HarAdmin: A Graphical Tool for Hyper–G Server Administration

Diplomarbeit der Technischen Mathematik, TU Graz

Claudia Windisch

Technische Universität Graz
Institut für Informationsverarbeitung und Computergestützte neue Medien (IICM)

Fertigstellung:	30. November 1995
Begutachter:	o.Univ.–Prof. Dr. phil. Dr. h.c. Hermann Maurer
Betreuer:	Dipl.Ing. Keith Andrews

Ich versichere, diese Arbeit selbstständig verfaßt, andere als diese angegebenen Quellen und Hilfsmittel nicht benutzt und mich auch sonst keiner unerlaubten Hilfsmittel bedient zu haben.

Die Diplomarbeit ist in englischer Sprache.

Danksagungen

Ich möchte mich beim Begutachter dieser Arbeit, o. Univ.–Prof. Dr.phil. Dr.h.c. Hermann Maurer, und bei meinem Betreuer, Dipl.Ing. Keith Andrews herzlich für Ihre Unterstützung bedanken.

Besonderer Dank gilt auch den anderen Mitarbeitern des Institutes, im besonderen Dipl.Ing. Bernhard Marschall, Dipl.Ing Gerald Pani, Dipl.Ing. Michael Pichler und Jürgen Schipflinger, die mir bei meiner Diplomarbeit stets mit Rat und Tat zur Seite standen.

Außerdem bedanke ich mich bei meiner Familie dafür, daß sie mir das Studium ermöglichte.

Abstract

This thesis gives a short overview of information systems available on the Internet and a detailed description of Hyper–G, the first second generation hypermedia information system. The biggest problem of distributed information systems is disorientation. Because of the huge amount of information available on the Internet the user has difficulty to gain an overview, to know how much information exists to a certain topic, etc. Hyper–G has navigational, structuring, and search facilities to help cope with this problem Two particular features are Hyper–G's user accounts and its hierarchical scheme of user groups, which are used to grant or deny access to specific parts of the information space.

Hgadmin, a VT100 terminal and HarAdmin, a graphical X–Windows based tool for user and group administration are explained in detail in this thesis. HarAdmin is based on the Hyper–G Client/Server Protocol and on hgadmin. It provides functionality for browsing through the user–group hierarchy, displays all existing users and groups of the current Hyper–G server, and makes it easy to create, delete, and manipulate user accounts and user groups.

Contents

List of Figures

Credits

Figures 3.2, 3.4, and 5.1 were created by Dipl.Ing Keith Andrews.

Figures 3.4 and 3.6 were created by Dipl.Ing Dr. Frank Kappe.

Chapter 1

Introduction

Since the beginning of the human race, people want to communicate to other people and to get information about certain topics. Many inventions were made to help people to do just that: for example writing on stone slabs, paper, telephone, and computers. The *Internet* [Kro94, The95], the network of networks, built twenty years ago was a great step forward. Nowadays computers can talk to millions of other computers over the Internet and can users can get almost any information they want.

Many different systems are available on the Internet[LPJ^{+}94], to help users locate information. *Archie* [ED92], developed at McGill University in Canada, is a dictionary server to help locate files residing on FTP servers. *WAIS* [Ste91] (Wide Area Information Server) supports a powerful search engine for full–text searching of large, previously indexed databases and uses *relevance feedback* to refine the search for certain documents. *Gopher* [MA95], developed at University of Minnesota, is the first information system that includes browsing and full–text searches. It structures its information as a hierarchy of menus. *WWW* [BLCGP92, Cai95] (World Wide Web, W3, or "the Web"), developed at CERN in Geneva, merges techniques of hypertext, information retrieval, and wide–area networking. Text documents can include hyperlinks, images, etc. Most WWW Servers have no integrated search facilities but uses WAIS as an add–on search engine. It is based on three basic concepts: URL (Uniform Resource Locator), HTTP (Hypertext Transfer Protocol), and HTML (Hypertext Markup Language).

Because of the huge amount of information available on the Internet, the user has despite the existing systems difficulty to gain an overview, to know is this all the user can see, is this information up–to–date.

Hyper–G, under development at Graz University of Technology, the first second generation hypermedia information system [AKMS95], is designed to cope with this problem of distributed hypermedia systems called disorientation or the "Lost in Hyperspace" syndrome. It provides search, orientational, and navigational facilities, structures its information in so called collections, and builds a hierarchical structure, called collection hierarchy. Hyper–G stores hyperlinks separatly from documents rather than embedded within them, and hence supports links in all kinds of documents such as MPEG–video clips [Mar95], images, audio, PostScript, and 3D-scenes.

Hyper–G uses the collection hierarchy to provide a structural overview of the information space for the user, and to grant or deny access rights to parts of the information structure. Hyper–G is not a "stand–alone" system, it supports as much interoperability as possible with existing systems like Gopher, WAIS, and WWW.

Hyper–G is one of the best Web servers[AKM95b] around and uses the Client/Server Model with clients and servers connected over the Internet using TCP/IP. The *Hyper–G Client/Server Protocol* [KP94] coordinates the communication between the Hyper–G client and the Hyper–G server. The Hyper–G server consists of three distinct servers: The Link Server, an object–oriented database of objects and relations between such objects, the Document Cache Server, for caching documents and protocol and format converter, and the Full–Text Server, for full–text retrieval, document clustering, and automatic link generation.

The *Harmony* Client of Hyper–G, a X–Windows based client, provides navigational facilities and information feedback about the location of information. The main part of Harmony is the *Session Manager*, which communicates with the Hyper–G server and coordinates all other activities. Harmony supports several viewers and every document type has its own viewer (text, image, audio, film, 3D scene, postscript). All viewers share a common user interface style – If you know one, you know all as well.

HarAdmin, the Harmony Administrator for Hyper–G Server Administration, is a graphical X–Windows based tool for maintaining user accounts and user groups. The basic functionality is based on hgadmin, a VT100 terminal based version and the Hyper–G Client/Server Protocol . This thesis provides a user manual for HarAdmin and discusses the Hyper–G Client/Server Protocol commands used by HarAdmin. The user manual describes in which way user or group objects can be inserted, manipulated or deleted from the Hyper–G database. HarAdmin provides functions to navigate through the user–group hierarchy, for example to see the parent groups or subgroups of a certain group, or the host accounts of a certain user. This thesis gives a detailed example of deleting a group, because deleting a group is more difficult than deleting multiple users. Hyper–G has a group hierarchy, and the preconditions for deleting a group are: all of the groups direct subgroups have to be deleted or unlinked and all direct members have to be deleted (or unlinked).

Finally, this thesis gives a short description of the Hyper–G Client/Server Protocol, the message used by HarAdmin, and the implementation of the three main C++ classes of HarAdmin.

Have fun using HarAdmin!

Chapter 2

Internet Information Systems

The Internet [Kro94, The95, LPJ$^+$94], the worldwide network of networks, is steady growing and supports several different information systems. The Internet was born about 20 years ago, to connect the ARPAnet (Advanced Research Project Agency) and various radio and satellite networks together. The purpose of the ARPAnet was to build a network that could withstand unforeseen breakdowns of certain nodes during a nuclear attack. The existing communication was only between a source and a destination computer, but the idea was to build a network, in which every computer could talk with any computer on the whole network. The ARPAnet was extended with services like remote login (telnet), file transfer (ftp), and electronic mail (email). In 1983 the ARPAnet switched to TCP/IP to connect diverse networks together, and the name Internet was born.

TCP/IP – Transmission Control Protocol/Internet Protocol

The Internet is a packet switched network. The Internet Protocol IP defines the addressing and routing mechanism, puts the message data in envelopes also called packets with source and destination address. Internet addresses are unique 32–bit numbers, in four 8–bit parts, for example: 129.27.153.18. Names are easier to remember and gratefully the Internet uses a Domain Name System (DNS), which uses a name server to translate domain names into Internet addresses, eg: the domain name fiicmal01.tu–graz.ac.at stands for the Internet address 129.27.153.18. If the packets are too large, the Transmission Control Protocol TCP takes the packets, breaks them into pieces, and numbers them. The numbers are useful to control the order of the received packets. If one is missing or corrupted, it is re–transmitted. The TCP data is placed in a TCP envelope which in turn is placed inside a IP envelope. The IP packets are then routed through the Internet until they reached their destination. At the destination the IP and TCP envelopes are unpacked and the original data is reconstructed and checked eg. are packets out of order, corrupted, or got lost etc.

2.1 Basic Internet Applications

Since the beginning of the Internet in 1983 four basic Internet applications exist:

- Remote Login (telnet)
  Telnet is used for logging into other computers on the Internet. The user can access all services the remote machine provides.
  Telnet command: `telnet address-of-foreign-host`,
  eg. telnet iicm.tu–graz.ac.at.

- File Transfer Protocol (ftp)
  The ftp is used to move files from one computer to another. There are two different ways to receive files from the foreign computer: *ascii* for text files and *binary* for program files. And there also exists two different access modes: *anonymous*, password can be the whole email address, and *identified*, account name and password.
  Ftp command: `ftp address-of-foreign-ftpserver`,
  eg. ftp ftp.iicm.tu–graz.ac.at.

- Electronic Mail (email)
  Email is the most used Internet application, because it is easy to use and much more faster than the traditional surface mail. The mail is sent from one computer to the next until it reaches its destination. Today many mail programs exist, the most popular being SMTP (Simple Mail Transfer Protocol). Mail has some weaknesses like security, which is on a low level and foreign people can snoop around in your own private mail.

- Network News (news)
  With news you can easily find or get an answer to any kind of question you have. There exist thousands of so called *newsgroups* to which you can talk to get a desired information. Newsgroups can be compared with discussion groups or bulletin boards. The newsgroups are hierarchically structured, with the broadest grouping first in the name, followed by an arbitrary number of subgroups. (separated by periods). The USENET is a set of voluntary rules for passing and maintaining newsgroups. There exists seven major news categories: comp (computer science), news (network news), rec (hobbies, recreational activities, arts), sci (scientific research), soc (social issues), talk (debating different topics), and misc (all other topics).

2.2 First Generation Information Systems

As mentioned at the beginning the Internet consists of different information systems but they all haven't enough functionality to provide the power to exploit the large information and communication resources available on the Internet. Each of the existing information systems has its own special features but also weaknesses.

2.2.1 Archie

On the Internet many ftp–servers exist and for users it is difficult to find the right one to get the information they are looking for. Archie [ED92], developed at McGill University in Canada, is probably the first and most famous dictionary server to help locating files by name. Nowadays over 15 archie–servers exist and they index over 1500 FTP–servers worldwide. The content of each archie–server is updated monthly.

2.2.2 WAIS

WAIS (Wide Area Information Server) [Ste91], developed in 1989 by Thinking Machines, Apple Computer, and Dow Jones supports a powerful search engine for full–text searching large, previously indexed databases. WAIS clients send queries to servers, display ranked results, and fetch and display desired documents. WAIS has no functionality for browsing (hyperlinks) or structuring information contents but it can be considered real search engine. An important feature of WAIS is *relevance feedback*: Parts of the search result can be taken as input for a further search and the search can be refined to find certain documents. Many Gopher and WWW servers can use WAIS to provide search functionality.

2.2.3 Gopher

Gopher was developed at the University of Minnesota in 1991 as a campus wide information system and was the first information system that includes browsing and full–text searches. The design philosophy of Gopher was to make it possible for departments and other groups at the University, to publish information on their own desktop systems and to hide the distributed nature of the server from the user using the system [AAL+92]

Gopher [MA95] structures information as a hierarchy of menus comparable with a file system which is easy to use but has many weaknesses. For the user browsing through the menus is like browsing through a big graphical tree. In reality the user browses through a graph, from one menu the user comes to the next sub–menu and so on, but sub–menus can reside in many different places and a loop can be built. The user has difficulty to come back to previous selected menus and after gets lost.

Most Gopher servers support no search facilities themselves but use WAIS as an add–on search engine. They also have no integrate hyperlinking facilities. It is based on the Client/Server Model, servers accept connections from clients and return either documents, collections of documents, references of objects on other servers, or perform searches of document collections.

2.2.4 WWW

The World–Wide–Web (WWW, W3 or “The Web”) [BLCGP92, BLCL$^+$94, DR94] was developed at CERN in Geneva in 1989 as an information system for the particle physics community. WWW was created by Tim Berners–Lee and Robert Cailliau as a hypertext information system. It is now a distributed hypermedia information retrieval system which provides access to a large universe of documents. Merging the techniques of hypertext (the user clicks with the mouse), information retrieval, and wide–area networking produces the WWW–model.

Hypertext is not a linear text (one page after another), it consists of pieces of text (called nodes or documents), which are connected by links, shown in Figure 2.1. Hypertext forms a information space of documents and links, following the links is like navigating through this information space. A link is displayed as a hot–spot in the document, by clicking on it the user comes to the destination of the link, which can be a part of a document or the document itself.

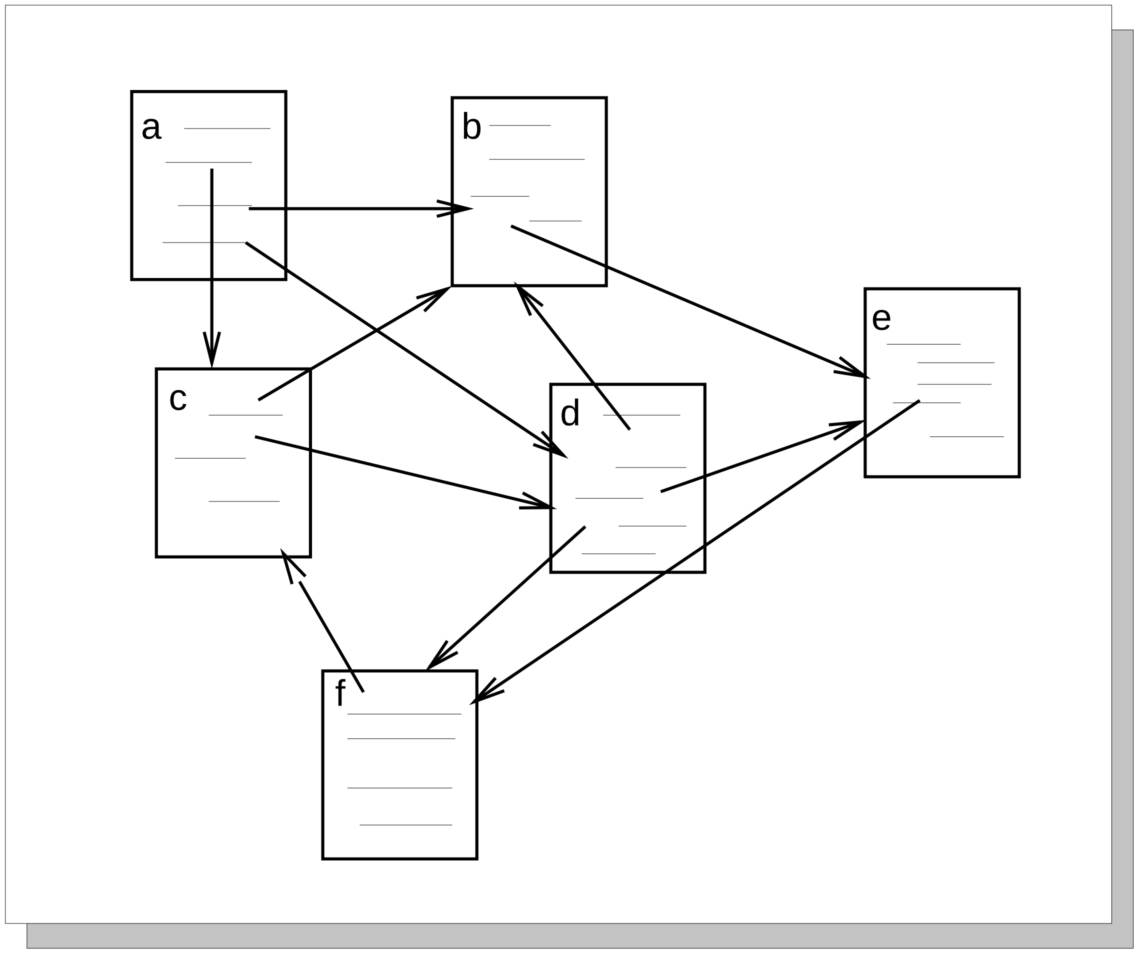


Figure 2.1: A Small Hypertext

Hypermedia [Mau92] is multimedia with hyperlinks. Documents may contain

text and images, audio, film, 3D–scenes, etc. There exists two different kinds of systems:
In *frame–based* systems, like ToolBook, each document must have the size of the computer's screen. Large documents must be spilt over many nodes and the author has full control over information content and presentation.
In *window–based* systems, like Hyper–G, documents can have any size, because a scrolling mechanism is used to scroll large documents. In such systems the user has no control over the presentation but specifies the information content.

The WWW consists of three basic concepts:

- **URL – Uniform Resource Locator**
  A URL is used to specify the location of a resource available electronically. They link WWW pages together and have a specific format, for example:

  HTTP: Hypertext Transfer Protocol with URL:

  `http://Host:port/selectorstring`

  Hyperg: Hyper–G protocol with URL:

  `hyperg://Host:port/selectorstring`

  ftp: File Transfer Protocol with URL:

  `ftp://host/filename(including path)`

  telnet: Remote login with URL:

  `telnet://user@host`

  News: Usenet Use with URL:

  `news://newsgroup/article`

  Gopher: Gopher protocol with URL

  `gopher://Host:port/selectorstring`

  WAIS: Wide Area Information Server with URL

  `wais://Host:port/selectorstring`

- **HTTP – Hypertext Transfer Protocol**
  HTTP is an ASCII protocol atop TCP/IP and implements a fast and flexible mechanism for following references between units of distributed information. The name is often misinterpreted, it doesn't only transfer hypertext, also other kind of information e.g. audio.

- **HTML – Hypertext Markup Language**
  HTML is the most important document format in WWW, it is an SGML (Structured Generalized Markup Language) encoded text format for WWW and is used to create hypertext documents with a logical structure. With HTML WWW embeds links and inline graphics in documents.

WWW structures its information in documents and links them together. Hierarchies are presented as lists of hyperlinks, and it is a distributed *hypermedia* system i.e. documents can comprise text, image, audio, and film. WWW uses HTML to embed links within text documents, but it supports no links from other kinds of documents e.g. links in film clips.

WWW is a great step forward but except hyperlinks, it has no structural facilities and links are not bidirectional. The user can't find out which other documents point to a particular document. This is a limitation of WWW, because in using one–way links so called *dangling links* can occur when a document is deleted or moved. Like Gopher, WWW servers generally have no integrated search facilities but can use WAIS for searching. Altogether, WWW should be considered a *first generation* networked hypermedia system.

2.3 Second Generation Information Systems

Second generation [AKMS95] networked hypermedia information systems attempt to alleviate some of the previously mentioned weaknesses. The largest problem is arguably that of disorientation, also known as the "Lost in Hyperspace" syndrome. When the user browses through the information space many questions arise:

- Where am I?
  The user sees only one current document at a time. The document can contain hyperlinks and for users it is difficult to gain an overview. They don't know which links from other documents point to the current document (incoming links) and to which other documents the hyperlinks in the current document point to (outgoing links).

  In the existing systems like Gopher, WAIS, and WWW to look backwards or using a map is not possible. Hyper–G uses a local map to show the relation of incoming and outgoing links of the current document to other documents.

- How much information can I see about a certain topic
  It is difficult for a user to see how many documents and megabytes exist about a certain topic.

- How much of the information have I already seen or is left
  Without so called *footprints* or checkmarks the problem of seeing the same document again and again can occur.

- Is this information the newest one
  If users found information about a certain topic, they are not sure if that information is really the newest one.

Hyper–G is a *second generation hypermedia system* which helps solve these problems by using tightly coupled structuring, search and linking facilities, and sophisticated user interface metaphors.

Chapter 3

Hyper–G

Hyper–G is the first *second generation* hypermedia information system [AKMS95, Flo95]. It is designed as a general–purpose, large–scale, distributed, multi–user system and it is similar in scope to Xanadu [Nel87], Intermedia [Mey86], WAIS, Gopher and WWW. Hyper–G[AKM95a, DH95] is under development at Graz University of Technology. It is based on the Client/Server Model across the Internet, and provides as much interoperability as possible with similar existing tools and services.

Some primary design goals were to:

- Provide orientational and navigational aids (like local map and hyperlinks).
- Provide automatic structuring and maintenance.
- Reduce fragmentation across servers, the consistency of data, links, etc. is preserved, eg. no dangling links can occur when a document is deleted or moved.
- Support user identification and access control.
- Support multilinguality (German, English, Spanish, etc.).
- Maintain interoperability with existing systems like Gopher, WAIS, and WWW.

3.1 Special Facilities of Hyper–G

Hyper–G combines a number of navigational, structuring, and search facilities to help overcome the "Lost in Hyperspace" syndrome, as shown in Figure 3.1:

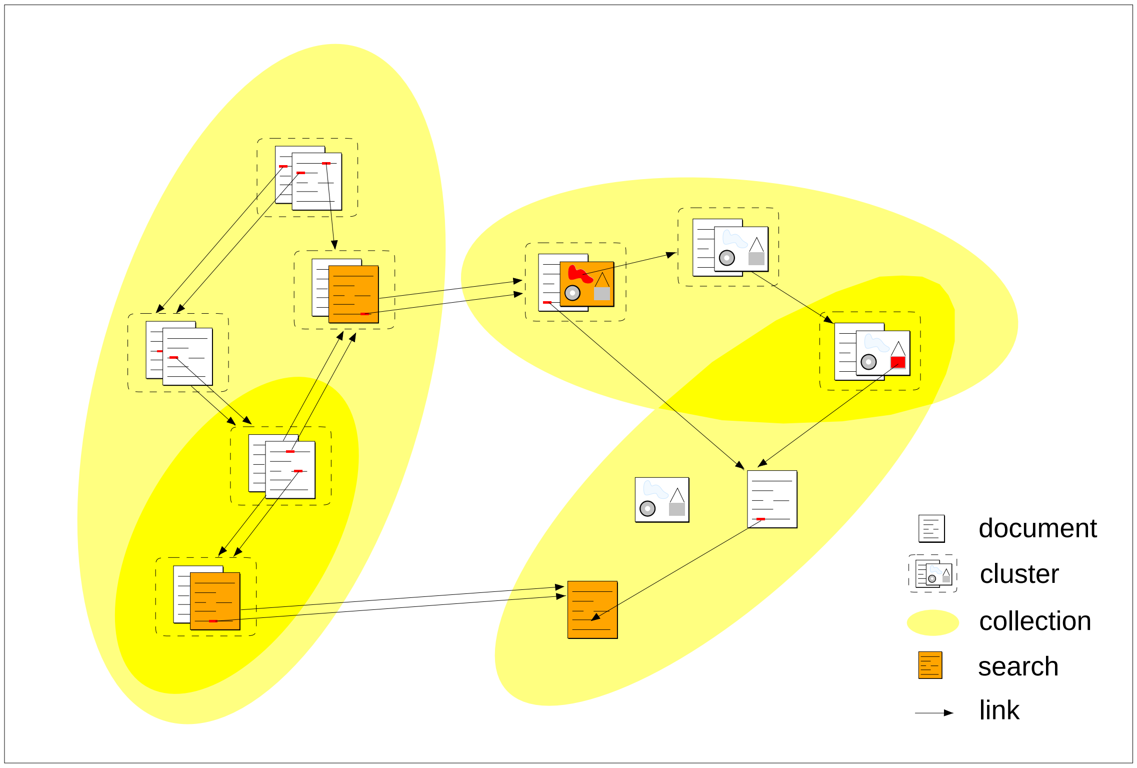


Figure 3.1: The Hyper–G Data Model

Hyperlinks

In Hyper–G the user navigates by clicking on a *hyperlink*, displayed as a hot spot, attached to a document. The destination of the link can either be a certain area in another document or the whole document itself, which in turn can contain hyperlinks. Not only text documents can contain links but also image, audio, and film clips [Mar95], 3D–scenes, and PostScript documents.

