer to support the

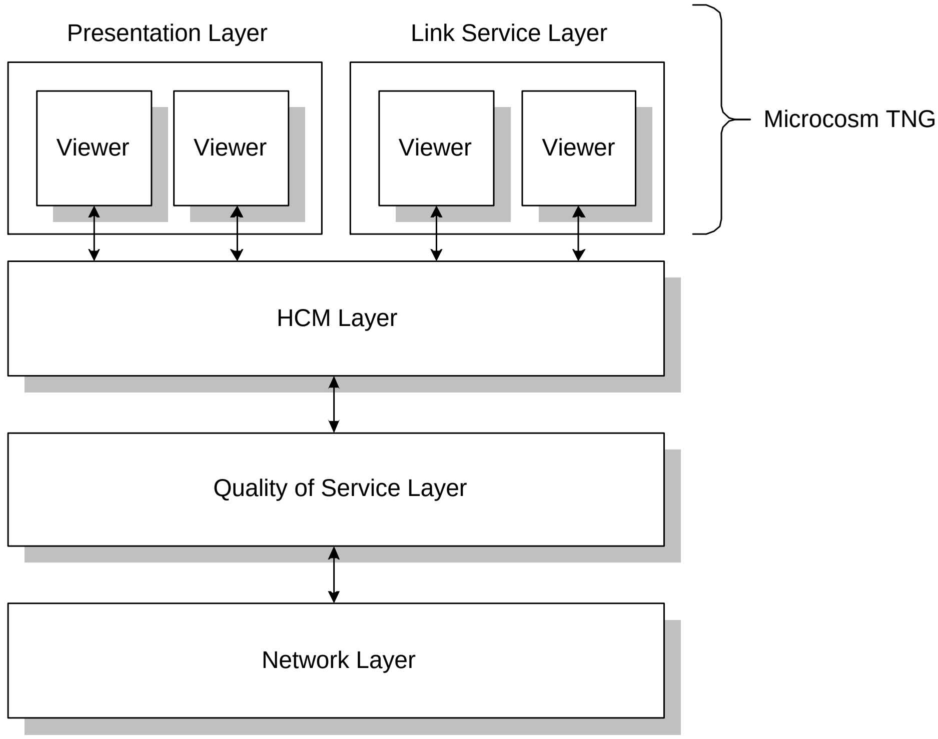

Figure 5 : Service Layers to Support Multimedia Extensibility

transport requirements of the different media formats.

4.1. Heterogeneous Presentation Layer

A viewer is responsible for providing an interface for allowing a user to manipulate a particular data type and also to create and traverse links. The flexible architecture of Microcosm facilitates the easy integration of new applications; through the introduction of new viewer modules, extensible multimedia support has been achieved. Although a trivial task to integrate the array of existing Microcosm media viewers within Microcosm TNG, moving into a distributed environment forces the issue of heterogeneous user interfaces to be confronted.

Through the careful engineering of portable code, a process can be implemented enabling it to be compiled for multiple platforms. Yet, it is very difficult to apply the same techniques for the implementation of user interfaces as no Graphical User Interface (GUI) standard is supported across the majority of machine platforms. Furthermore, although GUIs are based around the same windowing and mouse-driven metaphor, each possesses a differing set of features and functionality. As user interface designers tend to exploit such features, the likelihood of producing portable user interfaces is reduced.

A solution to this problem that we have developed is to split a process into two separate, but communicating physical processes; a Process Handler (PH) and an Interface Handler (IH). The PH deals with the generation of user interfaces and the processing of data, and the IH deals with choosing a suitable viewer for a given media format and handling user input data from the viewer and pre-processing it into a format suitable for the PH.

A prime requirement of any GUI-based process is the ability to either construct new user interfaces dynamically in response to certain data conditions, or to allow user interfaces to include dynamic data. Therefore, the construction of a user interface must occur within the PH to give the system the flexibility needed to be able to construct interfaces dynamically. As the IH is now unaware of the data that passes through it, it can be reused for each PH, thus decreasing the development time for processes and systems.

To abstract the PH from specific interface details, a platform and GUI independent display description language is required. This enables the PH to build and subsequently communicate new user interfaces to the IH in terms that are descriptive but abstract. HTML is such a language that is currently in widespread use within the WWW. HTML was created to allow user interfaces to be rendered in a GUI independent manner, and, by subsuming a core subset of the windowing metaphor (buttons, text boxes, list boxes, etc.), it provides a flexible and consistent set of display components. When a user interface is to be displayed, an HTML viewer (such as Netscape or Mosaic) can be invoked by the IH to render the interface into GUI-dependent terms. This means that only the HTML viewer needs to be GUI-dependent; both the PH and the IH can remain interface independent and possess the ability to execute anywhere within the local domain.