Structuring

Hyper–G structures its information in so–called *collections*. Collections may themselves belong to one or more collections to create the *collection hierarchy*, an example of which is shown in Figure 3.2.

The collection hierarchy is actually an acyclic directed graph, rather than a strict hierarchy, since objects may belong to multiple parent collections. The three main purposes of the collection hierarchy are:

- Navigation
  Whenever the user opens and closes collections and sub collections, starting at topmost root collection or activates a certain document no matter if selecting it by clicking or as a destination of a hyperlink the location is displayed in the collection hierarchy. The probability of becoming "lost in hyperspace"

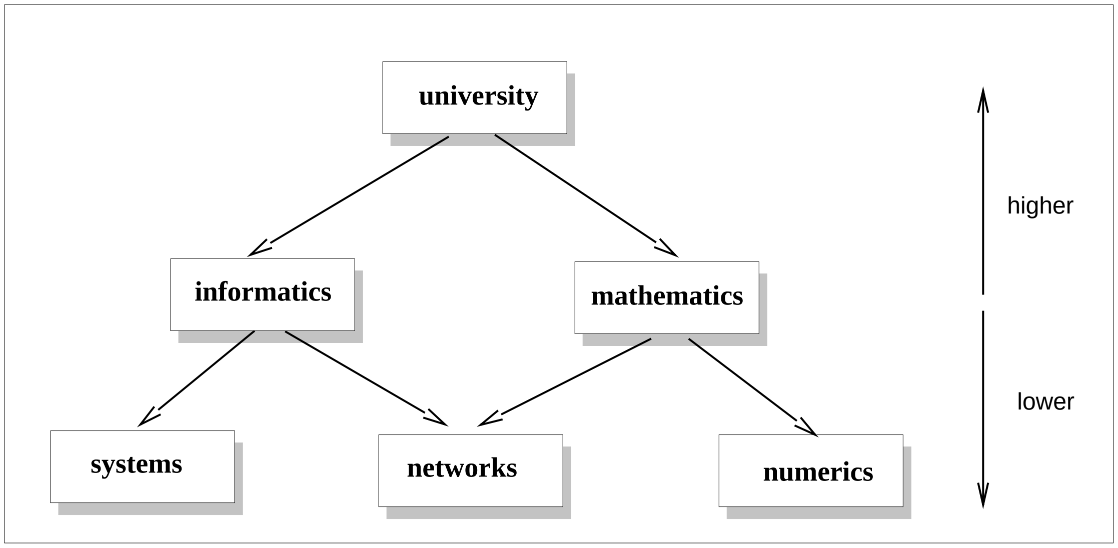


Figure 3.2: The Collection Hierarchy

is reduced, because the collection hierarchy can be seen as a global map for showing where you are in the Hyperspace.

- Search Scope
  The user has the facility to restrict the search to particular collections by marking them "active". The scope of the search may be as narrow as one collection on a single server or as wide as all collections on all Hyper–G servers worldwide.

- Access Rights
  The collection hierarchy is used to grant or deny

  – Write permission to certain parts of the information structures to authors

  – Read Access to certain users and user groups

Currently three classes of collections exist as shown in Figure 3.3:

collection: displays a list of menu items when visited. The list can be sorted by certain attributes (title, author, . . .).

cluster: When a cluster is visited all its members are visited too. In Hyper–G a cluster is used to implement multimedia documents, to support multilingual documents (German, English, . . .) and version control.

tour: is a collection that visits all its members in a certain order. A special form is a *guided tour*. Guided tours are paths through the information system and they are useful for preparing hypermedia presentations of existing materials.

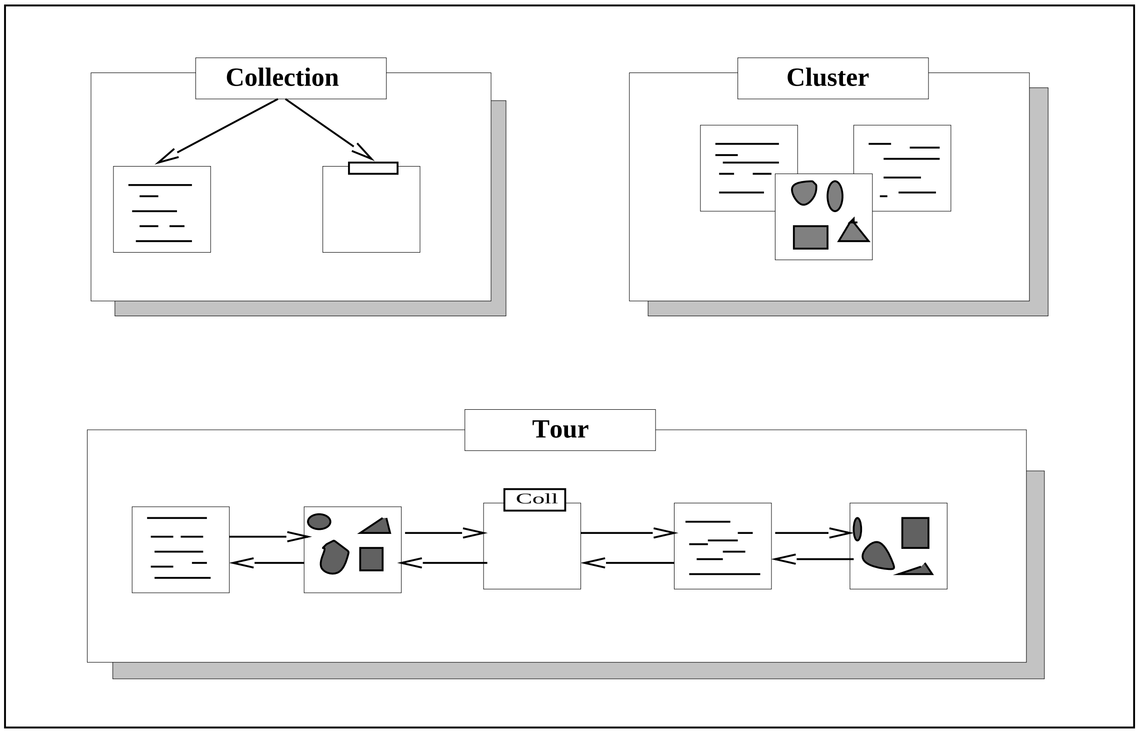


Figure 3.3: Three Types of Collection

Search Facilities

Hyper–G uses an object–oriented database, and every object is indexed automatically on creation. In Hyper–G two different modes of searching exist:

- Boolean queries
  Every Hyper–G object has a set of associated attributes like title, keywords, author, etc. which can be searched for. Typical queries might be "Give me all documents with *User accounts* in the title", or "Give me all documents with author *Claudia Windisch*".

- Full-text queries
  Text documents are automatically full text indexed on insertion. The full text server supports:
  – fuzzy boolean queries
  – WAIS-like-nearest-neighbour searches based on the vector space model.

The result of the search is a ranked list.

3.2 Interoperability

Hyper–G supports as much interoperability with other Internet information systems as is possible. It can interact with Gopher, WWW, WAIS, and FTP servers and

Gopher and WWW clients, as shown in Figure 3.4.

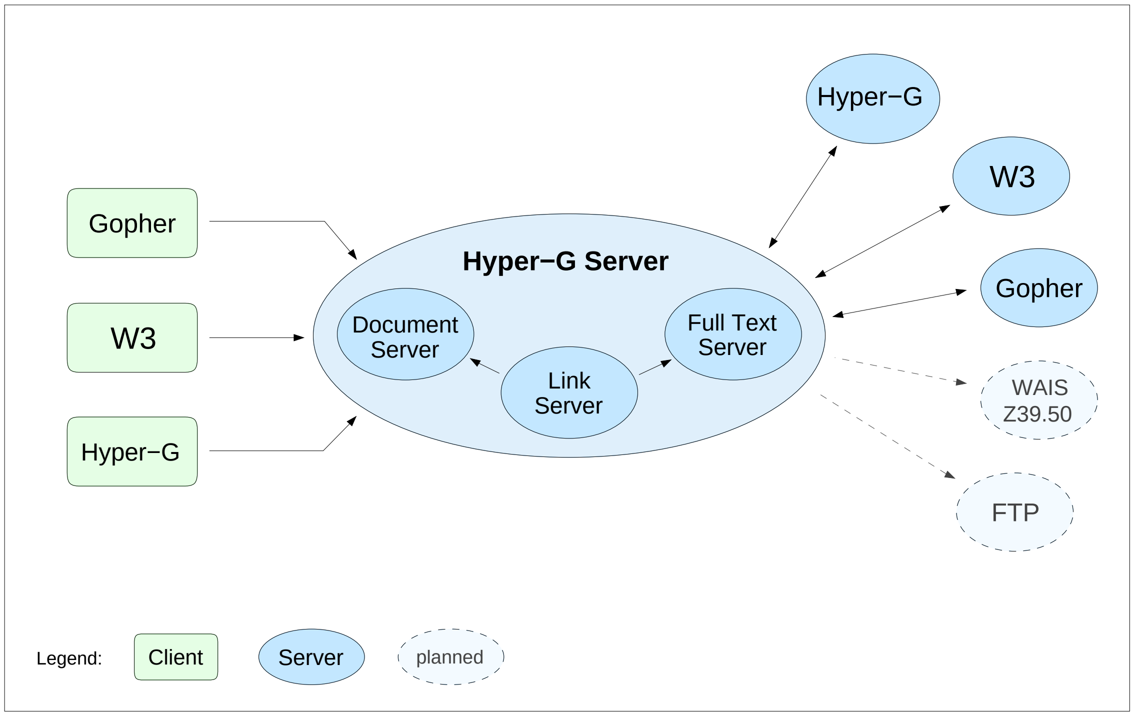


Figure 3.4: The Architecture of Hyper–G

On the left side of Figure 3.4, we see that:

- A Gopher client accessing the Hyper–G server: The Hyper–G server maps the collection hierarchy into a Gopher menu tree. Note that Gopher has no hyperlinks.
- A WWW client accessing the Hyper–G server: Each level of the collection hierarchy is converted to a HTML document containing a menu of links to the sub–menus. The menus are marked as searchable.

On the right side of Figure 3.4, we see that Hyper–G clients can connect to Gopher, WWW, and WAIS servers to retrieve information from them (stored on the Hyper–G server as pointers to the remote objects):

- A Gopher server being accessed by a Hyper–G client: Gopher menus are transformed into Hyper–G collections.
- A WWW server being accessed by a Hyper–G client: WWW text documents are transformed into Hyper–G text documents, including also hyperlinks.
- A WAIS server accessed by a Hyper–G client: WAIS queries and responses are transformed into Hyper–G queries and responses.

- A FTP server accessed by a Hyper–G client: FTP directories are transformed into Hyper–G collections.

Users and User Groups

Other features supported by Hyper–G and not found in comparable systems as Gopher and WWW include:

- User identification modes: from anonymous to fully identified.
- Support for a hierarchical structure of user groups.
- Access rights per user and/or user group for documents and collections.

3.3 The Hyper–G Server

The material in this section has been adapted from the paper "Hyper-G: A Distributed Hypermedia System" [Kap93] written by Frank Kappe.

The Hyper–G server[AKM95b] is a rather complex object–oriented database of *objects* i.e descriptions of documents, links, anchors, remote databases, etc. as well as *relations* between such objects, attributes (metadata), full text index, and users and groups. The Hyper–G server stores not only documents, but also anchors, collections, users, user groups, and server descriptions.

3.3.1 Architecture

Hyper–G uses the Client/Server Model, with clients and servers connected over Internet using TCP/IP. Unlike Gopher or WWW clients, Hyper–G clients usually talk to the same server for an entire session. If information from a remote server is needed, the local Hyper–G server fetches it and delivers it to the client. This approach has many advantages:

- The clients can be kept (very) simple, because the Hyper–G server can convert protocols and document formats for the client.
- A connection–oriented protocol can be used, there is no point in opening and closing the connection for every request.
- Remote information can be cached in the local server (see Document Cache Server).
- User accounts and access rights have to be maintained only in the local server, the user has to identify to one server only.
- Statistics and user profile information can be gathered on a per–session basis.

- Since every call passes through the local server, billing can be performed.

The Hyper–G server connects transparently to other servers on demand.

The architecture of Hyper–G shown in Figure 3.4 consists of three distinct servers:

- Link Server
- Document Cache Server
- Full Text Server

3.3.2 Link Server (hgserver)

As mentioned in the previous chapter, Hyper–G separates links from documents and stores them in a separate database, as was pioneered by Intermedia). One advantages of this is that the user can attach links such as annotations to a document, even when the document itself cannot be changed, for example if the user has no access rights or the document is stored on a CD-ROM.

Another feature of a centralized link store is that the system can support *bidirectional* links. The user has also the chance to go backwards. Such links are preconditions for automatic link maintenance (No *dangling links* can occur when a document is deleted or moved) and for a local map, which shows incoming and outgoing links of a certain document. Links also may be assigned attributes such as access permission, so links may be made visible only for a certain user or user groups. A big difference to other systems is that Hyper–G supports links in all kinds of documents, such as in MPEG video clips, images, 3D-scenes, PostScript documents, and audio.

The main functions of the link server are [Kap93]:

- It assigns unique object IDs to objects similar to an ISBN number, ensuring that no two objects share the same ID. When an object is modified it receives a new ID and the user can distinguish between the old and new version. If the object is deleted, the ID is never reused.
  Currently the *local* object ID is a 32–bit entity (there are 2^{32} object IDs per server). By concatenation of the local object ID and a 32–bit server ID we get a 64–bit *global* object ID, which is globally unique.
- Only the link server stores more information about an object, like author, title, keywords, etc.. If an object has to be modified, only the information stored in the link server has to be modified. And of course it receives a new object ID.
- It is aware of the collection hierarchy to maintain database consistency, e.g. making sure that every user is a member of at least one group.

- It performs complex boolean, and fuzzy boolean queries over the whole database or a subset of the collection hierarchy. Every document and every collection is indexed on creation.
- It contains a scheme of access rights to allow or disable access to certain documents or collections to certain users and user groups.

Clients connect to the link server to browse and search through the information space. The client sends a query to the link server, which returns a number of objects. The client lets the user choose of the list of hits and sends the object ID of the selected one to the document cache server to retrieve the whole document.

Links are represented in the database by means of a number of objects and relations between objects, there exists a number of different objects [KPS93] as shown in Figure 3.5.

Document Object
The document object holds information about a document, including document type, author, title and access information like host, port and protocol.

Anchor Object
It represents an anchor by a position attribute. The format depends on the document type of the anchor (text document or MPEG video clip, image, audio, PostScript,...). The anchor is used by clients to interpret the document.

Link Object
The link object stores information about the link like link type, author, access rights and it is used to search for certain links.

The corresponding relations:

Document–Anchor Relation
Specifies which anchor belongs to which document by means of their object IDs.

Link–Anchor Relation
Joins anchors (the source and the destination anchor) and a link object to form a link. The destination anchor can be a part of a document or a whole document. Note: No destination anchor object exists, rather the destination anchor field of the Link–Anchor Relation stores the destination document.

Example For Document–Anchor Relation

1. The client sends a request for document "1234" (1234 is the document ID) to the server. The client receives the document object "1234" and the anchor object "1235" attached to this document (look at the Document–Anchor Object in Figure 3.5).

2. The clients sends then the selector string containing the access information (Host, Port, Protocol, Path) of the document object “1234” to receive the whole document and afterwards displays the document object and with the help of the *Position* information of the anchor object the client also can highlight the attached anchor (here “1235”).

Example For Link–Anchor Relation

1. Having displayed the document with document ID “1234” and the attached link anchor “1235”, assume the user clicks now on the link anchor “1235”. The client sends the anchor ID “1235” to the Link Server to receive the corresponding destination document object.

2. The Link Server looks at the link–anchor relation for the source anchor “1235” and finds as destination anchor “2001”. (Also checks if the user has access rights, remember links can be assigned with attributes such as access permission, here “3000”).

3. Now the Link Server looks at the document–anchor relation to find the document with the source anchor “2001” and finds the destination document “2000” and sends the document object “2000” and the attached anchor objects to the client, and we are at step 2 of the example for Document–Anchor Relation and the circle is closed.

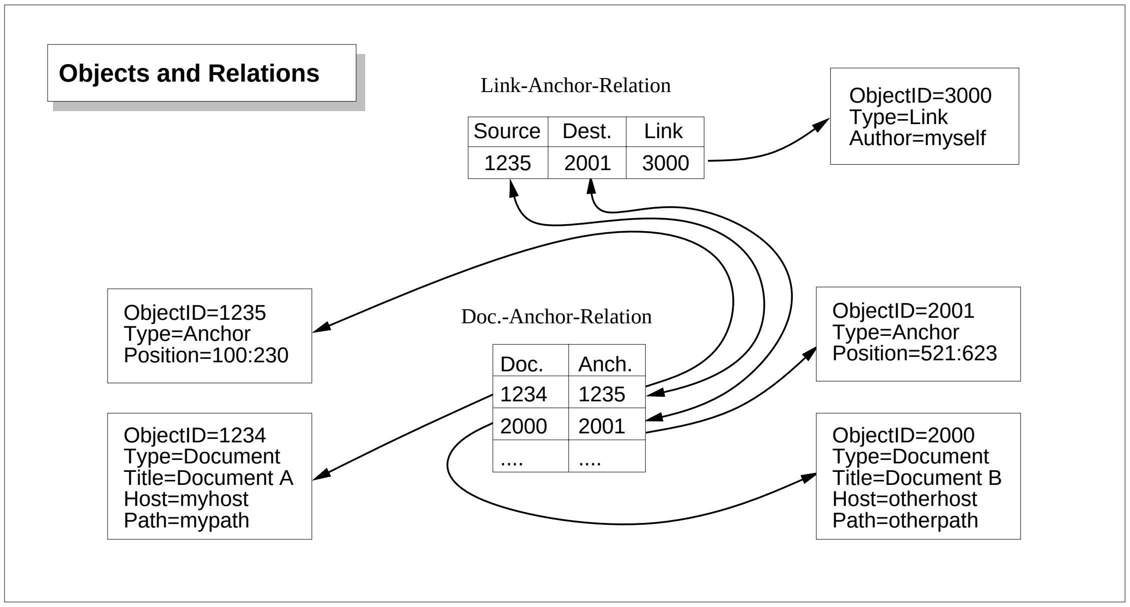


Figure 3.5: Some of the Link Server’s Internal Objects and Relations

Internally the link server comprises several components, as shown in Figure 3.6:

- a low–level database (*dbserver*)
  The dbserver knows about object and relations on a primitive level. It is used to perform fast, atomic functions e.g. locking an object and stores the link information of all documents.

- a full text server (*ftsever*)
  The *ftserver* is dedicated to full–text retrieval, document clustering and automatic link generation.

- a number of high–level database engines, one per connected client (*hgserver*)
  Hgserver talks to the client using the Hyper–G Client/Server Protocol. The hgserver knows about the user context and is a high level view of the database. For example, it knows what to do when an object is to be deleted. The important point is that the hgserver splits operations that might take a long time into a number of smaller and therefore faster transactions for the dbserver and ftserver.

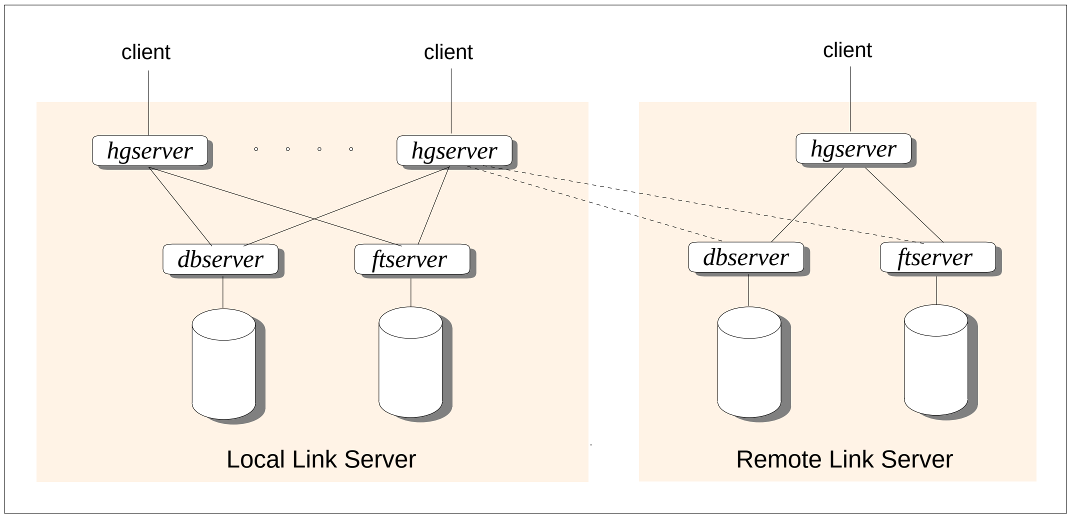


Figure 3.6: Internals of Link Server

Hgserver can also connect to dbserver and ftserver of remote servers and this feature makes it possible to perform a distributed search over a number of servers in parallel.

Searching the Hyper–G Database

Searches in the Hyper–G database are performed in three steps [Kap93]:

1. Find the ID of the object that matches the query. It can be done in O(log n) where n is the number of objects in the server. An array of object IDs is

returned. (If the content of a document is searched, the search is done by ftserver otherwise by dbserver).

2. This returned array is matched against a set of activated collections. The object IDs found are returned to the client (done by hgserver and dbserver).
3. The client must request the full object for the set of objects IDs found (done by hgserver alone).

3.3.3 Document Cache Server (dcserver)

Whenever a document is needed, the link server fetches it from the document cache server. The document cache server stores local documents and caches remote ones, therefore it speeds up the retrieval of documents by clients, at least for documents which are retrieved very often.

All document requests are routed through the local document cache server. When a client needs a document, the client sends the document cache object including ID, host, port, protocol, path, etc., to the document cache server. If a document is not in the cache, the document cache server fetches it from the remote document server, otherwise the document is send to the client directly. When space is exhausted, the server removes the *least recently accessed* document from the cache.

Hyper–G copes with a typical problem of caching in distributed environments: *Which version of a document is the newest?* When a document is modified, the new document gets, as described before, a new ID, which is passed to the client when the user visits that document. The old version is also stored in the document cache server, but it never can be accessed again and after a certain time it will be deleted anyway, as a result of the recently accessed cleansing strategy.

The document cache server can also be used as a *protocol* and *format converter*. If information is stored in other databases, like Gopher, WAIS, and WWW, clients may connect to the document cache server. The document cache server retrieves the information and caches it for the client. The client may request the document in a specific representation, the document cache server converts it for the client and caches it. This feature makes clients more simple and software maintenance easier, because only the cache's servers code has to be modified to support new protocols or file formats.

3.3.4 Full–Text Server (ftserver)

The full–text server is dedicated to:

- full text retrieval
- document clustering
- automatic link generation

It maintains an inverted index of all text documents for searching. Whenever a new document is inserted it is sent to the full text server for indexing it. When the document is removed its index is removed too. Searches can be full text queries and the full text server supports fuzzy boolean queries and WAIS–like–nearest–neighbour searches based on the vector space model. The result is a ranked list of objectIds.

Chapter 4

Server Administration Tools

Hyper–G maintains user accounts and user groups to deny or grant access rights to certain parts of the information structure. Users may be combined in user groups, which are hierarchically structured into groups and subgroups. The group hierarchy is an acyclic directed graph, since a group may belong to multiple parent groups. An example of a group hierarchy is shown in Figure 4.1.

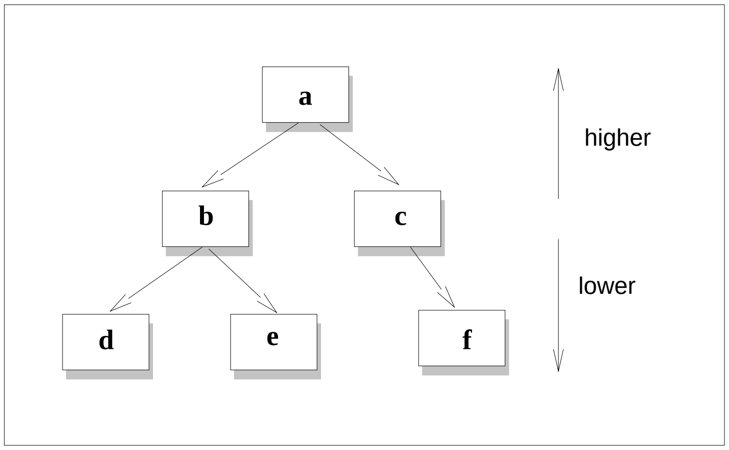


Figure 4.1: A Small Group Hierarchy

Within a group hierarchy, there exist direct and indirect parent groups, and direct and indirect subgroups.

Direct and Indirect Parent Groups

Direct parent groups are the nearest parents of a group. For example in Figure 4.1, group `f` has direct parent group group `c`. An indirect parent group is a group with an higher hierarchical level than a direct parent group. For example, group `f` has indirect parent group group `a`.

Direct and Indirect Subgroups

Direct subgroups are the nearest subgroups of a group. In Figure 4.1, group `a` has direct subgroups group `b` and group `c`. An indirect subgroup is a group with an lower hierarchical level than a direct subgroup. For example, group `a` has indirect subgroup group `d`, group `e`, and group `f`.

Hyper–G has two server administration tools

- hgadmin which is a VT100 terminal–based tool
- HarAdmin which is a graphical X–Windows based tool with the same look and feel as the Harmony client.

4.1 hgadmin

Hgadmin[1] [Mau96] is a VT100 terminal–based administration tool that is automatically installed with the Hyper–G server software. It has the same basic functionality as HarAdmin, but HarAdmin is a graphical X–Windows based administration tool and supports somewhat more functionality e.g. browsing through the user–group space and deleting multiple groups and users with one command.

Hgadmin has following options and parameters:

-h, -help Shows short help text and default-values.

-hghost hghost Hyper-G server host to connect to. Default is *hyperg* or **$HGHOST** from environment.

-hgport port Portnumber of Hyper-G server on *hghost*. Default is *418* or **$HGPORT** from environment.

-updhelp Update Hyper-G help entries.

4.1.1 Using hgadmin

Note that only the system administrator `hgsystem` and members of group `system` are allowed to insert, edit and delete user accounts and user groups. Other persons are only allowed to see all existing user accounts and user groups, and they can only change their own attributes (except adding themselves to a parent group).

After typing `hgadmin` in the commandline the user will see output similar to the following:

[1]The material in this section has been adapted from the manpages for "hgadmin" written by Gerald Pani.

```
>>>>>>>>>>>>>>> Hyper-G Administration-Tool <<<<<<<<<<<<<<<
>>          User: cwind(system)
>>
>>
>> 1 New User
>> 2 New Group
>> 3 Edit User
>> 4 Edit Group
>> 5 Identify
>> 6 Update Help
>>
>> 7 [q]uit
>>
>> Enter your choice:
```

To choose a menu-entry the user has to type in the corresponding number or the beginning of the description (say "5"', "I", "Id", ... for Identify).

In the first line of the Main Screen the identified user is displayed, if this user is a system administrator or a member of group `system` it is displayed like in the above Main Screen: eg. cwind(system).

Commands of the Main Screen

- **New User** Choose this to insert a new user.
- **New Group** Choose this to insert a new group.
- **Edit User** Choose this to edit a new user.
- **Edit Group** Choose this to edit a new group.
- **Identify** Choose this to reidentify yourself.
- **Update Help** Choose this to update Hyper-G help entries.

4.1.2 Managing Users

To see all user accounts existing on the current Hyper–G host, choose the number "3" from the Main Screen, then type in "*" (or nothing), and press `Enter`. If you want only see user accounts with prefix "c*" type in "c*" and press `Enter`.

Inserting a New User

Choose number "1" from the Main Screen, enter the name of the new user, press `Enter` and this leads to:

```
>>>>>>>>>>>>>>> New User <<<<<<<<<<<<<<<
>>          UserName        [newUser]
>>
>> 1 Add Host
>> 2 Add to (Parent) Group
>> 3 Add Password
>> 4 Add Password (encrypted form)
>> 5 Add Description
>> 6 Add Name of home collection
>> 7 Rem Host
>> 8 Rem from (Parent) Group
>> 9 Rem Password
>> 10 Rem Password (encrypted form)
>> 11 Rem Description
>> 12 Rem Name of home collection
>> 13 Insert User-Object
>>
>> 14 [q]uit
```

Commands of the New User Screen

- **Add Host** Add a new `Host`-entry (enter '<username>@<hostname>'). This attribute is used for an automatic identification after invoking a client (Client sends the UNIX user name and corresponding encrypted password; the host address is known from the connection).
- **Add to (Parent) Group** Add the new user to a parent group to inherit access rights from the group.
- **Add Password** Add a new `Password`-entry for the user.
- **Add Password (encrypted)** Add a new `Password`-entry in encrypted form for the user (It must be an exact copy from the UNIX password file, if it's used for automatic identification.) The encrypted password is stored under UNIX in the directory `/etc/passwd`.

  Multiple passwords are allowed.
- **Add Description** Choose this command to add a short description of the user to the user object, for example, the whole name of the user.

- **Add Name of home collection** Specify the name of the user's home collection.
- **Rem Host/ Group/Password/Password (encrypted)** To remove an attribute of the user, the corresponding text of the attribute has to entered between square brackets.
- **Rem Name of home collection** Choose this to remove the user's home collection name.
- **Insert User-Object** If no error occurs the new user object is inserted in the Hyper–G database. Remember only the system administrator and members of the group `system` are allowed to insert new user objects.

Editing an Existing User

Choose number "3" from the Main Screen, enter the name of the user, and press `Enter`. If you do not type any username you get a list of all existing users. If you type 'c*', you get a list of all users with prefix 'c'. This may lead to:

```
>>>>>>>>>>>>>>> Select User <<<<<<<<<<<<<<<
>>          UserName        [c*]
>>
>> 1 cderler    [Christian Derler]
>> 2 cwind      [Claudia Windisch]
>>
>> 3 [q]uit
```

Now choose one of the displayed users. This may lead to:

```
>>>>>>>>>>>>>>> Edit User <<<<<<<<<<<<<<<
>>         Host    [cwind@129.27.153.12]   (fiicmsg01.tu-graz.ac.at)
>>         Host    [cwind@129.27.153.14]   (fiicmds06.tu-graz.ac.at)
>>         Host    [cwind@129.27.153.17]   (fiicmhp01.tu-graz.ac.at)
>>         Host    [cwind@129.27.153.18]   (fiicmal01.tu-graz.ac.at)
>>         Host    [cwind@129.27.153.21]   (fiicmpc21)
>>         Host    [cwind@129.27.153.25]   (fiicmss03)
>>         Host    [cwind@129.27.153.3]    (fiicmsg02)
>>         Host    [cwind@129.27.153.4]    (fiicmhp02.tu-graz.ac.at)
>>         Host    [cwind@129.27.153.5]    (fiicmss01.tu-graz.ac.at)
>>         Host    [cwind@129.27.153.7]    (fiicmds03.tu-graz.ac.at)
>>         Host    [cwind@129.27.153.8]    (fiicmss02.tu-graz.ac.at)
>>         Host    [cwind@129.27.191.40]   (fiicm1pc40)
>>         Password        [s4pRxms5VOuy6]
>>         Group   [iicm]
>>         Group   [system]
>>         Description     [Claudia Windisch]
>>         Home collection [~cwind]
>>         Additional groups:  ed95_part
>>
>> 1 Add Host
>> 2 Add to (Parent) Group
>> 3 Add Password
>> 4 Add Password (encrypted form)
>> 5 Add Description
>> 6 Add Name of home collection
>> 7 Rem Host
>> 8 Rem from (Parent) Group
>> 9 Rem Password
>> 10 Rem Password (encrypted form)
>> 11 Rem Description
>> 12 Rem Name of home collection
>> 13 Delete
>>
>> 14 [q]uit
>>
>> Enter your choice:
```

In this screen you see first all existing attributes of the user, hosts displayed as Internet address and as domain name. The listed groups are all direct parent groups the user is a member of and additional groups shows all indirect parent groups (recursively, iicm or system are subgroups of ed95_part).

Commands of the Edit User Screen

- **Add/Rem entries** Adding or removing an entry updates the database on each time.

Deleting an Existing User

Choose number "3" from the Main Screen, enter the name of the user, and press `Enter`. Select the number "13" from the Edit User Screen to delete the user. If no error occurs like object locked from other users, or permission denied the user is deleted from the Hyper–G database.

4.1.3 Managing Groups

To see all existing groups on the current Hyper–G host, choose number "4" from the Main Screen, then type "*" (or nothing), and press `Enter`. If you want see all groups with prefix "i" type 'i*" and press `Enter`.

Inserting a New Group

Choose number "2" from the Main Screen, enter the name of the new group, and press `Enter`, this leads to:

```
>>>>>>>>>>>>>>> New Group <<<<<<<<<<<<<<<
>>          GroupName        [newGroup]
>>
>> 1 Add to (Parent) Group
>> 2 Add Description
>> 3 Rem from (Parent) Group
>> 4 Rem Description
>> 5 Insert Group-Object
>>
>> 12 [q]uit
```

Commands of the New Group Screen

- **Add to (Parent) Group** Choose this command to add a parent group to the new group, to inherit the access rights. Remember groups can be members of multiple groups, which recursively leads to the collection hierarchy.
- **Add Description** Choose this to add a short description of the new group to the group object, eg: the meaning of the group.

  DON'T FORGET to use the command
- **Insert Group-Object** Try to insert the group-object into the Hyper-G database.

Editing an Existing Group

Choose number "4" from the Main Screen, enter the group name, and press `Enter`. If you don't type any groupname you get a list of all existing groups. If you type 'i*', you get a list of all groups with prefix 'i'. This may lead to a screen similar to the following:

```
>>>>>>>>>>>>>>> Select Group <<<<<<<<<<<<<<<
>>          GroupName        [i*]
>>
>> 1 idg   [IDG Communications News Service]
>> 2 iicm  [IICM/IHM Mitarbeiter]
>>
>> 3 [q]uit
```

Now choose one of the displayed groups. This may lead to:

```
>>>>>>>>>>>>>>> Edit Group <<<<<<<<<<<<<<<
>>         GroupName        [iicm]
>>         BaseGroup        [ed95_part]
>>         Description      [IICM/IHM Mitarbeiter]
>>
>> 1 Add to (Parent) Group
>> 2 Add Description
>> 3 Rem from (Parent) Group
>> 4 Rem Description
>> 5 Show User
>> 6 Delete
>>
>> 7 [q]uit
>>
>> Enter your choice:
```

In this screen you see first all existing attributes of the group. The listed base groups are all direct parent groups the group is a member of and additional groups shows all indirect parent groups (if exist).

Commands of the Edit Group Screen

Add/Rem entries Adding or removing an entry updates the database on each time.

Show User Shows all users within this group (recursively).

Deleting an Existing Group

Choose number "4" from the Main Screen, enter the name of the group, press `Enter`, and choose the number "6" (command Delete) from the Edit Group Screen to delete a group.

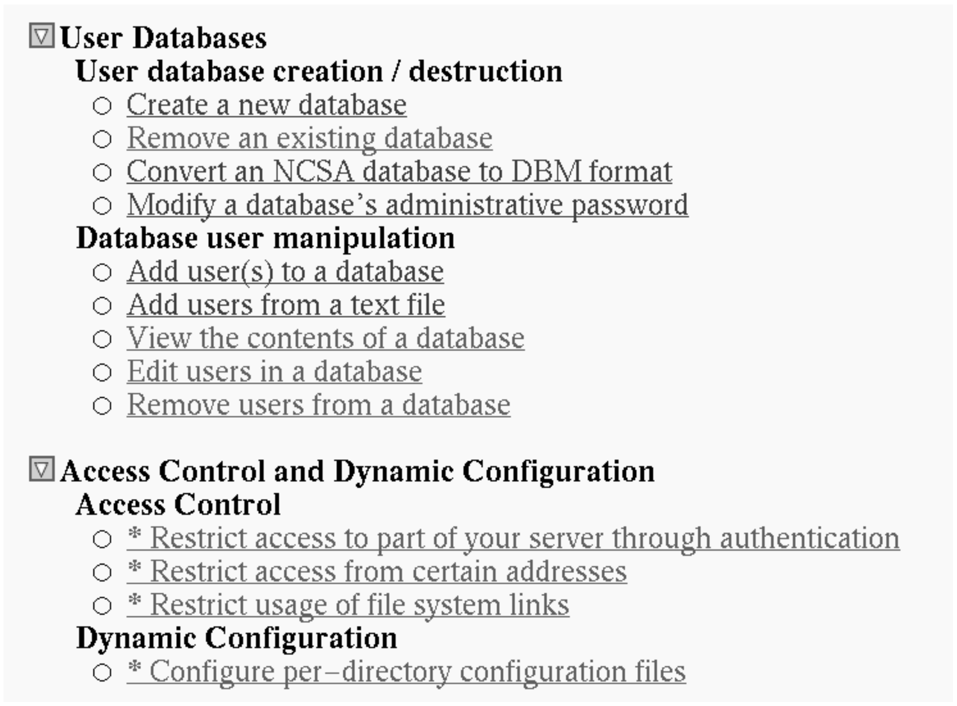
4.2 Netscape Commerce Server

In October 1995 Netscape announced their new Netscape Commerce Server, with special features: It can support flexible user authorization (HTTP 1.0 access authorization) and it can control access to individual files or directories using a username and a password, domain name, host name, IP address, or named group (Netscape user groups are different to Hyper–G groups, because Netscape can not support a hierarchical structure of user groups).

After installing a Netscape Commerce Server (NCS), you have to create a new user database, add users to the database, and then apply access control.

4.2.1 Managing Users

The NCS uses HTML forms to manage users and user groups. The main page of the Netscape Commerce Server administration tool is shown in Figure 4.2. Each underlined line points to forms, which in turn have many links to each other.



User Databases

User database creation / destruction
- Create a new database
- Remove an existing database
- Convert an NCSA database to DBM format
- Modify a database's administrative password

Database user manipulation
- Add user(s) to a database
- Add users from a text file
- View the contents of a database
- Edit users in a database
- Remove users from a database

Access Control and Dynamic Configuration

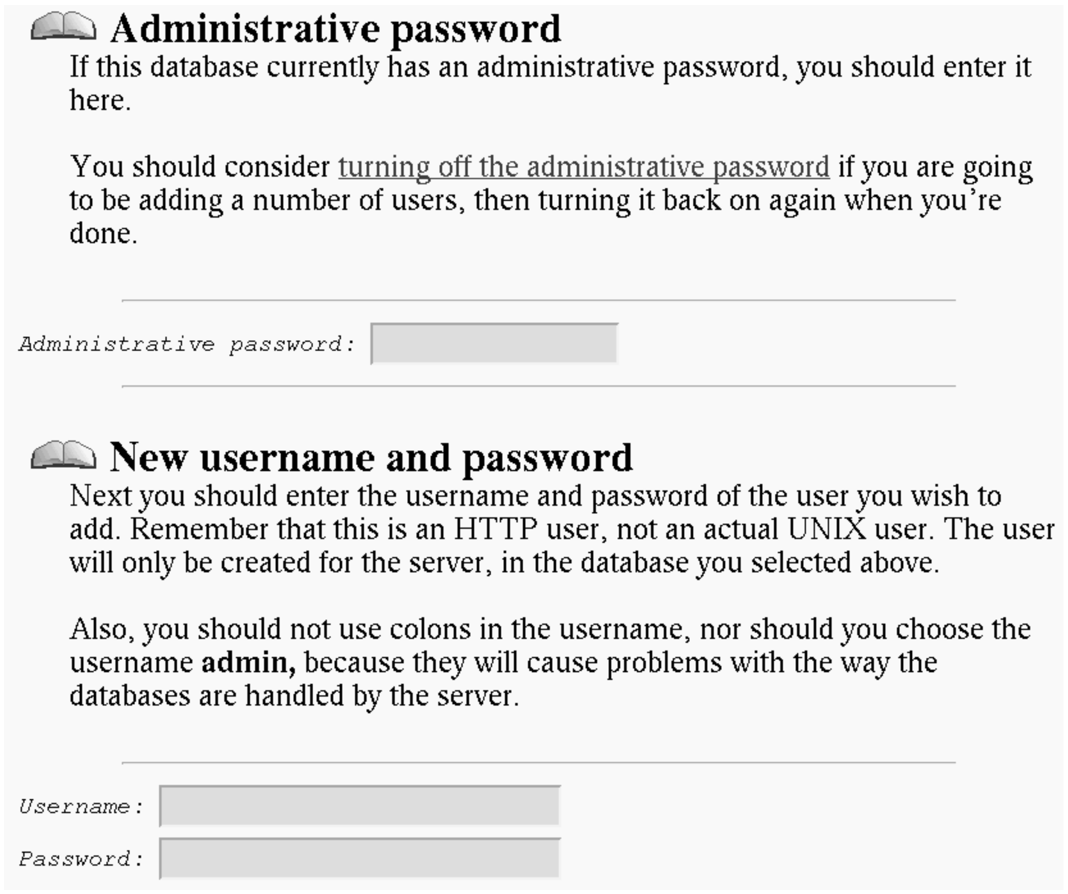
Access Control
- * Restrict access to part of your server through authentication
- * Restrict access from certain addresses
- * Restrict usage of file system links

Dynamic Configuration
- * Configure per–directory configuration files

Figure 4.2: The Main Page of The Netscape Commerce Server

Insert, Delete, Edit Users

The forms for adding, deleting, and editing shown in Figures 4.3 and 4.4 have a simple user interface. A user object consists of an username and one password, and there exists no group hierarchy like in Hyper–G. After having identified oneself as the system administrator you can insert, edit, and delete users.



Administrative password

If this database currently has an administrative password, you should enter it here.

You should consider turning off the administrative password if you are going to be adding a number of users, then turning it back on again when you're done.

Administrative password:

New username and password

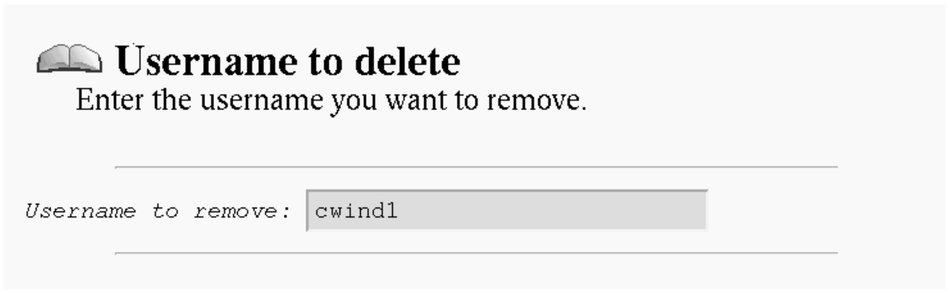
Next you should enter the username and password of the user you wish to add. Remember that this is an HTTP user, not an actual UNIX user. The user will only be created for the server, in the database you selected above.

Also, you should not use colons in the username, nor should you choose the username **admin,** because they will cause problems with the way the databases are handled by the server.