The use of HTML means that processes can be developed far more rapidly, since less programming effort is required in producing the interfaces. However, due to HTML's reliance upon the configuration abilities of the viewer that is rendering it, there exists a lack of seamless integration for new media formats. Netscape, for example, only recognises a core set of data types inherently; application associations with unhandled media formats require a third-party viewer to be invoked when these media formats are encountered. This has the advantage that Netscape can handle a range of data types, but means that there is a lack of cohesion within the system as a whole.

A superior level of integration is currently under investigation using Java technology. By providing a generic media viewer that can work within the Hot Java browser and is aware of the underlying hypermedia layer, each new data type can be handled by a separate applet. Such an applet needs to possess enough information about a media format to be able to render the data and also to present an manipulation interface to the user. For example, if a document contained AVI video information, then the Hot Java browser would locate the applet code to handle the AVI media format, which would dynamically incorporate it into the functionality of the browser and render it. Upon selection, an interface would be presented to the user with suitable controls to allow them to control the playback of the video data (play, pause, fast forward, etc.).

4.2. Synchronisation Support

Various researchers have confronted the issues of specifying and delivering synchronised multimedia documents, for example, [11,4,22]. This is an important area for the future development of multimedia information systems. Of the many proposed solutions to this non-trivial task, the most favourable is to provide support amongst the layers of the operating system. Existing mechanisms such as semaphores, message passing and clock scheduling have proved inadequate on their own, so mechanisms such as triggering have been introduced to complement these existing services. These problems are made worse when considering what types of mechanisms are required for synchronous co-ordination within a distributed environment.

Alternative high level approaches may be sought through designing the media viewers in such a way that they may communicate and hence, collaborate. A generic or composite media viewer could assume responsibility for playing and rendering all media formats, thus giving it absolute control over the rate at which each medium may progress. Research is currently being followed to extend the IH to act as a composite media viewer that directs the synchronisation of the different media format viewers from a script describing the interdependencies that exist between them

4.3. Quality of Service Layer

The guaranteed delivery of continuous media across a network is an issue that serious multimedia information system designers must be concerned with. The multimedia and networking research community have been addressing this central problem for a significant period and over this time have proposed many worthy solutions [14].

Microcosm TNG was developed to enhance the scaleability, reliability and performance of the hypermedia system through distributed processing practices. It is envisioned that the technologies alluded to above for providing negotiation and guaranteed bandwidth delivery, can serve the higher layers of our model.

Preliminary experimentation is also currently being conducted to establish how best exploit the advantages of ATM within the context of a layered communication architecture.

5. Future Work

The initial implementation of the model described in the previous sections has proved the viability of the communication architecture and the Microcosm TNG prototype has allowed preliminary experimentation. This prototype currently supports multi-user interaction and application sharing across a network, but users are restricted to a single session. As the prototype system is extended and refined, there are a number of additional areas that need to be addressed:

- *Security*. In a system where users are able to add and update information in shared resources, some form of security is necessary in order to prevent unauthorised access to information. When users publish information resources, they need to be able to specify who is to be allowed access to a hypermedia application and what degree of access they are allowed. As has been stated when describing the distributed hypermedia model, the richness of the permissions system will determine how flexible the system is at sharing hypermedia applications between users. Security needs to be addressed at the user level (how users allow applications to be shared with other users), the hypermedia level (how users are allowed to act upon an application in terms of documents, hypermedia links and processes) and the transport level (how process entities within the system are to be authenticated).
- *Brokering*. One of the most powerful attributes of the Microcosm architecture has been its ability to integrate with almost any third party application on the Microsoft Windows desktop, albeit at varying levels, as demonstrated by Davis [7]. These techniques must now be advanced to allow Microcosm TNG to integrate effectively within a distributed environment. To satisfy this need, we are currently experimenting with emerging technologies such as Common Object Request Broker Architecture (CORBA) [18] and ANSAware, which display great promise for improved distributed and heterogeneous application inter-operability.
- *Resource discovery*. In a widely-distributed, shared, information environment, there can be an acute problem of locating appropriate information resources. This is illustrated by the large number of search facilities that have been developed for the WWW, which has no integrated resource discovery facilities. We intend to further the work on agent technology undertaken by Wilkins [24], which will investigate the provision of a framework for distributed agents to allow the development of a range of resource discovery services.
- *Link maintenance*. With any hypermedia system where links are being generated within and between hypermedia objects, it is important to ensure that links do not become invalidated when the resource base of an application changes. This problem can be tackled from a number of angles. One method is to establish a dependency between the two hypermedia applications involved as a link is authored. Then, when any changes are made to either end of the dependency, the consistency of the links can be validated automatically. Another approach is again to employ agent technology; an agent could be instructed to prove the validity of all links within a given application. The agent would work in the background, testing each link at local and remote locations, to ensure that they are consistent and still exist in the context in which they were created.
- *Computer Supported Co-operative Working (CSCW)*. The HCM was designed to accommodate collaboration and as such broadcasts a range of notification events upon users beginning and terminating various activities. Much research has focused upon providing advanced support for group working and version control within information systems [16] and this will undoubtedly influence the future development of Microcosm TNG. Due to the process-oriented and modular nature of the framework, each of these research areas can be incorporated into the system as and when they are developed.

6. Conclusions

The success of the WWW in using hypermedia facilities to provide access to vast amounts of distributed information provides firm evidence for the viability of hypermedia navigation as a component of an information management system. However, we have shown how few existing hypermedia systems provide all the facilities necessary to act as the basis for such a system.

The distributed hypermedia model outlined in this paper is designed to act as a modular framework within which a flexible and extensible information system may be developed. It achieves this through its ability to be extended through several levels:

- The system is based on a peerless communication model. This allows a wide range of information resources to be integrated within a single, cohesive architecture without requiring the user to address each resource independently. Thus, otherwise stand-alone resources, such as on-line databases may be interfaced into the system to augment other forms of information access.
- By altering the component processes that are incorporated, a user is able to vary the hypermedia structure that they perceive. Thus, alternative views upon the same information may be supported according to a particular user's current requirements.
- By providing a complete underlying hypermedia link service, the system allows a variety of existing applications to be easily modified to access the available hypermedia functionality.
- The ability to package resources into logical groups which may then be published for other users to utilise allows scaleable access to information.
- The ability to utilise widely distributed components allows users to build a system which makes the best use of available resources. For instance, if a number components are required to run on particular hosts, the system is able to provide a unified view of these distributed resources.

The utility of this logical architecture is demonstrated by the Microcosm TNG prototype that has been developed. There are clearly a number of issues that must be addressed when attempting to provide a robust distributed system such as that envisaged, some of which are described in the Future Work section. We are currently extending the prototype system to address these issues.

References

[1] ANDREWS, K., KAPPE, F. and MAURER, H., Serving Information to the Web with Hyper-G. *In:* Computer Networks and ISDN Systems, Volume 27, Number 6, pages 919-926, 1995.

[2] BERNERS-LEE, T., CAILIAU, R., GROFF, J. and POLLERMANN, B., World-Wide Web: The Information Universe. *In*: Electronic Networking: Research, Applications and Policy, Vol. 2 No 1, Spring, Meckler Publishing, Westport, CT, USA, pages 52-58, 1992.

[3] BERNERS-LEE, T. and CONOLLY, D., Hypertext Mark-up Language Specification 2.0, 1995.

[4] BUCHANAN, M. C. and ZELLWEGGER, P.T., Specifying Temporal Behaviour in Hypermedia Documents. *In:* Lucarella, D., Nanard, J. and Paolini, P., *Eds*., The Proceedings of the ACM Conference on Hypertext, ECHT 92, Milano (November), ACM, pages 181-190, 1992.

[5] BURGER, A. M., MEYER, B. D., JUNG, C. P. and LONG, K. B., The Virtual Notebook System. *In:* Hypertext 91, Proceedings of Third ACM Conference on Hypertext, San Antonio, Texas, USA (December), ACM Press, pages 395-402, 1991.

[6] CARR, L. A., DeROURE, D. C., HALL W., and HILL, G. J., The Distributed Link Service: A Tool for Publishers, Authors and Readers. *In:* The World Wide Web Journal Issue One, Conference Proceedings of the Fourth International Word Wide Web Conference, Boston, USA, O'Reilly and Associates Inc., pages 647-656, 1995.