Username:

Password:

Figure 4.3: Adding a New User



Username to delete

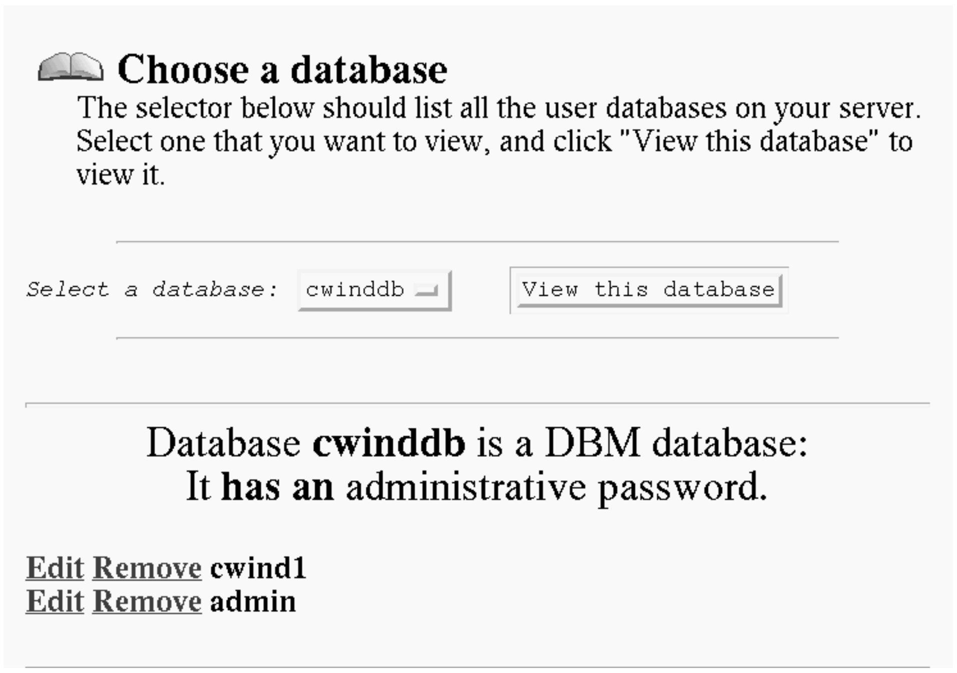
Enter the username you want to remove.

Username to remove: cwind1

Figure 4.4: Deleting Existing Users

Viewing the Content of the Database

A simple form, shown in Figure 4.5, which shows the content of the previously selected database. Only the usernames are displayed, nothing more. But you can easily switch to the froms for editing or removing an existing user.



Choose a database

The selector below should list all the user databases on your server. Select one that you want to view, and click "View this database" to view it.

Select a database: cwinddb View this database

Database **cwinddb** is a DBM database:
It **has an** administrative password.

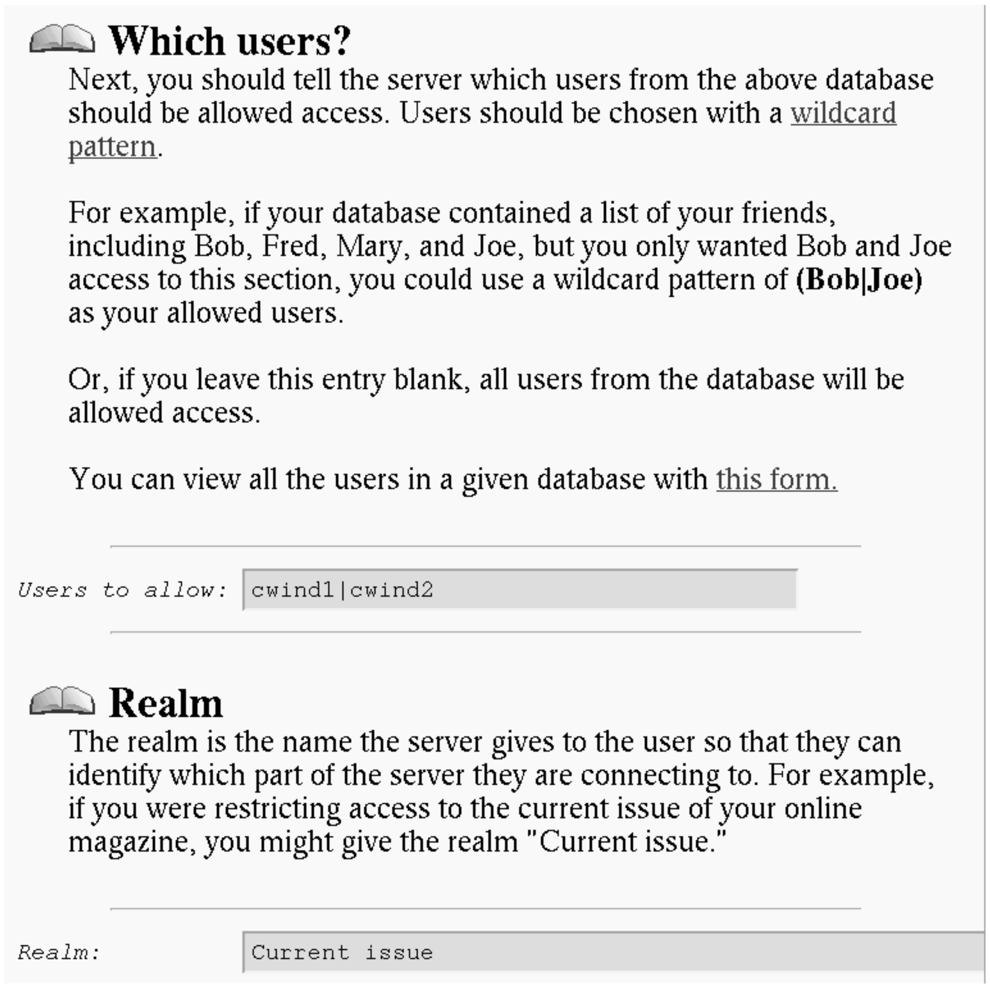
Edit Remove **cwind1**
Edit Remove **admin**

Figure 4.5: Viewing the Database

Apply Access Control

After added users to a database you have to apply access control. As said before, the Netscape Commerce Server supports user authorization and distinguishes between three different types of access, as shown in Figure 4.2

- Restrict access to parts of the server (Figure 4.6)
- Restrict access to certain addresses (Figure 4.7)
  You can restrict access to pages on your server by hostname and IP address
- Restrict usage of file system links (a file can be in two places without coping it) (Figure 4.8)
  Certain files are too important that every user can create links to such files. (such as system password files)



Which users?

Next, you should tell the server which users from the above database should be allowed access. Users should be chosen with a wildcard pattern.

For example, if your database contained a list of your friends, including Bob, Fred, Mary, and Joe, but you only wanted Bob and Joe access to this section, you could use a wildcard pattern of **(Bob|Joe)** as your allowed users.

Or, if you leave this entry blank, all users from the database will be allowed access.

You can view all the users in a given database with this form.

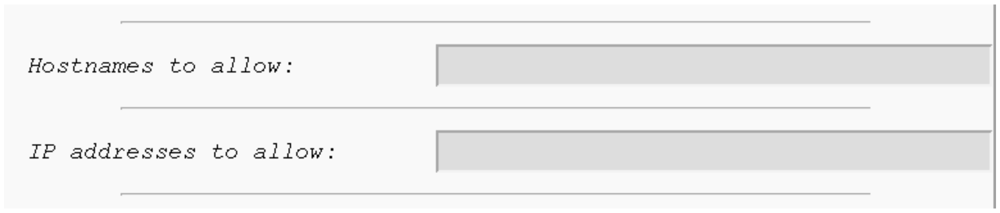
Users to allow: cwind1|cwind2

Realm

The realm is the name the server gives to the user so that they can identify which part of the server they are connecting to. For example, if you were restricting access to the current issue of your online magazine, you might give the realm "Current issue."

Realm: Current issue

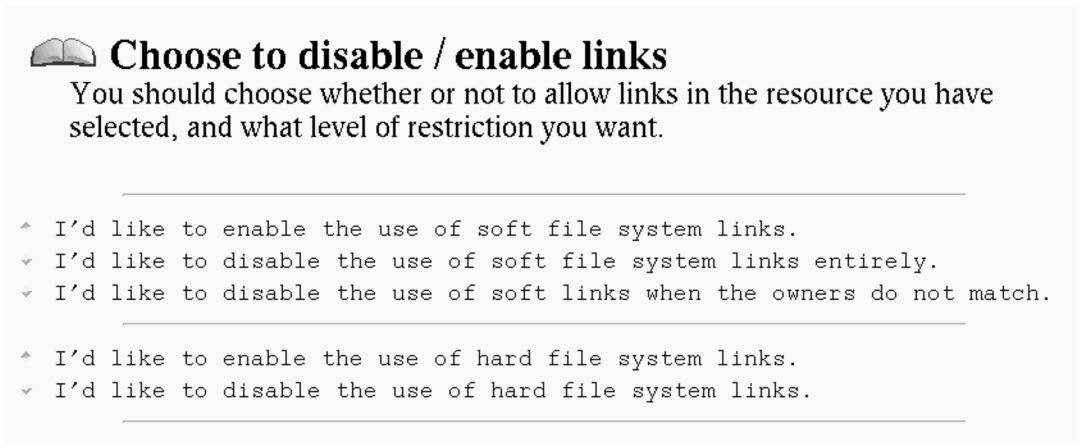
Figure 4.6: Page for Access Control 1 (Part of Server)



Hostnames to allow:

IP addresses to allow:

Figure 4.7: Page for Access Control 2 (Certain Addresses)



Choose to disable / enable links

You should choose whether or not to allow links in the resource you have selected, and what level of restriction you want.

- I'd like to enable the use of soft file system links.
- I'd like to disable the use of soft file system links entirely.
- I'd like to disable the use of soft links when the owners do not match.

- I'd like to enable the use of hard file system links.
- I'd like to disable the use of hard file system links.

Figure 4.8: Page for Access Control 3 (File System Links)

The Netscape Commerce Server distinguishs between two different database formats:

– DBM format, which is a high speed database format

– NCSA-style, which is a simple text format file including user names and passwords, each line is in the form of:

```
user1:password1
user2:password2
```

4.2.2 Changing a Text File to DBM Format

The big advantages to have a NCSA file is that the system administrator can use such an text file to add many users to an existing database and Netscape can convert it to a DBM format database (the passwords can be encrypted or not). Note that you must give the absolute path of the text file. The text file is stored in the server root in the subdirectory `userdb`
An example: `/usr3/ns-home/userdb/cwinddb2.pwf` is a NCSA–style text format file, including the users

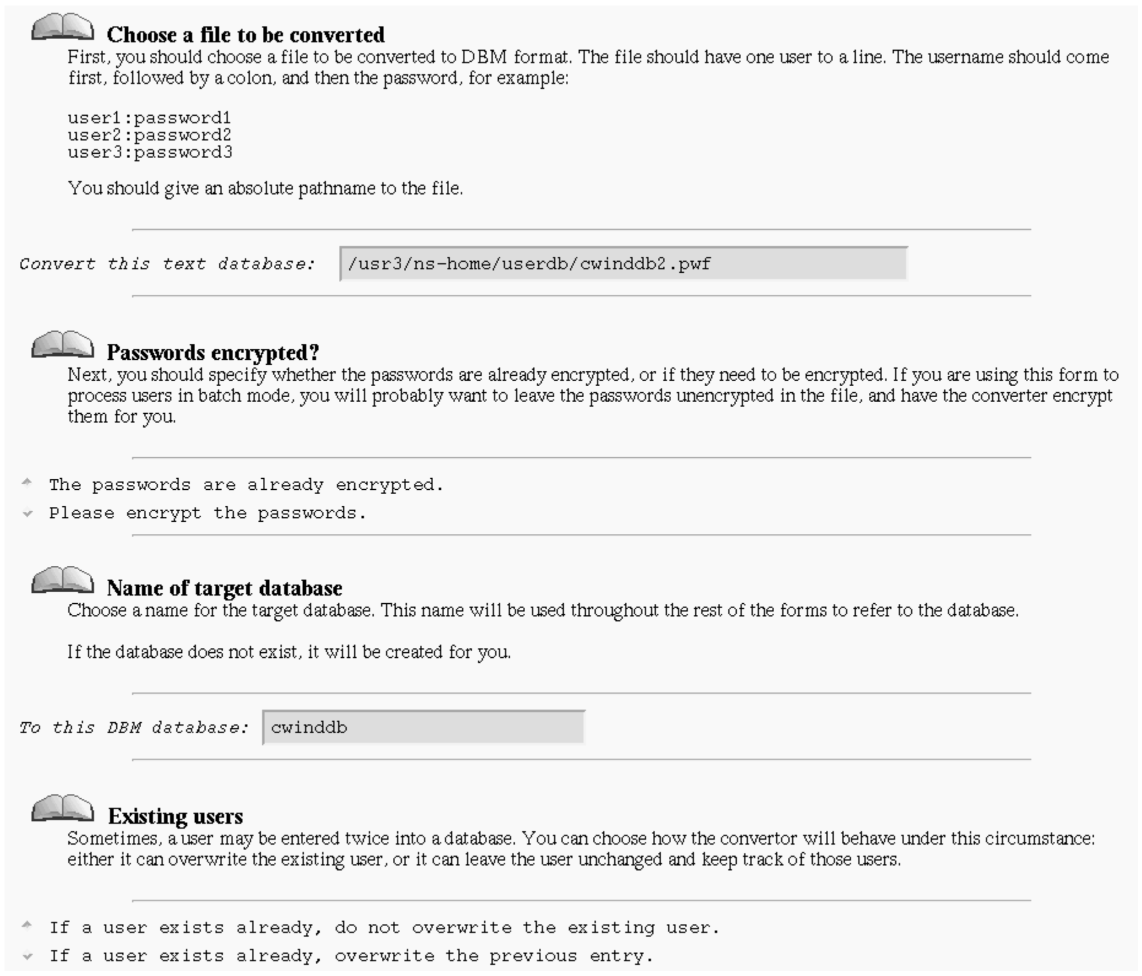
```
user1:password1
user2:password2
```

See Figure 4.9 for more detailed information.

The Big Differences to HarAdmin:

- It is a simple user interface

- It has no hierarchical structure of user groups
- Is supports no browsing through the user–group hierarchy
- User object consists of a username and one password



Choose a file to be converted
First, you should choose a file to be converted to DBM format. The file should have one user to a line. The username should come first, followed by a colon, and then the password, for example:

```
user1:password1
user2:password2
user3:password3
```

You should give an absolute pathname to the file.

Convert this text database: /usr3/ns-home/userdb/cwinddb2.pwf

Passwords encrypted?
Next, you should specify whether the passwords are already encrypted, or if they need to be encrypted. If you are using this form to process users in batch mode, you will probably want to leave the passwords unencrypted in the file, and have the converter encrypt them for you.

- The passwords are already encrypted.
- Please encrypt the passwords.

Name of target database
Choose a name for the target database. This name will be used throughout the rest of the forms to refer to the database.

If the database does not exist, it will be created for you.

To this DBM database: cwinddb

Existing users
Sometimes, a user may be entered twice into a database. You can choose how the convertor will behave under this circumstance: either it can overwrite the existing user, or it can leave the user unchanged and keep track of those users.

- If a user exists already, do not overwrite the existing user.
- If a user exists already, overwrite the previous entry.

Figure 4.9: Changing a Text File to DBM Format

For more information about the Netscape Commerce Server, see:
`http://home.netscape.com/comprod/netscape_commerce.html`

Chapter 5

The Harmony Client for Hyper–G

Harmony [AK94], is a graphical X–Windows based Hyper–G client, written in C++ [Str91] and using the InterViews[LVC89, LCV87] It takes advantage of Hyper–G's structuring and retrieval facilities to provide both intuitive navigational facilities and informative feedback about the location of information.

5.1 Architecture

Harmony is a multi–process Unix application, as illustrated in Figure 5.1.

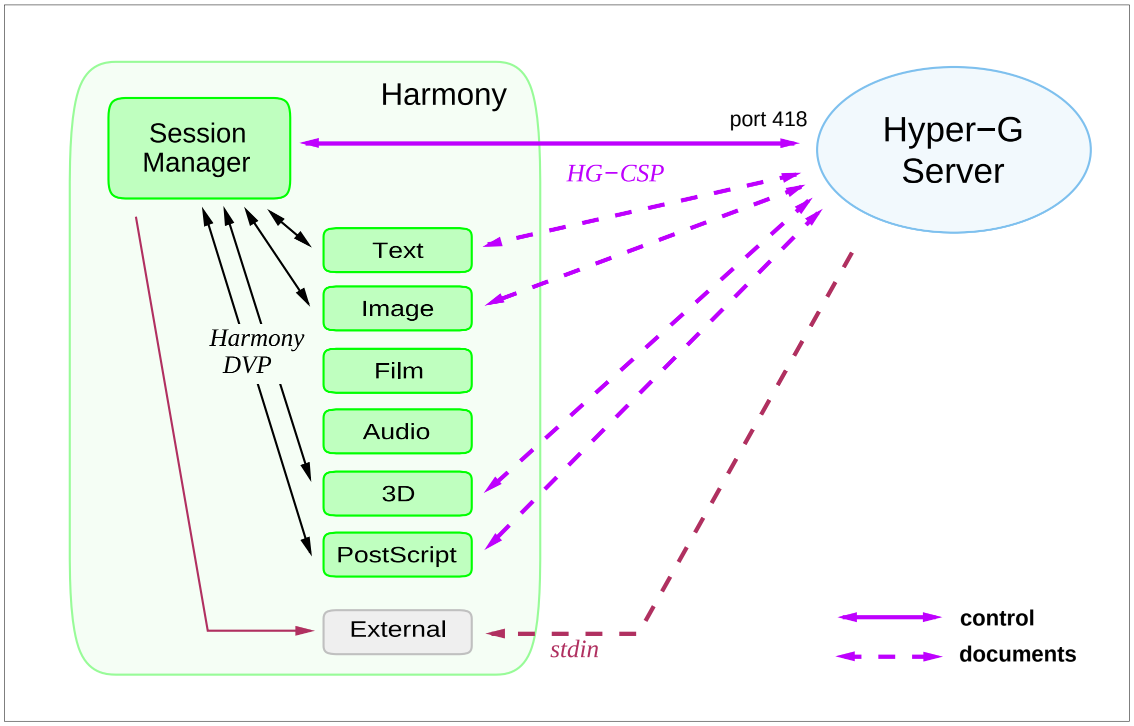


Figure 5.1: The Architecture of Harmony

The main part of Harmony is the *Session Manager*, which communicates with the Hyper–G server. The Session Manager provides navigational facilities and co-ordinates all other activities. As mentioned in the previous chapters, Hyper–G structures its documents in so–called collections, which in turn can be members of other collections. This recursive definition provides the *collection hierarchy*, which is an acyclic directed graph. The Session Manager displays this collection hierarchy in the *collection browser* and uses it to give the user an easy way to navigate through the information space, see Figure 5.2. Collections may be opened and closed and documents activated by double–clicking. Already visited documents or collections are marked with a tick, the number of documents belonging to a certain collection maybe displayed, etc.

The *History List* is a list of user's previously accessed collections. The user can easily go back to certain collections (or documents) by double–clicking them and the Session Manager shows the location of the selected collection (or document). This feature is called *location feedback*.

5.2 Viewers

Harmony supports several native document viewers, sharing a common user interface style:

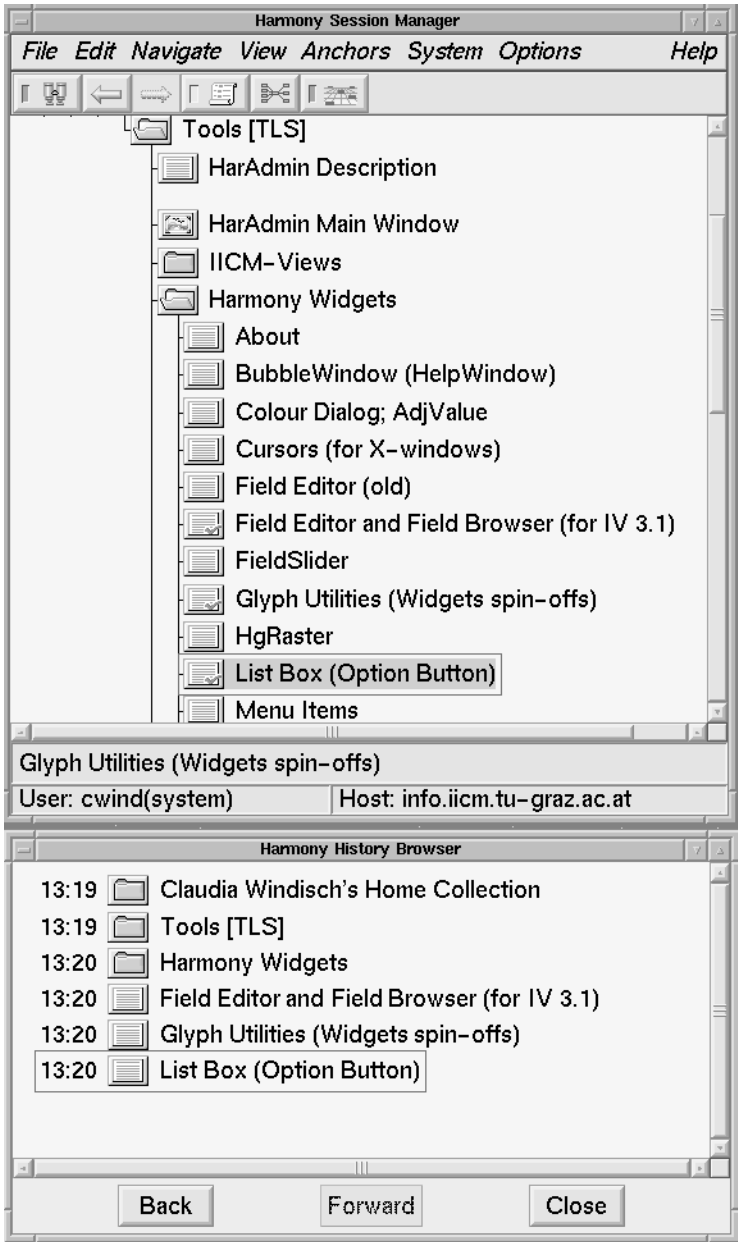
- Text Viewer
- Image Viewer
- MPEG Film Viewer
- Audio Clips
- 3D–Scene Viewer
- Postscript Files

The Harmony native viewers are illustrated in Figure 5.3.

The *Text Viewer* uses a generic SGML parser to display marked–up text documents like Hyper–G's HTF and WWW's HTML format. It supports scrolling, searching, selecting and inline images in TIFF, GIF, and JPEG format. Links are displayed as hot–spots and activated by double–clicking.

The *Image Viewer* accepts raster images of TIFF, GIF and JPEG format. Links can be assigned as rectangles, polygons, circular or elliptical areas.

The *Film Player* [Mar95] displays MPEG–1 videostreams. An important feature of Hyper–G are links in such video streams.



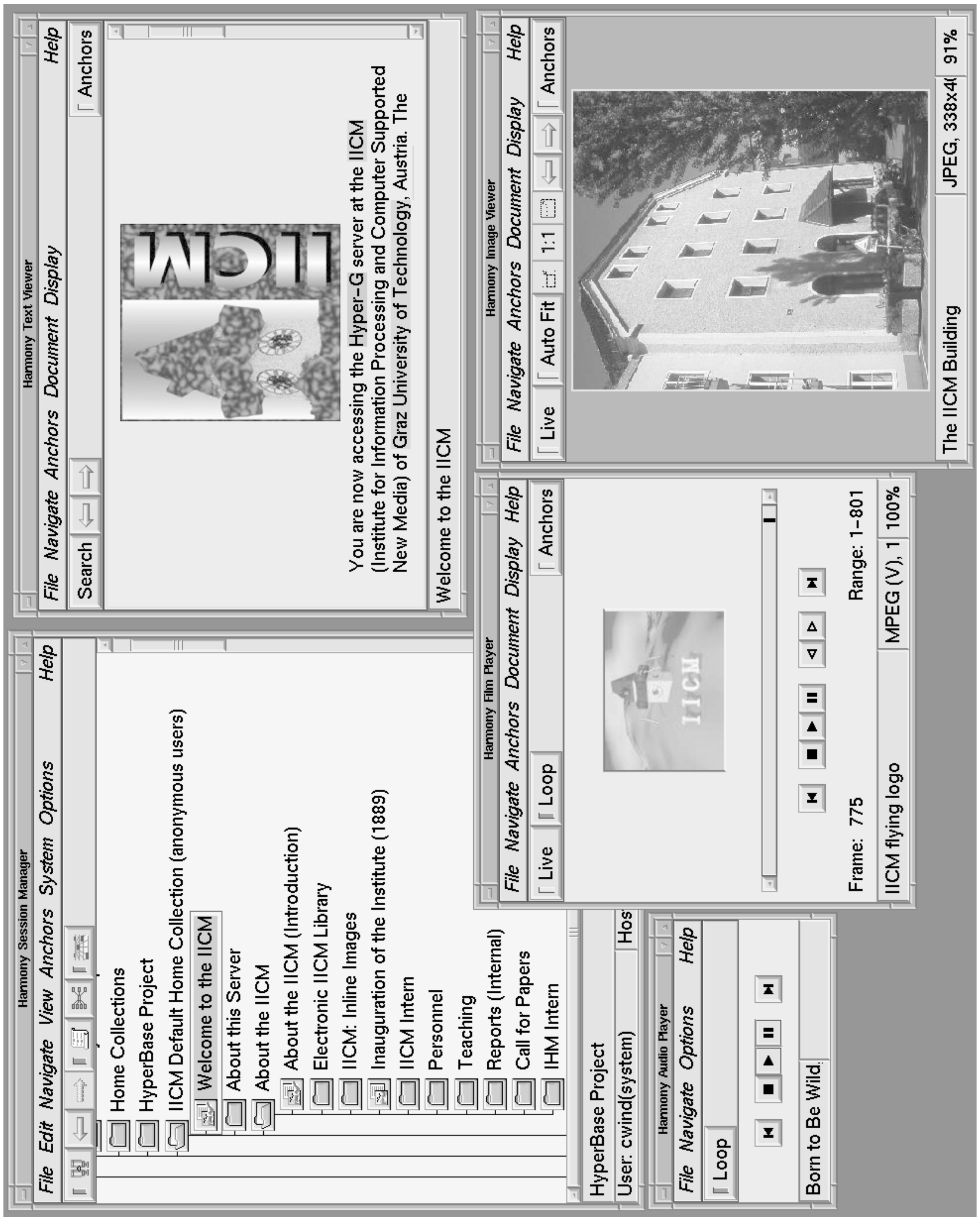
Tools [TLS]
HarAdmin Description
HarAdmin Main Window
IICM-Views
Harmony Widgets
About
BubbleWindow (HelpWindow)
Colour Dialog; AdjValue
Cursors (for X-windows)
Field Editor (old)
Field Editor and Field Browser (for IV 3.1)
FieldSlider
Glyph Utilities (Widgets spin-offs)
HgRaster
List Box (Option Button)
Menu Items

Glyph Utilities (Widgets spin-offs)

User: cwind(system) Host: info.iicm.tu-graz.ac.at

13:19 Claudia Windisch's Home Collection
13:19 Tools [TLS]
13:20 Harmony Widgets
13:20 Field Editor and Field Browser (for IV 3.1)
13:20 Glyph Utilities (Widgets spin-offs)
13:20 List Box (Option Button)

Figure 5.2: The Session Manager



You are now accessing the Hyper-G server at the IICM (Institute for Information Processing and Computer Supported New Media) of Graz University of Technology, Austria. The

Figure 5.3: Viewers of Harmony

The *Audio Player* can be configured to use either the Network Audio Server or local audio commands to play audio files in a variety of common formats.

Hyper–G also supports 3D–scene documents displayed with the Harmony VRweb *3D scene viewer*, see Figure 5.4. Users can view a model of a scene by moving themselves (walk, fly, fly to, headsup) and manipulate an object (translate, rotate, zoom). Links can be attached to objects within a scene or groups of polygons within an object.

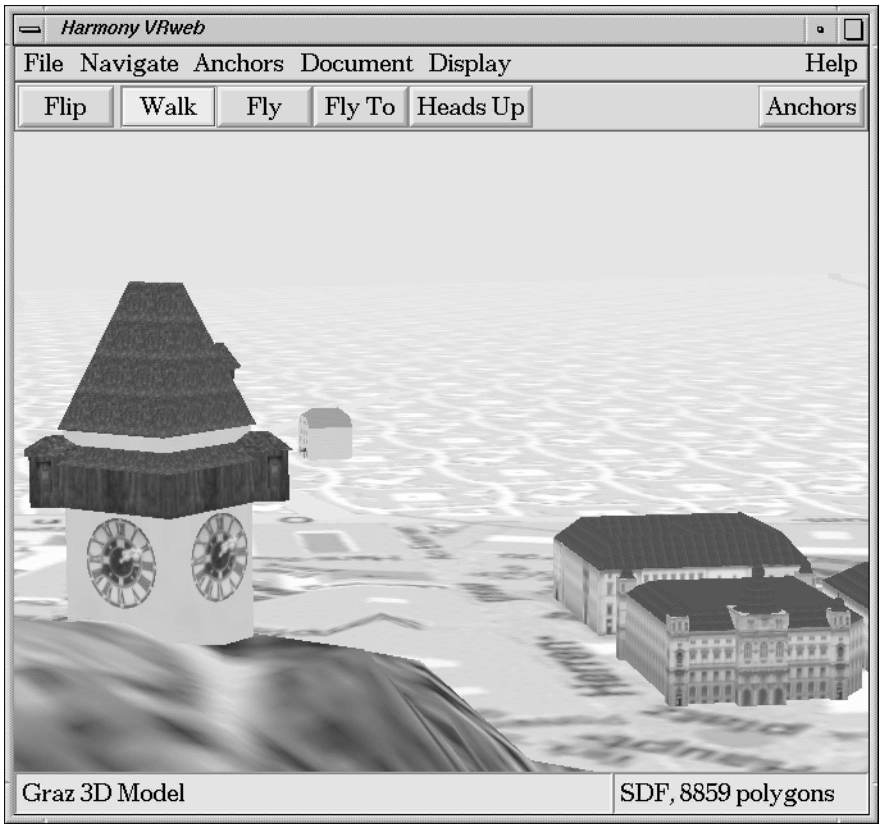


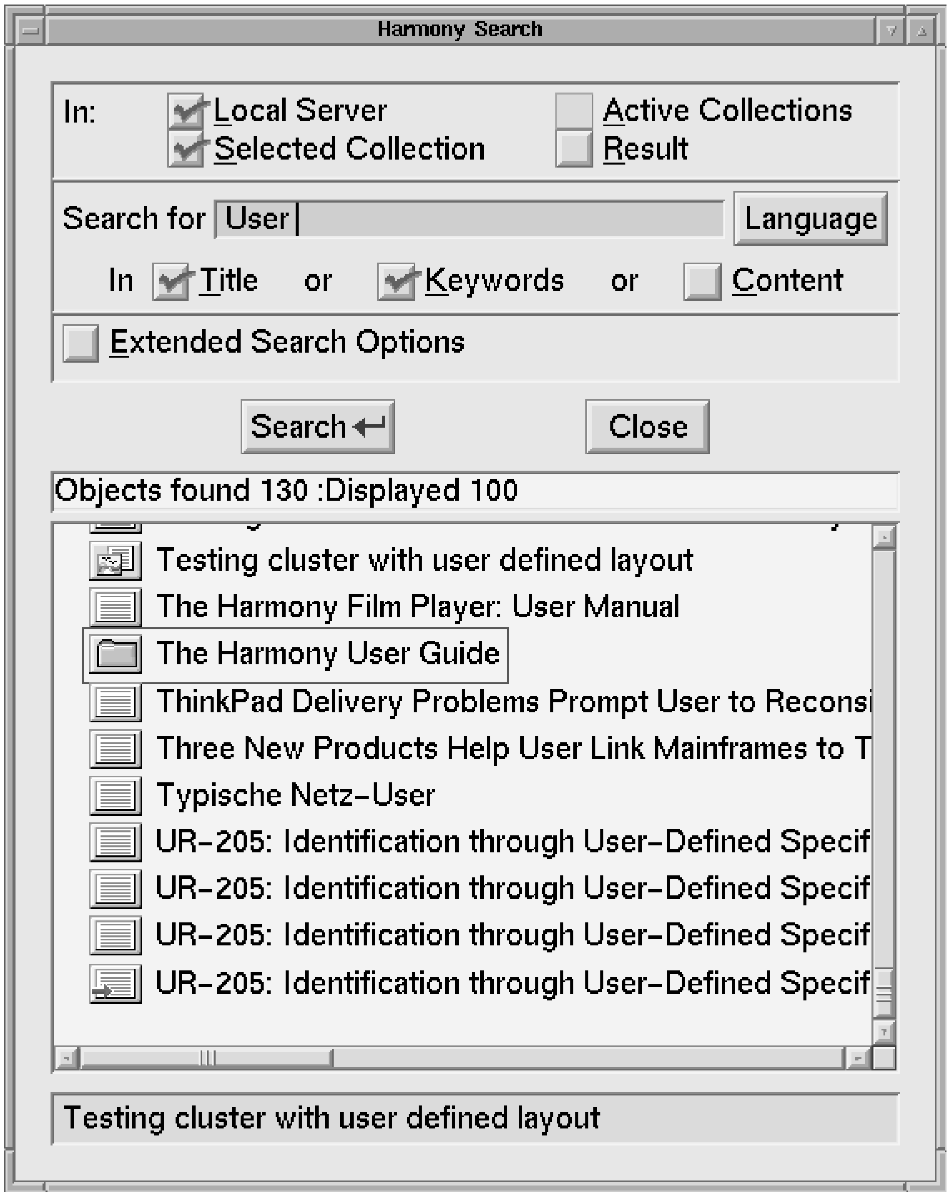
Figure 5.4: The 3D Scene Viewer

The *PostScript Viewer* displays documents in PostScript format and supports rectangular link anchors. The documents are uncompressed locally by the viewer.

Harmony supports title, keyword, and full text search. The user has the possibility to restrict the search to the local server, a particular collection or all Hyper–G servers in the world.

The result of the search is an ordered list of matching objects as shown in Figure 5.5. By single–clicking a certain object the Session Manager updates the collection hierarchy to display the path to the selected object.

The *local map* facility (Figure 5.6), similar to the local map of Intermedia [Mey86]. It provides a kind of short–range radar, generating on request a map of the vicinity of a document. The user has the possibility to double–click on an



Objects found 130 :Displayed 100

Testing cluster with user defined layout
The Harmony Film Player: User Manual
The Harmony User Guide
ThinkPad Delivery Problems Prompt User to Reconsi
Three New Products Help User Link Mainframes to T
Typische Netz–User
UR–205: Identification through User–Defined Specif
UR–205: Identification through User–Defined Specif
UR–205: Identification through User–Defined Specif
UR–205: Identification through User–Defined Specif

Testing cluster with user defined layout

Figure 5.5: Search Facilities of Harmony

icon to activate a certain document. The Session Manager shows the whole path, and the document is displayed in the corresponding viewer.

The *information landscape* [Eyl95] is a three–dimensional graphical overview map, as shown Figure 5.7. The collection hierarchy is visualized as three dimensional cubes on a plane. Users can fly over the landscape and activate cubes by double–clicking. Again the Session Manager is updated and the location of the collection or document is displayed. An overview window shows the local map from a bird's–eye view to help the user to orientate.

5.3 Inserting New Objects

To insert new objects in Harmony, the user has to select *Insert* from the *File* menu of the Session Manager. For example, if the user wants to insert a new document with an English title, special keywords and access rights, and afterwards the user adds a German title to the new inserted documents, the result might be similar to Figure 5.8.

Harmony supports *multiple languages* and changes the user interface (control elements ...) and displays the text in the selected language, an example is shown in Figure 5.9.

5.4 Linking Objects

As mentioned in the previous chapters, Hyper–G provides links between text documents, MPEG video clips, audio, images, PostScript documents, and 3D scenes. Harmony gives an easy way with the *Link Creator* to link them together. Figure 5.10 shows an example of linking a text document and an image document together. The text document contains the source anchor (the words "Main Window" are marked) and you have to define the destination anchor by selecting the destination document (or part of a document) in the collection browser (here "HarAdmin Main Window"), and after pressing the Create Link button in the Harmony Link Creator Window the link is created (displayed as a hot–spot in the source document "HarAdmin Description").

5.5 Communicating with Other Users

The *Status Browser* shown in Figure 5.11 displays a list of current logged in users, the local time, the up time, the performance index, and the name of the current Hyper–G server. You can select multiple users by holding the shift key and selecting with the left mouse button and talk (sending messages) to them. Only identified users have the possibility to send messages.

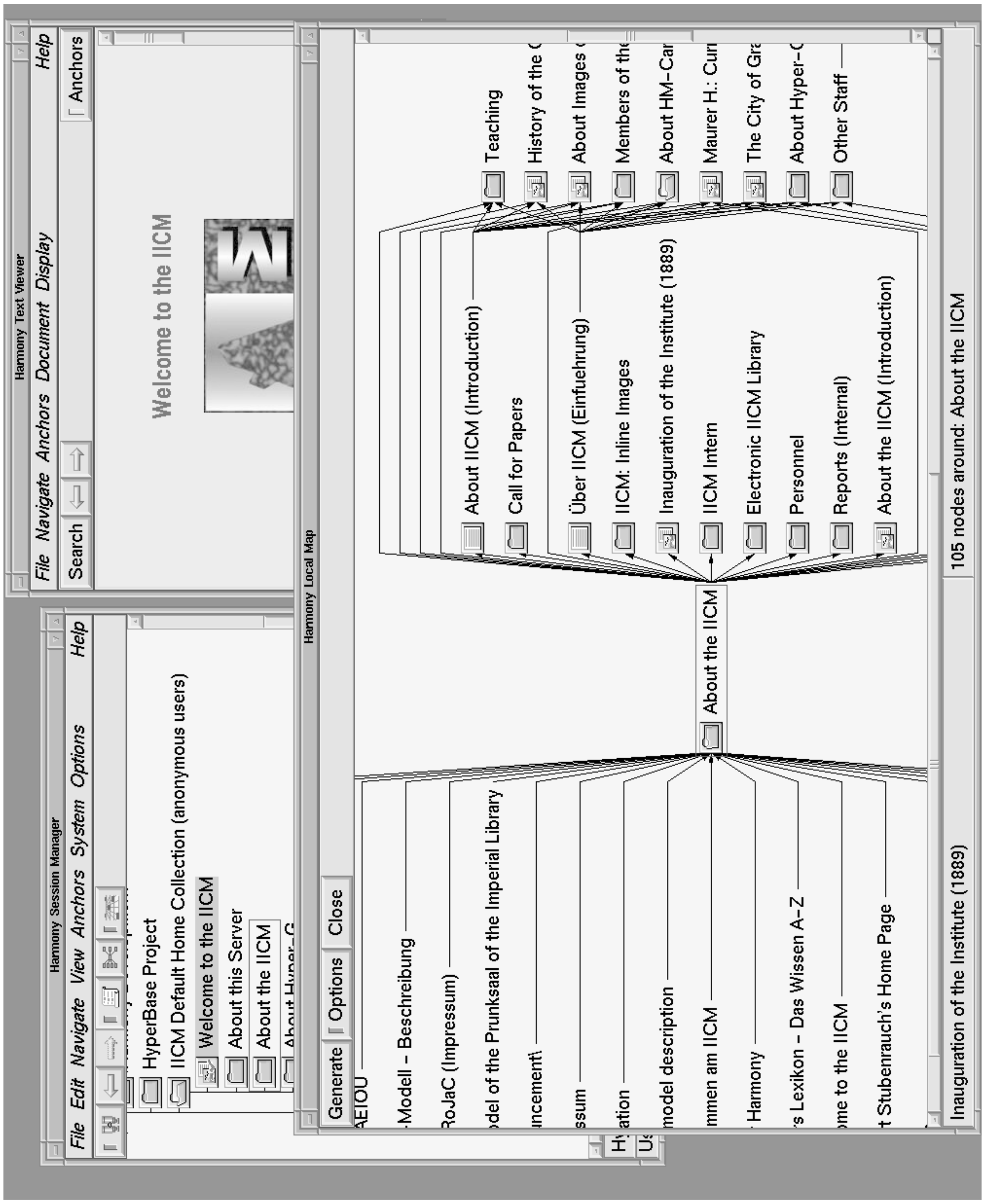


Figure 5.6: The Local Map

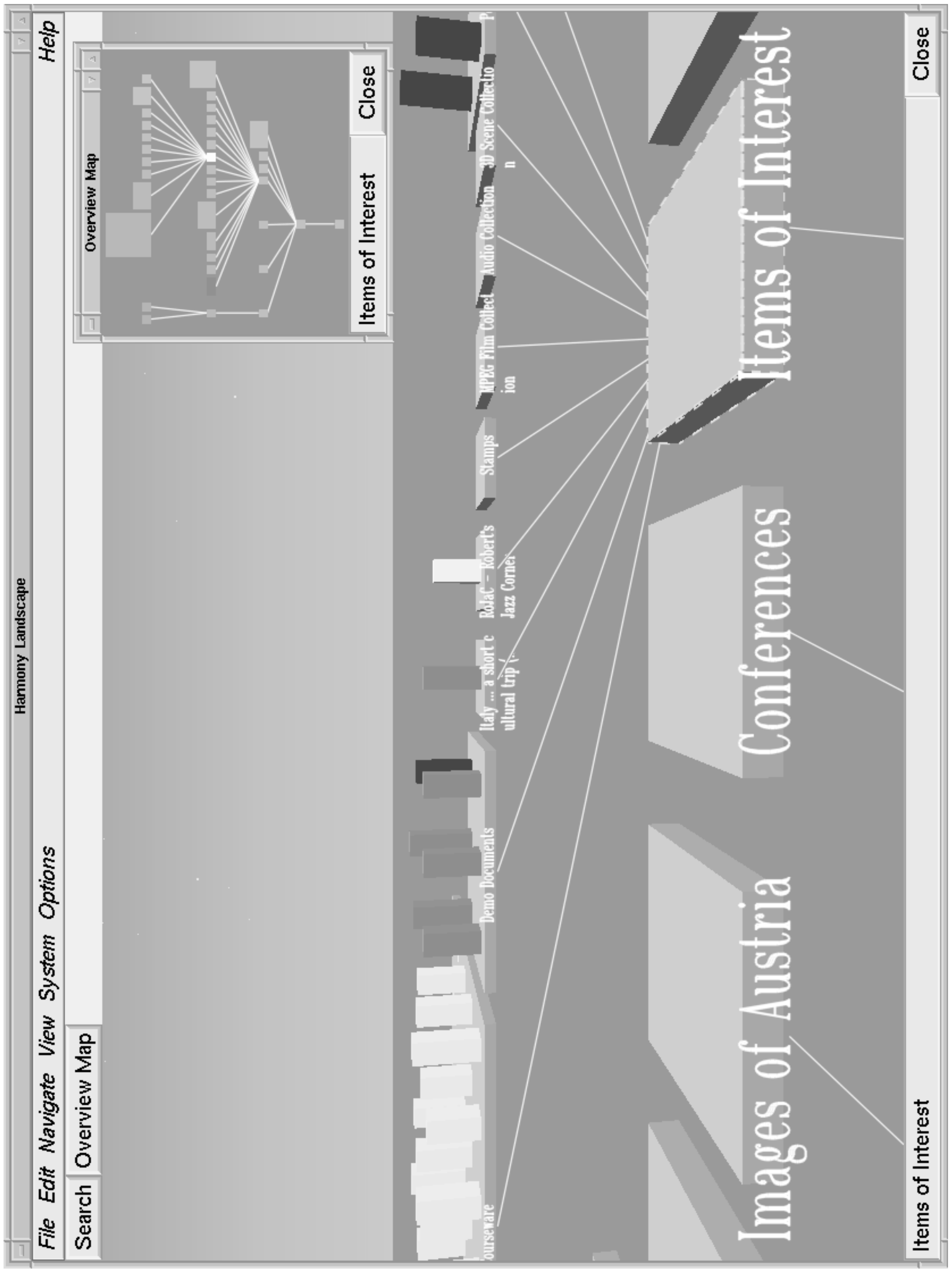
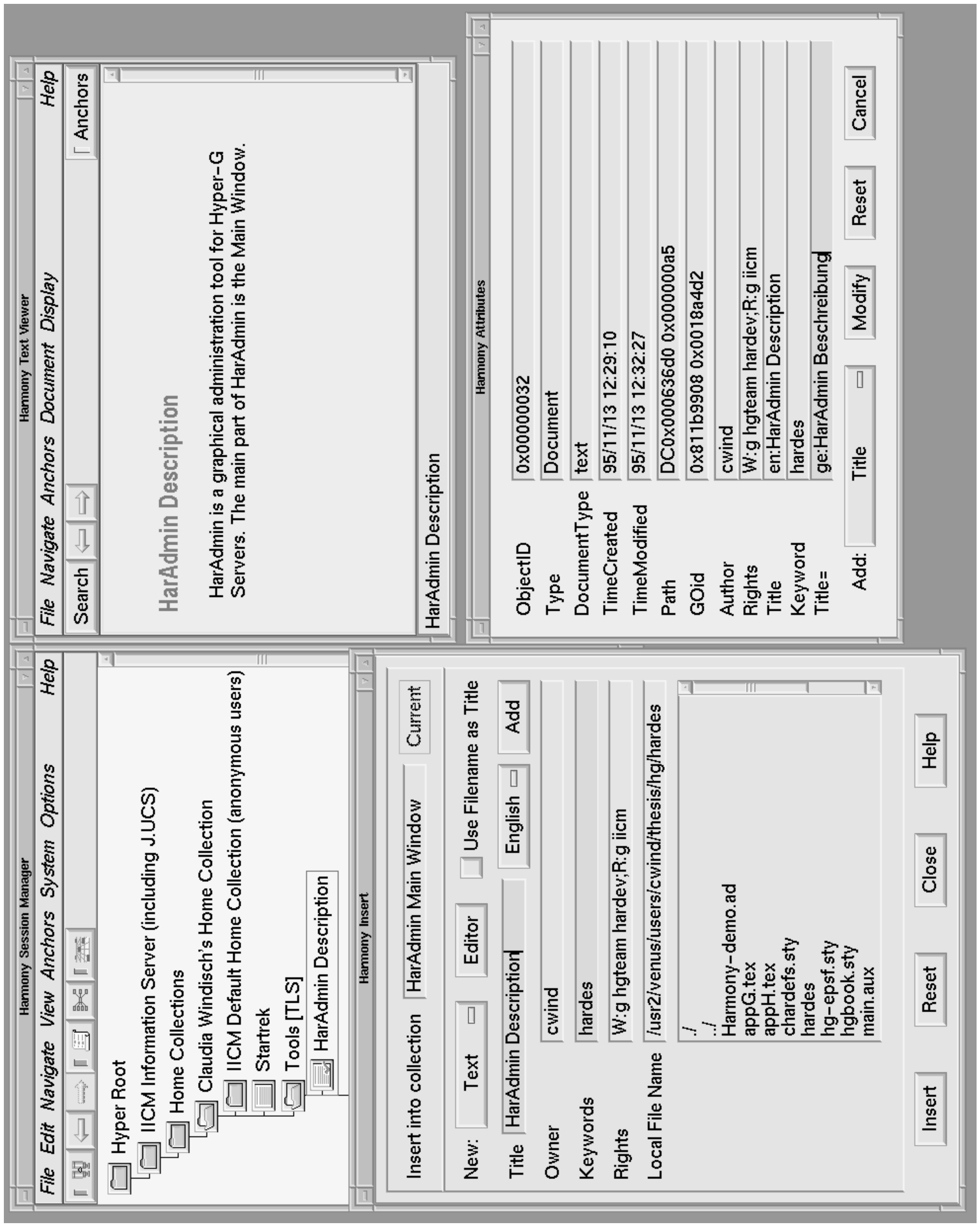