[7] DAVIS, H. C., KNIGHT, S. J. and HALL, W., Light Hypermedia Link Services: A Study of Third Party Application Integration. *In*: ECHT '94 Proceedings, Edinburgh, Scotland (September), ACM Press, pages 41-50, 1994.

[8] DAVIS, H. C., HALL, W., HEATH, I., HILL, G. J. and WILKINS, R. J., Towards an Integrated Information Environment with Open Hypermedia Systems. *In*: Lucarella, D., Nanard, J., Nanard, M. and Paolini, P., *Eds*., ECHT '92, Proceedings of the Fourth ACM Conference on Hypertext (November 30-December 4), Milan, Italy, ACM Press, pages 181-190, 1992.

[9] FOUNTAIN, A., HALL, W., HEATH, I. and DAVIS, H. C., Microcosm: An Open Model with Dynamic Linking. *In:* Rizk, A., Stroetz, N. and André, J., *Eds.*, Hypertext: Concepts, Systems and Applications, Proceedings of the European Conference on Hypertext, (INRIA, France, November), 298-311, 1990.

[10] GOOSE, S., The Design Of A Generic, Yet Customisable, Distributed Communication Framework, Multimedia Technical Report M95/4, Department of Electronics and Computer Science, University of Southampton, 1995.

[11] HARDMAN L., BULTERMAN, D. C. A. and VAN ROSSUM, G., The Amsterdam Hypermedia Model: Extending Hypertext to Support Real Multimedia. *In:* Hypermedia, 5(1), pages 47-69, 1993.

[12] HILL, G. J., WILKINS, R. J. and HALL, W., Open and Reconfigurable Hypermedia Systems: A Filter Base Model. *In:* Hypermedia, 5(2), pages 103-118, 1993.

[13] HILL, G. J. and HALL, W., Extending the Microcosm Model to a Distributed Environment. *In:* ECHT '94 Proceedings, Edinburgh (September 18-23), Scotland, ACM Press, 32-40, 1994.

[14] LITTLE, T. and GHAFOOR, A., Synchronisation and Storage Models for Multimedia Objects. *In:* IEEE Journal on Selected Areas of Communications, 8(3), pages 413-427, April 1990.

[15] MALCOM, K. C., POLTROCK, S. E. and SCHULER, D., Industrial Strength Hypermedia: Requirements for a Large Engineering Enterprise. *In*: Hypertext '91, Proceedings of Third ACM Conference on Hypertext, San Antonio, Texas, ACM Press, pages 13-25, 1991.

[16] MELLY, M., and HALL, W., Co-operative Work in Microcosm, Computer Science Technical Report, Department of Electronics and Computer Science, University of Southampton, 1995.

[17] NELSON, T., Literary Machines 87.1, published by the author, Mindful Press, 1987.

[18] Object Management Group, The Common Object Request Broker: Architecture and Specification, OMG Technical Document, Number 91-12-1, Revision 1.1, December 1991.

[19] SHIPMAN, F. M. III, CHANEY, R. J. and GORRY, G. A., Distributed Hypertext for Collaborative Research: The Virtual Notebook System. *In*: Hypertext '89 Proceedings, Pittsburgh, USA (November), pages 129-136, 1989.

[20] SMITH, K. E. and ZDONIK, S. B., Intermedia: A Case Study of the Differences Between Relational and Object-Oriented Database Systems. *In:* OOPSLA '87 Proceedings (October), pages 452-465, 1987.

[21] Sun Microsystems, Java Language Reference Manual, Sun Microsystems Inc.

[22] STEINMETZ, R., Synchronisation Properties in Multimedia Sytems. *In:* IEEE Journal on Selected Areas in Communications, 8(3), pages 401-412, April 1990.

[23] WIIL, U. K. and LEGGETT, J., Hyperform: Using Extensibility to Develop Dynamic, Open and Distributed Hypertext Systems. *In*: D. Lucarella, J. Nanard, M. Nanard and P. Paolini, *Eds*. ECHT '92, Proceedings of the Fourth ACM Conference on Hypertext, Milan, Italy (November), ACM Press, pages 251-261, 1992.

[24] WILKINS, R. J., DeROURE, D. C., HALL, W. and DAVIS, H. C., The Role of Agents in Multimedia Information Systems. *In:* Proceedings of the Intelligent Agents and the Next Information Revolution, Manchester, UK (May), pages 14-23, 1995.