Figure 5.7: The Information Landscape



Hyper Root
IICM Information Server (including J.UCS)
Home Collections
Claudia Windisch's Home Collection
IICM Default Home Collection (anonymous users)
Startrek
Tools [TLS]
HarAdmin Description

Insert into collection	HarAdmin Main Window
New:	Text
Title	HarAdmin Description
Owner	cwind
Keywords	hardes
Rights	W:g hgteam hardev;R:g iicm
Local File Name	/usr2/venus/users/cwind/thesis/hg/hardes

./
../
Harmony–demo.ad
appG.tex
appH.tex
chardefs.sty
hardes
hg–epsf.sty
hgbook.sty
main.aux

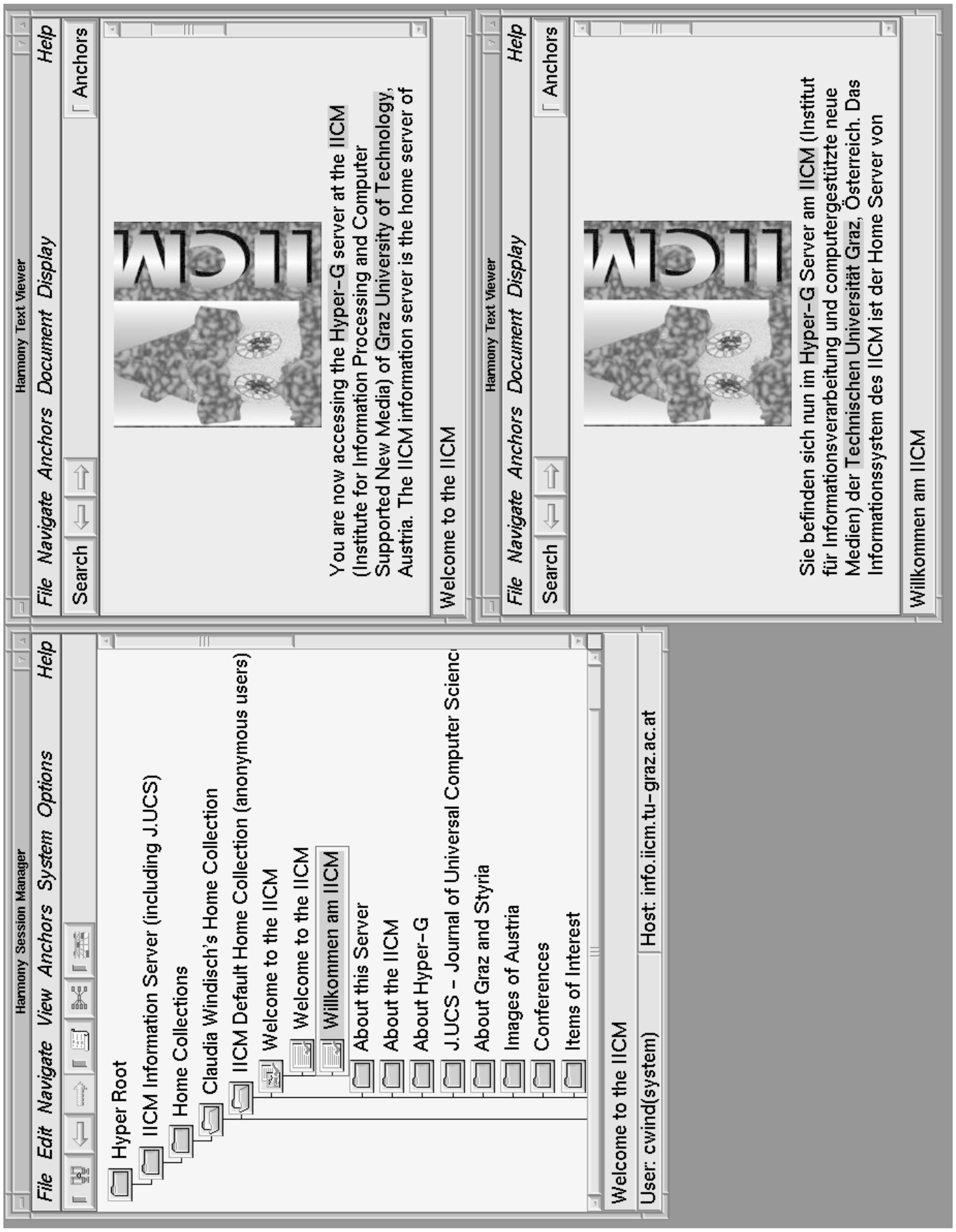
HarAdmin Description

HarAdmin is a graphical administration tool for Hyper–G Servers. The main part of HarAdmin is the Main Window.

HarAdmin Description

ObjectID	0x00000032
Type	Document
DocumentType	text
TimeCreated	95/11/13 12:29:10
TimeModified	95/11/13 12:32:27
Path	DC0x000636d0 0x000000a5
GOid	0x811b9908 0x0018a4d2
Author	cwind
Rights	W:g hgteam hardev;R:g iicm
Title	en:HarAdmin Description
Keyword	hardes
Title=	ge:HarAdmin Beschreibung
Add:	Title

Figure 5.8: Inserting New Objects


Harmony Session Manager
File Edit Navigate View Anchors System Options Help
Harmony Text Viewer
File Navigate Anchors Document Display Help
Search
Anchors
IICM
Harmony Text Viewer
File Navigate Anchors Document Display Help
Search
Anchors
IICM

Hyper Root
IICM Information Server (including J.UCS)
Home Collections
Claudia Windisch's Home Collection
IICM Default Home Collection (anonymous users)
Welcome to the IICM
Welcome to the IICM
Willkommen am IICM
About this Server
About the IICM
About Hyper-G
J.UCS – Journal of Universal Computer Scienc
About Graz and Styria
Images of Austria
Conferences
Items of Interest

Welcome to the IICM
User: cwind(system) Host: info.iicm.tu-graz.ac.at

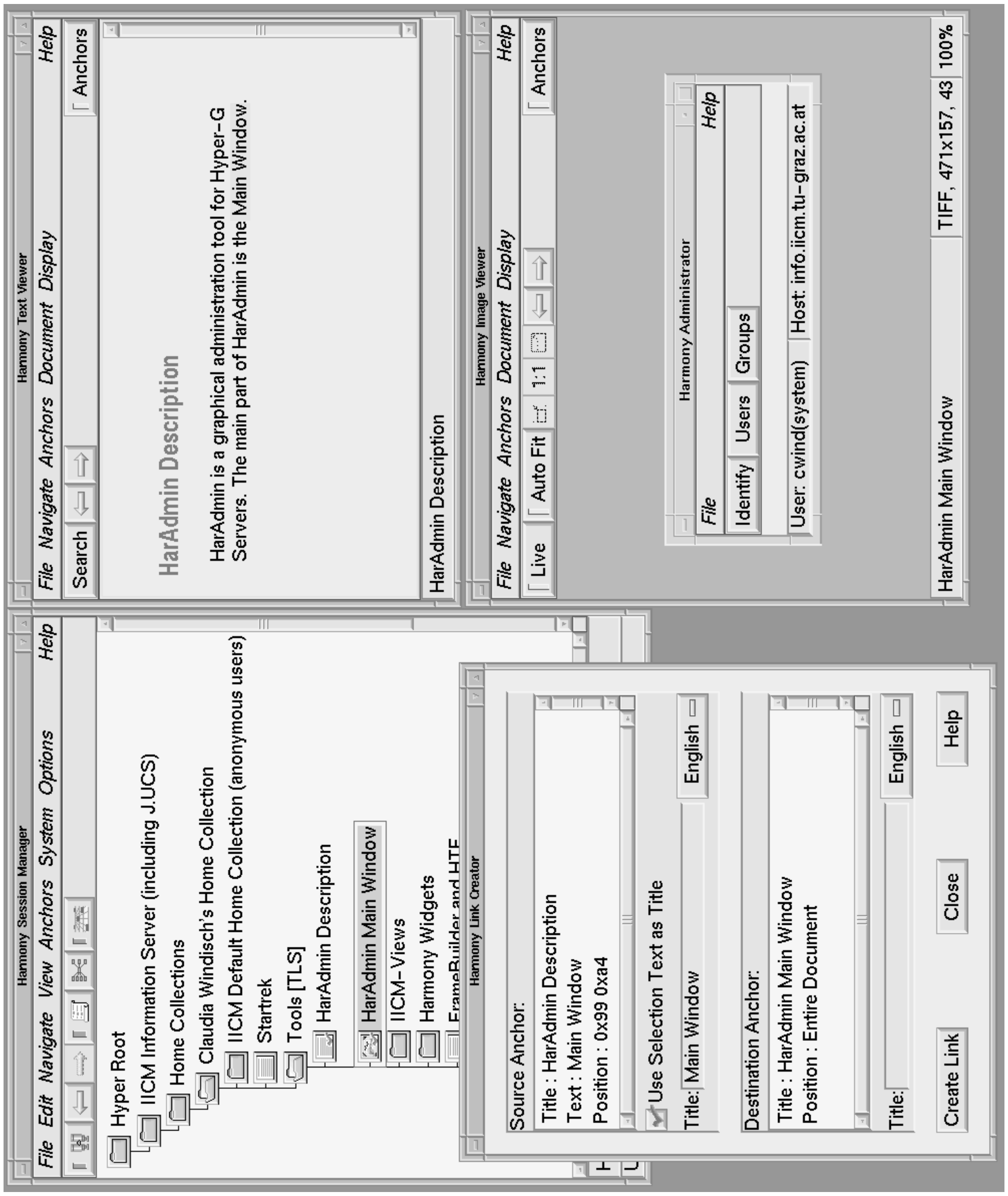
You are now accessing the Hyper-G server at the IICM (Institute for Information Processing and Computer Supported New Media) of Graz University of Technology, Austria. The IICM information server is the home server of

Welcome to the IICM

Sie befinden sich nun im Hyper-G Server am IICM (Institut für Informationsverarbeitung und computergestützte neue Medien) der Technischen Universität Graz, Österreich. Das Informationssystem des IICM ist der Home Server von

Willkommen am IICM

Figure 5.9: The Multilingual Feature



HarAdmin Description

HarAdmin is a graphical administration tool for Hyper–G Servers. The main part of HarAdmin is the Main Window.

HarAdmin Description

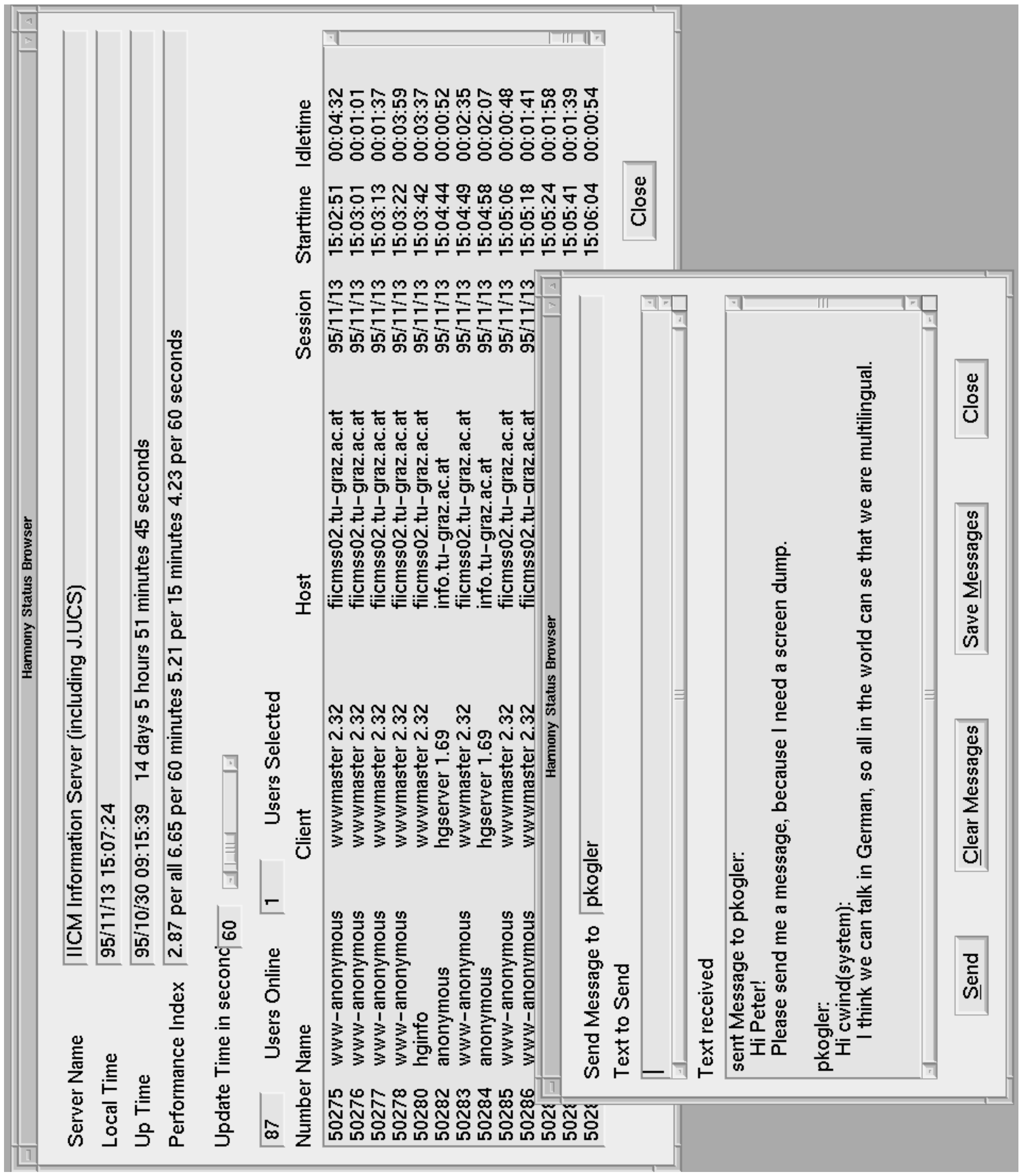
User: cwind(system)
Host: info.iicm.tu-graz.ac.at

HarAdmin Main Window
TIFF, 471x157, 43
100%

Source Anchor:
Title : HarAdmin Description
Text : Main Window
Position : 0x99 0xa4
✓ Use Selection Text as Title
Title: Main Window
English

Destination Anchor:
Title : HarAdmin Main Window
Position : Entire Document
Title:
English

Figure 5.10: Creating Links



Server Name: IICM Information Server (including J.UCS)
Local Time: 95/11/13 15:07:24
Up Time: 95/10/30 09:15:39 14 days 5 hours 51 minutes 45 seconds
Performance Index: 2.87 per all 6.65 per 60 minutes 5.21 per 15 minutes 4.23 per 60 seconds

Update Time in second 60

87 Users Online 1 Users Selected

Number	Name	Client	Host	Session	Starttime	Idletime
50275	www-anonymous	wwwmaster 2.32	fiicmss02.tu-graz.ac.at	95/11/13	15:02:51	00:04:32
50276	www-anonymous	wwwmaster 2.32	fiicmss02.tu-graz.ac.at	95/11/13	15:03:01	00:01:01
50277	www-anonymous	wwwmaster 2.32	fiicmss02.tu-graz.ac.at	95/11/13	15:03:13	00:01:37
50278	www-anonymous	wwwmaster 2.32	fiicmss02.tu-graz.ac.at	95/11/13	15:03:22	00:03:59
50280	hginfo	wwwmaster 2.32	fiicmss02.tu-graz.ac.at	95/11/13	15:03:42	00:03:37
50282	anonymous	hgserver 1.69	info.tu-graz.ac.at	95/11/13	15:04:44	00:00:52
50283	www-anonymous	wwwmaster 2.32	fiicmss02.tu-graz.ac.at	95/11/13	15:04:49	00:02:35
50284	anonymous	hgserver 1.69	info.tu-graz.ac.at	95/11/13	15:04:58	00:02:07
50285	www-anonymous	wwwmaster 2.32	fiicmss02.tu-graz.ac.at	95/11/13	15:05:06	00:00:48
50286	www-anonymous	wwwmaster 2.32	fiicmss02.tu-graz.ac.at	95/11/13	15:05:18	00:01:41
502					15:05:24	00:01:58
502					15:05:41	00:01:39
502					15:06:04	00:00:54

Send Message to pkogler
Text to Send

Text received

```
sent Message to pkogler:
   Hi Peter!
   Please send me a message, because I need a screen dump.

pkogler:
   Hi cwind(system):
      I think we can talk in German, so all in the world can se that we are multilingual.
```

Figure 5.11: The Status Browser of Harmony

5.6 Sending Mails

With Harmony the user has the possibility to archive or send mails or messages:

Hginsmail [Sch95] is a mail archive server written in Perl for archiving mails. Mails are inserted as text documents and if they are related (reply mails) they are linked together. The advantages of this are: you can use the search facilities of Harmony to find certain mails, you can see the thread of discussions, or easily navigate through the mail hierarchy, etc.

With *Hgsendmail* you can send messages to other people, the messages can be hypermedia messages because of the link anchors in a document.

Chapter 6

HarAdmin – The Harmony Administrator

User accounts, groups, home collections, and access rights are central features of Hyper–G. Users in Hyper–G can have their own user accounts, which consist of a username, one or more passwords, and a home collection. Home collections are comparable with UNIX home directories, in which users can manage their own private files (grant or deny access rights to certain users or user groups) and hotlists pointers to resources.

HarAdmin is a tool for administrating user accounts and user groups. It is a graphical X–Window based tool, written in C++ [Str91] and using the InterViews UI ToolKit [LVC89, LCV87]. The basic functionalities are based on the VT100 terminal based tool *hgadmin* [Mau96] and on the *Hyper–G Client/Server protocol* [KP94].

6.1 Setting Up and Running HarAdmin

HarAdmin is not automatically installed with the Hyper–G server software but is part of the *Harmony* client. It is available by anonymous ftp from *ftp.iicm.tu-graz.ac.at* in the directory */pub/Hyper–G/Harmony* for several major platforms:

- SUN Sparc (SunOS and Solaris)
- HP 700 series (HPUX)
- DEC Alpha (OSF/1)
- SGI (IRIX)
- IBM PC compatibles (LINUX)

There are two ways to start HarAdmin:

- stand alone
  By typing `haradmin` in the command line, the user comes to the Main Window shown in Figure 6.1. The default Hyper–G host is info.tu-graz.ac.at with port number 418. Change the default values with

  ```
  haradmin -hghost yourhost -hgport yourport
  ```

  or by setting the following X–Resources:

  ```
  Harmony.Haradmin.hghost: your host
  Harmony.Haradmin.hgport: your port
  ```

- From within Harmony
  By setting up the X–Attribute

  ```
  Harmony.Session.Collection.tools: haradmin
  Harmony.Session.Collection.Tools.haradmin.commandline:
                                haradmin -hghost $host -hgport $port
  Harmony.Session.Collection.Tools.haradmin.menuentry: HarAdmin
  ```

  the user can start HarAdmin from the *File* menu of Harmony.

HarAdmin has following commandline options:
`haradmin [options]`

-h, -help displays usage and defaults

-hghost name name of Hyper–G server

-hgport port port number (usually 418)

The Main Window of HarAdmin

After starting HarAdmin in one of the two above described ways, the Main Window in Figure 6.1 appears. The Main Window is the central point of HarAdmin and the three buttons each activate their own window. The windows and their functionalities are described in the following sections. HarAdmin uses automatic identification if configured, and displays the user name and the current Hyper–G host in the status line of the Main Window.

To exit the administration tool, choose from the *File* menu of HarAdmin the command *Exit Program*.

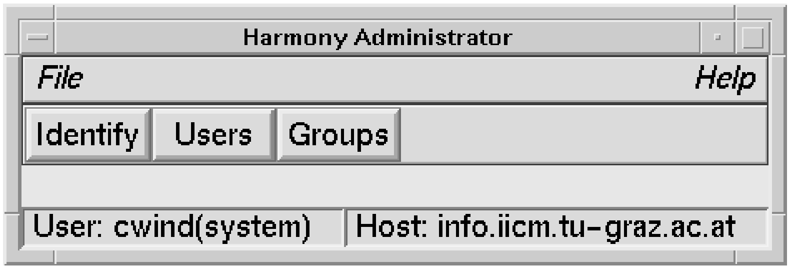


Figure 6.1: The HarAdmin Main Window

Automatic Identification

Having a user account with only username, password, home collection, and groups, implies that the user has to identify manually at the beginning of every Hyper–G session. This can be disturbing, especially when frequently using offline tools. Hyper–G uses automatic identification to prevent this. For automatic identification the username, the machine names and the matching encrypted password of the users Unix account on the client machine are to be added to the user attributes.

6.2 Identifying Yourself to the Server

Hyper–G distinguishes four user identification modes:

1. anonymous user
   Such users have no account names and no passwords and have no chance to see anything about users or user groups on the current Hyper–G server.

   The User and the Group button are disabled.

2. semi-identified user
   The user uses a pseudonym ("nickname") as username and only the system administrator knows the full identity of the user.

3. fully-identified user
   The user has an user account and a password and the identity of such users is generally known.

4. anonymously-identified user
   Have the same rights as an identified user. The difference is that the system administrator also does not know the real identity of such an user. But this mode is not really used in Hyper–G, yet.

Similar to the UNIX superuser, root, a Hyper–G server has a system administrator called `hgsystem`. All members of group `system` have the same rights as the system administrator: they can insert, manipulate, and delete users and user groups. The group `system` and the user `hgsystem` are created when installing the Hyper–G server.

Changing Identity with HarAdmin

To change identity press the Identify button in the Main Window and type in the username and the password, as shown in Figure 6.2.

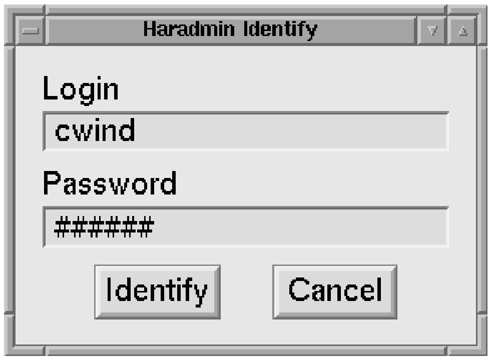


Figure 6.2: The HarAdmin Identify Window

If you change the identity from a system administrator or a member of group `system` you have only to type in the username and no password is needed. To change to an anonymously identified user, type in `anonymous` or `anon` for short.

Users have different possibilities in HarAdmin according to their identification:

- Anonymous:
  For an anonymous user only the Main Window is shown (all other possibly open windows of HarAdmin are closed) and only the Identify button is enabled.
- Identified and not a member of group `system`
  Such users have following privileges:
  - See all users and their attributes except passwords in the User Window
  - See all groups and their attributes in the Group Window
  - Can change their own attributes except adding to a group and changing the account value
- Identified and a member of group `system`
  Such users have all privileges: they can insert, manipulate and delete other users and user groups.

6.3 Managing User Groups

Users may be organized into user groups. User groups in Hyper–G are hierarchically structured and a group can inherit from one or more parent groups. The group hierarchy is like the collection hierarchy, it is an acyclic directed graph. As an example, Figure 6.3 shows a possible group structure for a university.

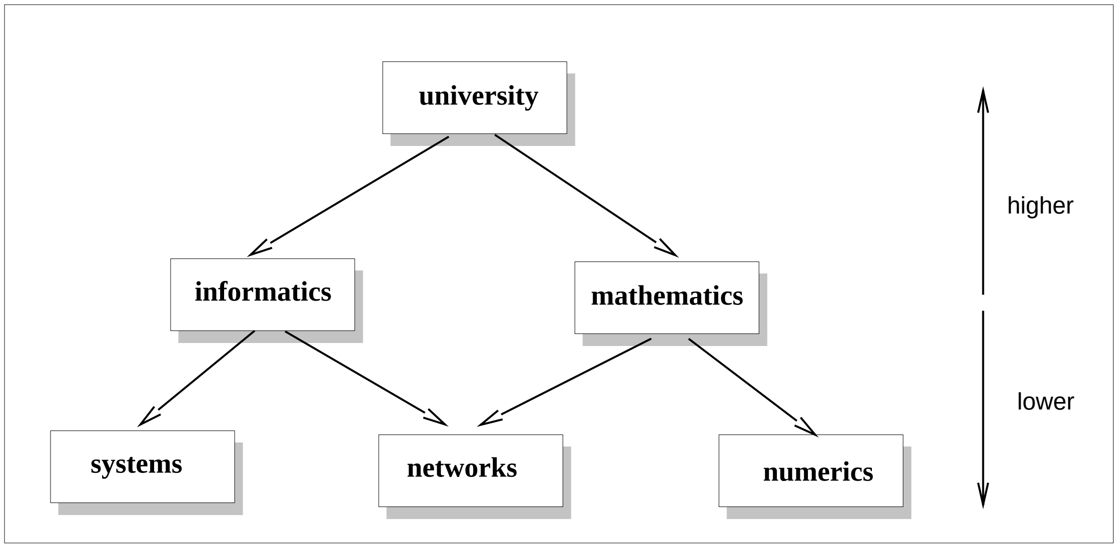


Figure 6.3: A Group Hierarchy for a University

Having an user that is member of the group `numerics` and `informatics` this user has all rights that are given to the groups `university, mathematics, informatics` and `numerics`, but has no access rights of `networks` and `systems`.

In a group hierarchy exists direct and indirect parent groups, and direct and indirect subgroups.

Direct and Indirect Parent Groups

Direct parent groups are the nearest parents of a group. For example group `numerics` has direct parent group group `mathematics`, see Figure 6.3. An indirect parent group is a group with an higher hierarchical level than a direct parent group. For example, group `numerics` has indirect parent group group `university`.

Direct and Indirect Subgroups

Direct subgroups are the nearest subgroups of a group. For example group `university` has direct subgroups group `informatics` and group `mathematics`, see Figure 4.1. An indirect subgroup is a group with an lower hierarchical level than a direct subgroup. For example, group `university` has indirect subgroup group `systems`, group `networks`, and group `numerics`.

To gather information about which groups exist or which attributes an existing group has, press the Group button in the Main Window (Figure 6.1) and the Group Window shown in Figure 6.4 appears.

6.3.1 The Group Window of HarAdmin

The function of the Group Window is to display several groups and the attributes of a selected group. With the topmost button bar you can

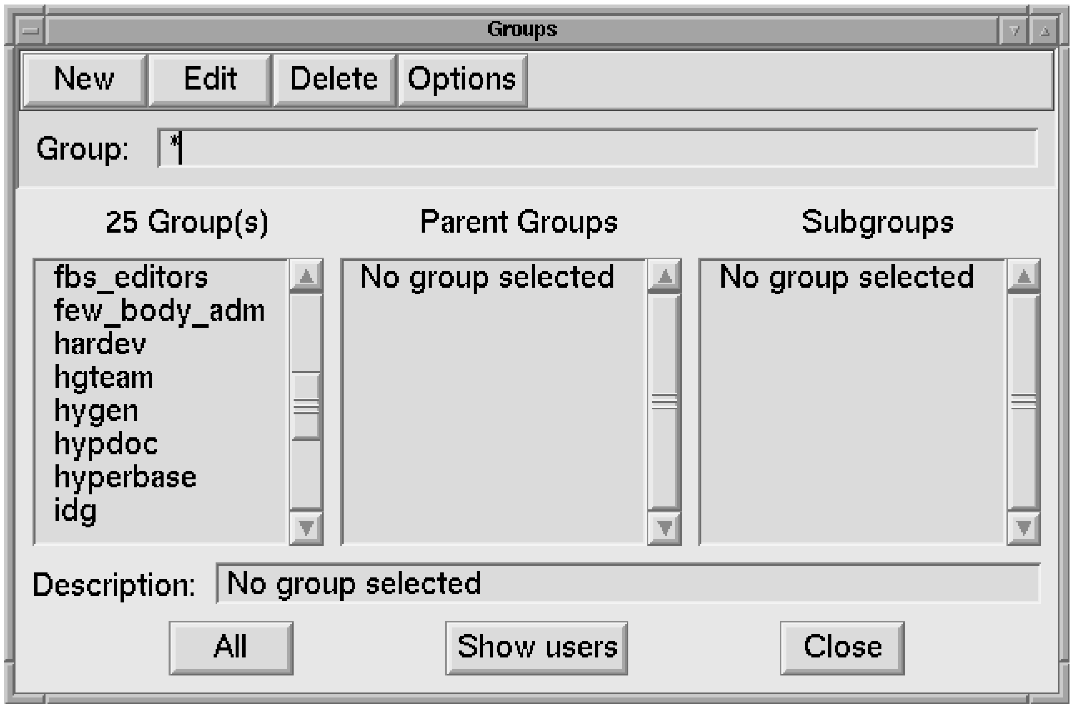
– Create new groups

– Edit existing groups

– Delete existing groups

– Change the look of the Group Window

The first time the Group Window appears, the `Group` field is set to "*". And after pressing `Enter` all groups of the current Hyper–G server are displayed.



Group: *

25 Group(s)

fbs_editors
few_body_adm
hardev
hgteam
hygen
hypdoc
hyperbase
idg

Parent Groups

No group selected

Subgroups

No group selected

Description: No group selected

Figure 6.4: The HarAdmin Group Window

Rows and Columns of the Group Window

Only the field `Group` is editable and froms the *selection string* for searching the current Hyper–G group database. To change the selection, the `Group` field is edited and after pressing `Enter` the new group(s) are displayed.

If only one group matches, the group is selected (highlighted) and all attributes of the group are displayed. Otherwise all groups are shown, and by double clicking the left mouse button the attributes of the selected group are shown. For example, to display groups with the prefix "x", type in the `Group` field "x*" and press `Enter`.

Group(s): A list of all currently matching groups on the current Hyper–G server. The number of matching groups is displayed in the line above the column.

Parent Group(s): A list of all groups, of which the selected group is a member. As said before Hyper–G structures groups in a hierarchy. Beside the "root" group all groups can have parent and subgroups. Indirect groups are shown in form "→ *groupname*".

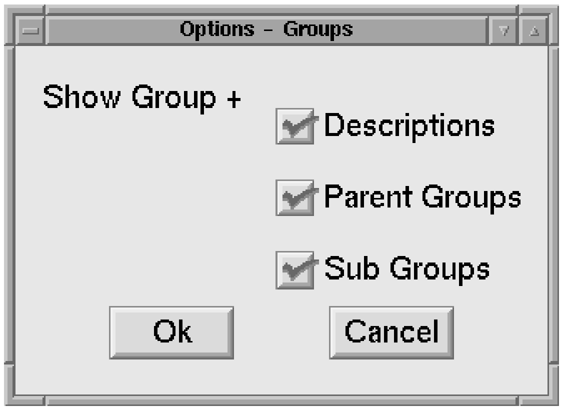
Subgroup(s): A list of all subgroups. Indirect groups are shown in the form "→ *groupname*".

Description: A short description of the selected group, usually the meaning of the group. There can be more than one description of a group, but only one description (the last) is displayed in the Group Window.

Special feature: You also can select a group in the columns Parent Group(s) and Subgroup(s) to see its the attributes.

Button Options

You can change the look of the Group Window either by X–Defaults or with the Options Window shown in Figure 6.5. The corresponding X–Defaults are listed in Appendix A.


Options – Groups

Show Group +
[✓] Descriptions
[✓] Parent Groups
[✓] Sub Groups

Figure 6.5: Group Window: Options Group

Button All

Changes the selection string to "*" and displays all existing groups on the current Hyper–G server in the Group Window.

Button Show Users

Selecting a group in the column Group(s) and pressing the button Show-User displays all direct and indirect users of the selected group in the User Window.

Button Close

Closes the Group Window but does not destroy it. If you press the Group button in the Main Window, see Figure 6.1, the previous selection string is reused to display all groups according to the selection string.

6.3.2 Creating a New Group

Only the system administrator and members of group `system` are allowed to create new groups.For users who are not members of the system group the New button in the Group Window is disabled. Groups should be created before user accounts, because users are members of at least one existing group. A group comprises the following attributes:

Group: A unique name of the group and which must be a single word. The group name is mapped to lower case letters on insertion.

Parent: Name of the parent group. This attribute is used to create a group hierarchy.

To add additional attributes, choose the corresponding one from the selection menu above the buttons Commit, Clear, Close. The focus is set automatically.

Button Commit

Sends the group object to the database for insertion. The field values of the new group are not cleared (except for the group name). This is useful if you add multiple groups with same attributes. If an error occurs, a message box is displayed and the focus is set to the attribute which contains the wrong values.

Button Clear

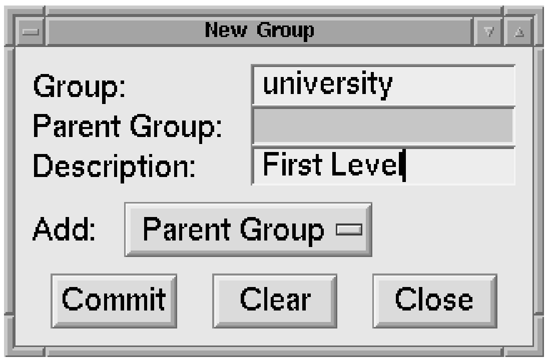
Clears all entries in the New Group Window. Once a field has been edited, the background colour of the field is changed to indicate an edited value.

Button Close

Closes the New Window, but does not destroy it. If you create a new group again, the old entries are displayed (except the group name).

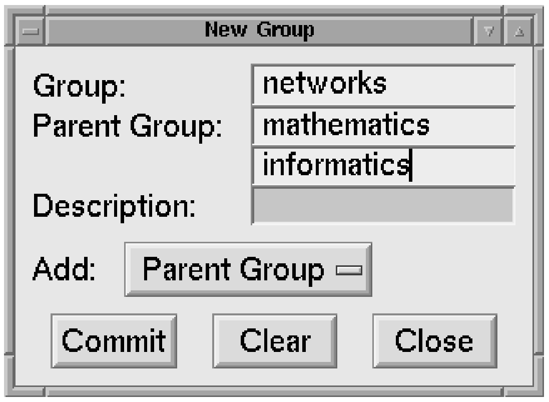
To create the structure shown in Figure 6.3 press the New button in the Group Window and type in the attributes shown in Figure 6.6 and Figure 6.7.

Because the group `university` is the first in the hierarchy, no parent group has to be entered. After pressing the Commit button (assuming no error occurs), the group `university` is created on the current Hyper–G server. The newly created group is automatically selected as the current group and the group's attributes are displayed in the Group Window. The remaining groups are created analogously:



Group: university
Parent Group:
Description: First Level

Figure 6.6: Creating the Group "university"



Group: networks
Parent Group: mathematics
informatics
Description:

Figure 6.7: Creating the Group "networks"

university: no parent group(s)

informatics: parent group : university

mathematics: parent group : university

systems: parent group : informatics

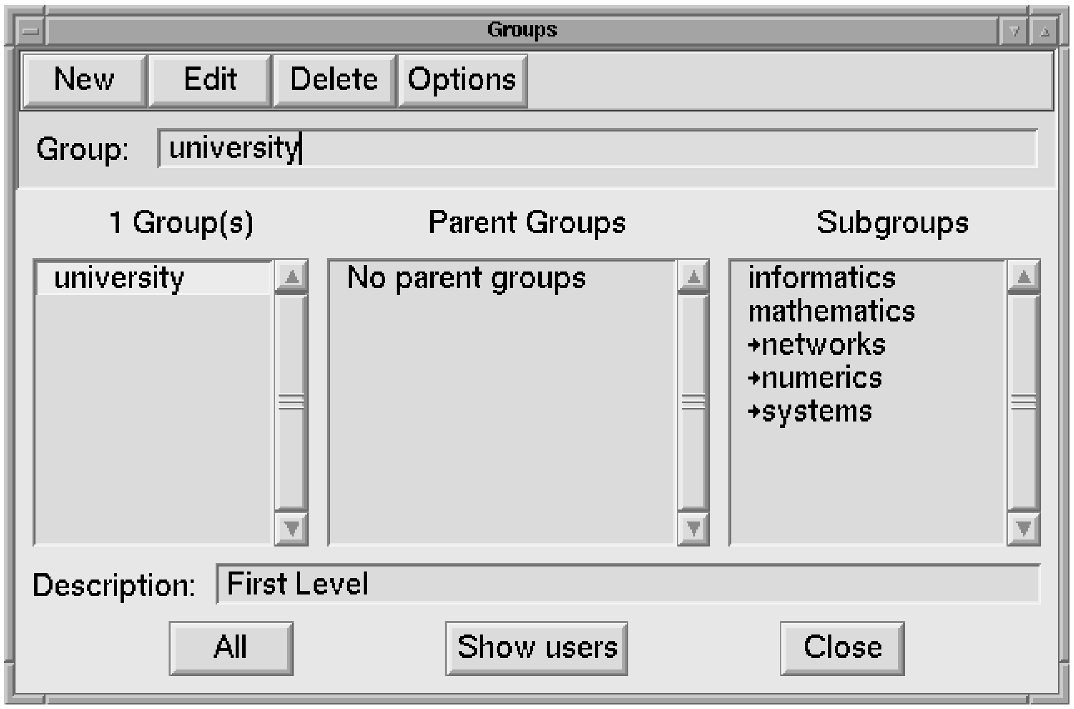
networks: parent groups : informatics and mathematics

numerics: parent group : mathematics

The complete hierarchy in Figure 6.3 is shown in the window of Figure 6.8 by selecting the group `university`. By selecting for example group `informatics` all columns are filled, as illustrated in Figure 6.9.

6.3.3 Editing an Existing Group

Only members of group `system` are allowed to change the attributes of an existing group. Select the group to be edited in the column Group(s) and press the Edit button. Overwriting an existing attribute changes the attribute, clearing the field



Group: university

1 Group(s)	Parent Groups	Subgroups
university	No parent groups	informatics
		mathematics
		→networks
		→numerics
		→systems

Description: First Level

Figure 6.8: Displaying the Group "university"

deletes the attribute. Note that you cannot easily change the name of the group. To change the name, the group must be deleted and created new again.

6.3.4 Deleting A Group

In contrast to deleting users, only one group can be selected in the column Group(s). The group hierarchy makes deleting a group much more complicated than deleting one or more users. Before a group can be deleted two conditions have to be fulfilled:

- All direct users of the group have to be deleted.
- All direct subgroups of the group must be unlinked or deleted.

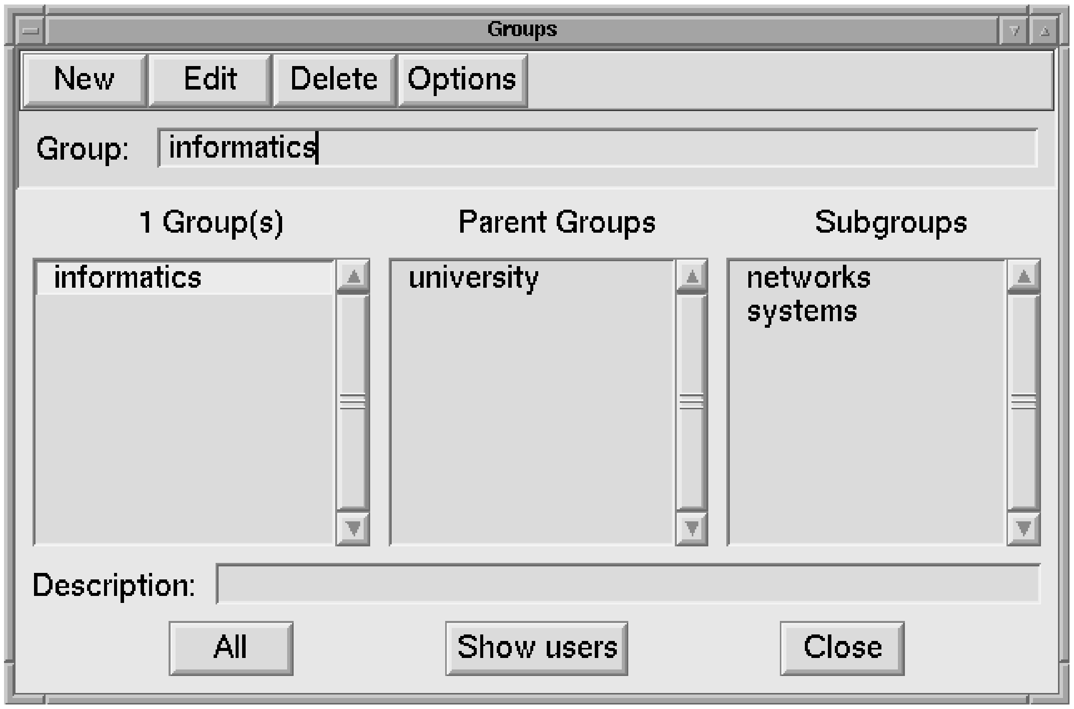
If at least one of the conditions is not fulfilled the Delete Dialog shown in Figure 6.10 appears:

Subgroups exist

If one or more subgroups exists, the column Subgroup(s) in the Group Window is filled, and all direct subgroups are highlighted.

If the parent of the group is deleted and the group itself not, than the group is simple unlinked from the parent.

Users exist



Group: informatics

1 Group(s)	Parent Groups	Subgroups
informatics	university	networks
		systems

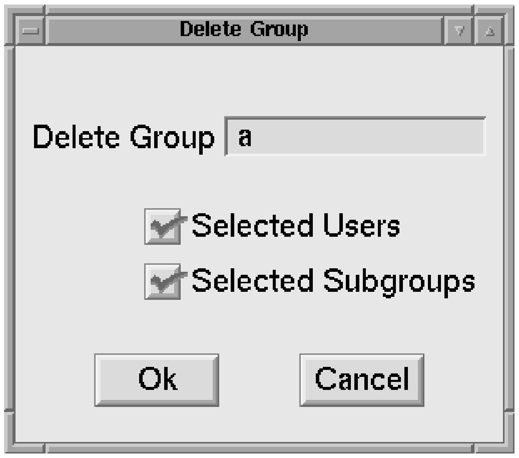
Description:

Figure 6.9: Displaying the Group "informatics"

All existing users are displayed in the column User(s) in the User Window. Direct users are also highlighted. If at least one direct user is present, the group cannot be deleted.

The user must either be selected for deletion or removed from the group using the Edit User Window (delete the group attribute).

If all conditions are fulfilled the selected group is deleted, the selection string is set to "*", and all remaining groups are displayed in the Group Window.



Delete Group a

[✓] Selected Users
[✓] Selected Subgroups

Figure 6.10: Group Window: Delete Dialog

6.4 Managing Users

A Hyper–G user account is not identical to a Unix user account, a user can have different Unix accounts on different machines but access a single Hyper–G account.

To gather information about which users exist or which attributes a certain user has on the current Hyper–G server, press the `User` button in the Main Window (Figure 6.1) and the User Window in Figure 6.11 appears. A user account is at least made up a username, one or more passwords, and a personal home collection.

6.4.1 The User Window of HarAdmin

The function of the User Window is displaying several user accounts and attributes of the currently selected user. With the topmost button bar you can

– Create new users

– Edit existing users

– Delete multiple users

– Change the look of the User Window

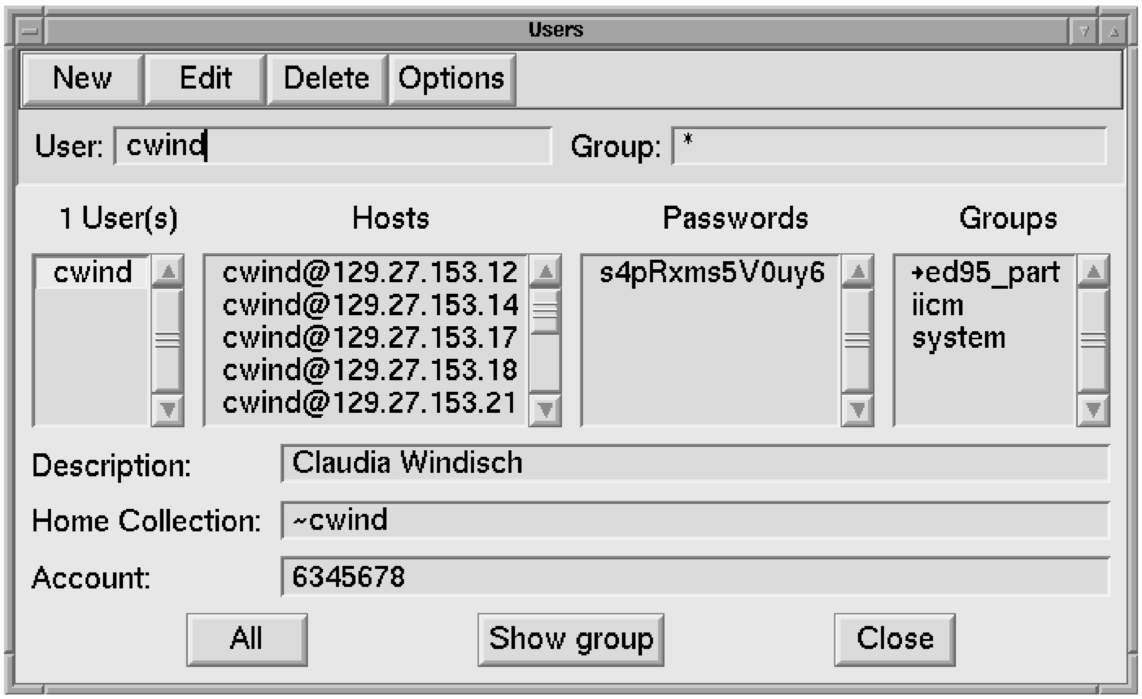
The first time the User Window appears, the `User` field is set to the name of the identified user, the `Group` field is set to "*", and all attributes of the user are shown automatically, see Figure 6.11.

Rows and Columns of the User Window

Only the fields `User` and `Group` are not read–only. They together build the *selection string* for searching the Hyper–G user database. To change the selection, edit the `User` and/or the `Group` field and press `Enter`. During searching, the mouse pointer changes to an hour–glass and no other database commands can be made. After a short time the matching user(s) are displayed and the mouse cursor changes back to a pointer. If only one user matches, the user is marked and the user's attributes are shown automatically, otherwise all matching users are displayed and only by double clicking the left mouse button are the attributes of the selected user shown.
For example:

1. To see all users with user names beginning with 'c' type in the `User` field "c*", in the `Group` field "*", and press `Enter` to send the selection string to the database.

2. To see all users of group 'iicm', set the `User` field to "*", the `Group` field to "iicm", and press `Enter` to send the selection string to the database.



User: cwind
Group: *

1 User(s): cwind

Hosts:
cwind@129.27.153.12
cwind@129.27.153.14
cwind@129.27.153.17
cwind@129.27.153.18
cwind@129.27.153.21

Passwords: s4pRxms5V0uy6

Groups:
→ed95_part
iicm
system

Description: Claudia Windisch
Home Collection: ~cwind
Account: 6345678

Figure 6.11: The HarAdmin User Window

User(s): All users according to the selection string = `User` + `Group` field. The number of matching users is displayed in the line above the column.

Hosts: Is a list of hosts from which automatic identification is possibly. An entry is in form of `user@host`. A host is displayed in one of the following formats:

- Domain Name:
  for example: fiicmal01.tu-graz.ac.at
- IP-Address:
  for example: 129.27.153.18
- Alias Name:
  for example: jupiter

The display format can be specified by X–Defaults or by selection in the Option Window (see Figure 6.12). The corresponding X–Defaults are listed in Appendix A.

Passwords: A list of encrypted passwords.

Groups: A user is a member of at least one group. Hyper–G uses a *group hierarchy* to coordinate access rights. For all indirect parent groups HarAdmin uses the form "→ *groupname*". Only members of group `system` have the right to add users to groups.

Account: Hyper–G supports billing in sofar as documents can have a price. A user has a certain amount of virtual money to access charged documents. Only members of group `system` have the right to change the Account attribute for a certain user. The server decrements this value whenever the user accesses a priced document, until the value would become negative. In this case, the document is not transferred and the server sends an error code.

Description: A short description of the selected user, usually the full name of the user. There can be more than one description for a user, but only one description (the last) is displayed in the User Window.

Home Collection: In Hyper–G a user generally has a personal home collection. Such a home collection is comparable to a home directory under UNIX and a hotlist. A home collection can be created by using Hyper–G clients such as *Harmony, Amadeus*, or *hgtv*.

Button Options

Either by X–Defaults or with the Option Window shown in Figure 6.12, you change

- The look of the User Window by enabling or disabling certain columns and lines in the User Window.
- The display format for hosts.

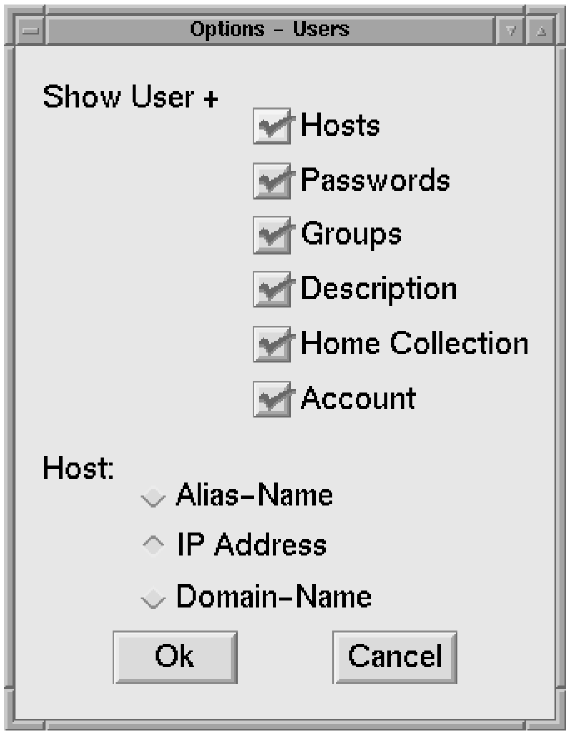


Figure 6.12: The HarAdmin User Option Window

Button All

This button changes the selection string. `User` and `Group` field are set to "*" and the User Window is automatically updated.

Button Show Group

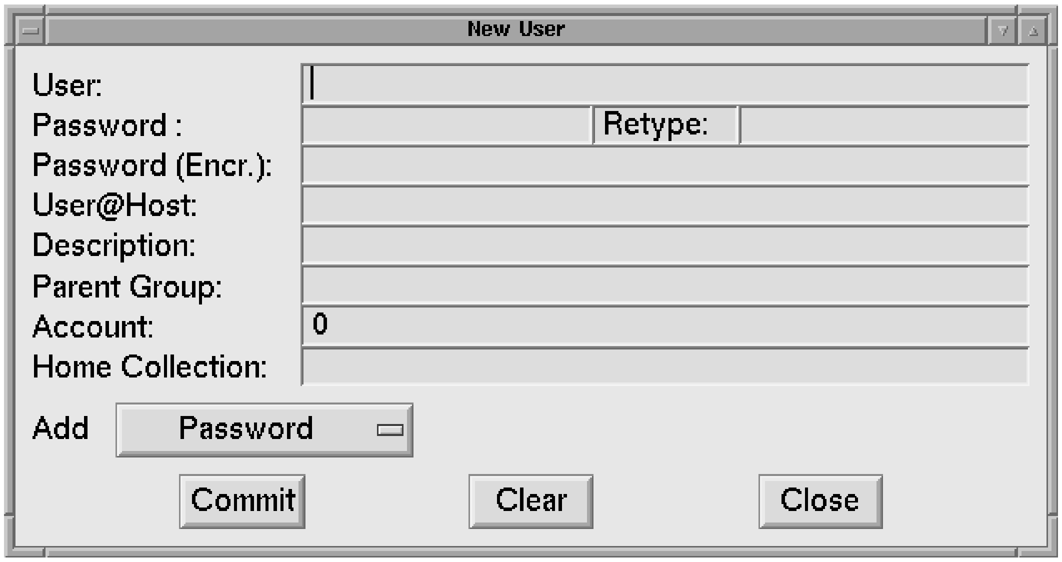
Selecting a group in the column Group(s) and pressing this button, the Group Window is updated, all attributes of the selected group are shown. You also can double click with the left mouse button on the group in the column Group(s) to display the group in the Group Window.

Button Close

Unmaps the User Window, but does not destroy it. If you press the User button in the Main Window again, the old filters are reused to display all users of the current Hyper–G server according to the selection string in the User Window.

6.4.2 Creating a New User

Only the system administrator and members of group `system` are allowed to create new users, for users who are not members of the system group the New button in the User Window is disabled. Pressing the New button in the User Window the New User Window shown in Figure 6.13 is displayed.



New User

User:	
Password :	Retype:
Password (Encr.):	
User@Host:	
Description:	
Parent Group:	
Account:	0
Home Collection:	

Figure 6.13: The HarAdmin New User Window

The fields correspond to attributes of the Hyper–G user object:

User: The name is unique for the current Hyper–G server and has to be a single word. The username is mapped to lower case letters on insertion.

Password: The password is entered in clear text. Individual characters are echoed as '#'. The password must be entered twice to counter possible typing mistakes.

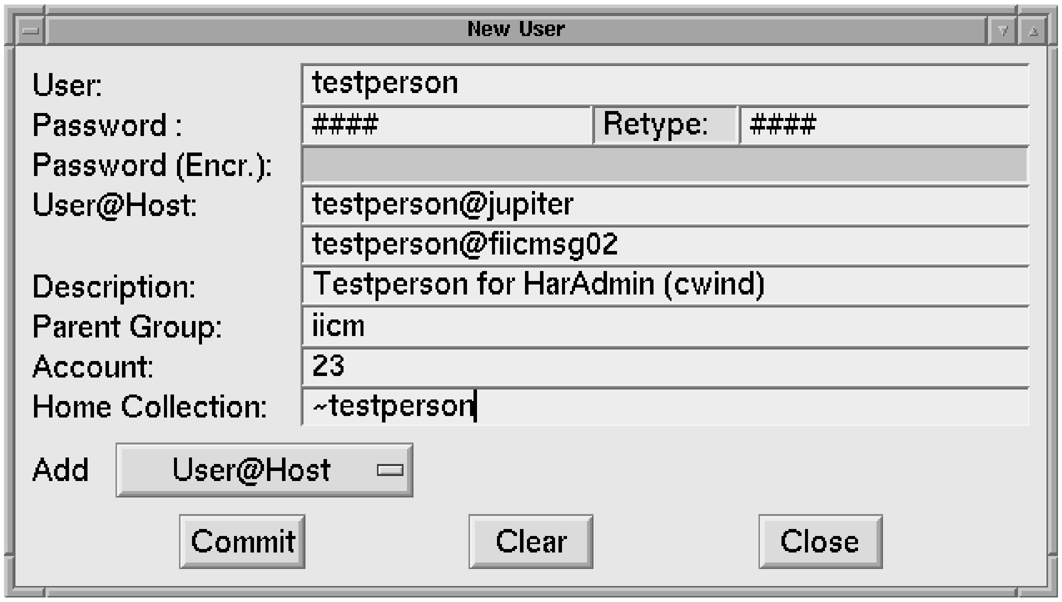
Password encrypted: The encrypted password is stored under UNIX in the file `/etc/passwd` or you can get the encrypted password with the SUN Yellow Pages command `ypcat passwd`. The advantage of this method is that the system administrator can set up a user with the user's Unix password without knowing the password itself.

User@Host: If the host is unknown or already exists a message box appears and after pressing OK, the focus is set to the corresponding line. The host can be entered as Domain Name, IP Address or Alias Name (HarAdmin maps the host names to IP Addresses).

Home Collection: A home collection has the form "˜ " followed by the name of the new user.

To add more attributes, choose the corresponding one from the selection menu above the buttons Commit, Clear, Close. The focus is set automatically.

A complete new user can look like the one shown in Figure 6.14.



User:	testperson		
Password :	####	Retype:	####
Password (Encr.):			
User@Host:	testperson@jupiter		
	testperson@fiicmsg02		
Description:	Testperson for HarAdmin (cwind)		
Parent Group:	iicm		
Account:	23		
Home Collection:	~testperson		

Figure 6.14: An Example of a Complete New User

Insert a new user by pressing the Commit button. If no error occurs, the entries can be inspected in the User Window. The newly created user is automatically selected as the current user and the user's attributes are displayed in the User Window.

Button Commit

Sends the user objects to the database for insertion. The values of the new user are not cleared (except the user name). This is useful for adding multiple users with the same attributes (eg. Hosts, Groups) to the database. If an error occurs a message box is displayed and the focus is set to the attribute which contains wrong values.

Button Clear

Clears all field entries in the New User Window. Once a field has been edited, the background colour of the field is changed to indicate an edited value.

Button Close

Closes the New User Window, but does not destroy it. If you create a new user again, the old entries are displayed (except the user name).

6.4.3 Editing an Existing User

Only members of group `system` are allowed to change the attributes of other users. Select the user to be edited in the column User(s) and press the Edit button. Adding attributes is done in the same way as in the New User Window. Overwriting an attribute changes the value, clearing the field deletes the attribute. Note that you cannot easily change the name of a user (as for all other Hyper–G objects). To change the name of the user, the user must be deleted and created new again.

6.4.4 Deleting Users

To delete one or more users on the current Hyper–G server, one or more users have to be selected in the column User(s) in the User Window and afterwards the Delete button must be pressed.

By shift–left–clicking users, multiple users can be selected and deselected for deletion.

6.5 Example of Group Deletion

Here is an example to show the way in which a group with members and subgroups is deleted. Figure 6.15 shows a group hierarchy and Figure 6.16 the corresponding Group Window with selected group `a`. Groups `b` and `c` are direct subgroups and groups `d`, `e`, `f` are indirect subgroups (indicated by "$\rightarrow$ *groupname*").

Assume that we want to delete

- group a
- all direct members of group a
- direct subgroup b
- all direct members of group b
- indirect subgroup d
- all direct members of group d.

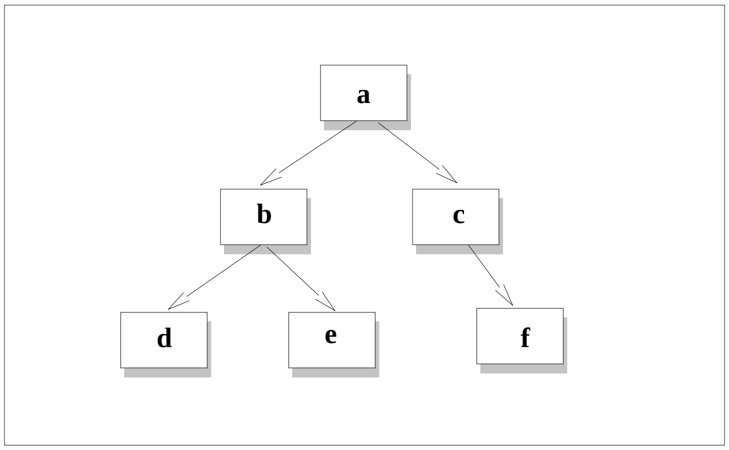


Figure 6.15: Example: Group Hierarchy for Deleting Group “a”

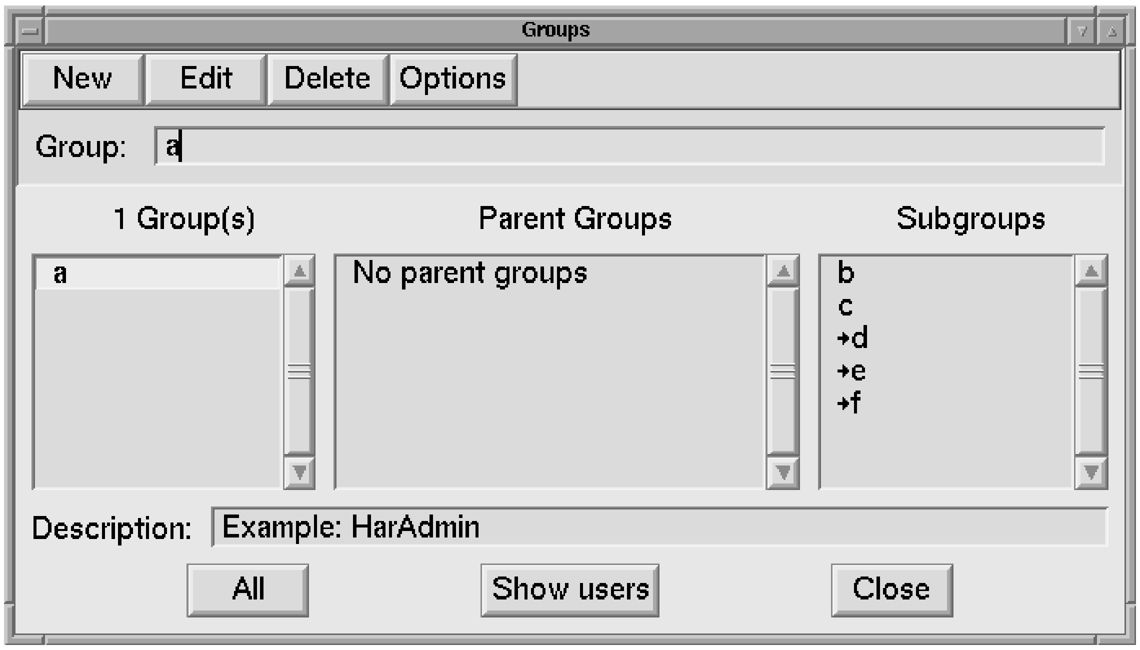
In this example each group has two users named by ”group name#”, e.g. group a has members a1 and a2.

To delete group a, select group a in the column Group(s) and press the Delete button in the Group Window. Group and User Windows are updated and the Delete Dialog appears, see Figure 6.17. Remember, direct subgroups and direct users are highlighted.

The Group b and group c are direct subgroups and highlighted, but as said before we want to delete group b and also the indirect group d.

Unselect group c by holding the shift key and clicking the left mouse button on group c, select group d by holding the shift key and clicking the left mouse button on group d. We also must select the direct users of group d: d1 and d2, otherwise group d can't be deleted (and same for group b), see Figure 6.18.

Pressing the Ok button in the Delete Dialog all selected subgroups and users are deleted. The new hierarchy after deleting groups a, b, and d is shown in Figure 6.19

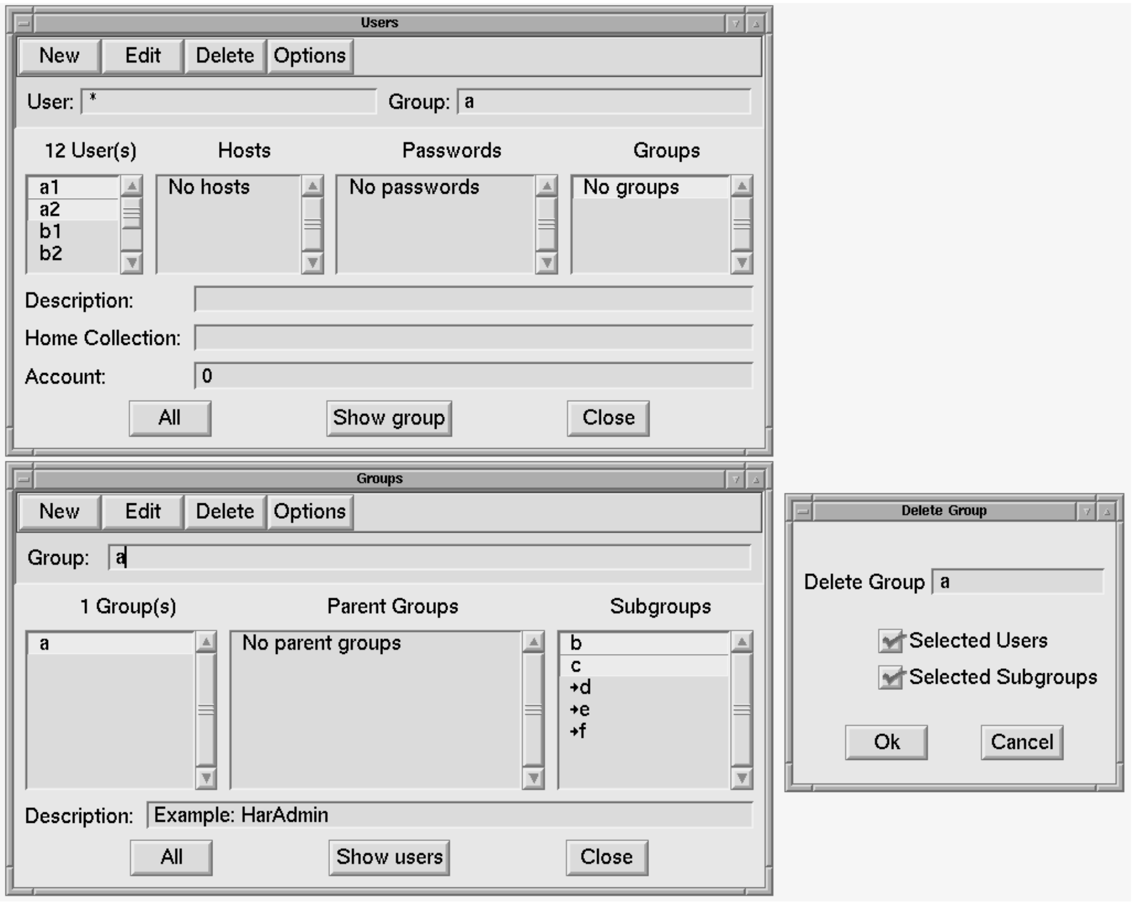


Group: a

1 Group(s)	Parent Groups	Subgroups
a	No parent groups	b
		c
		→d
		→e
		→f

Description: Example: HarAdmin

Figure 6.16: Displaying the Group “a”



User: * Group: a

12 User(s)	Hosts	Passwords	Groups
a1	No hosts	No passwords	No groups
a2			
b1			
b2			

Description:

Home Collection:

Account: 0

Group: a

1 Group(s)	Parent Groups	Subgroups
a	No parent groups	b
		c
		→d
		→e
		→f

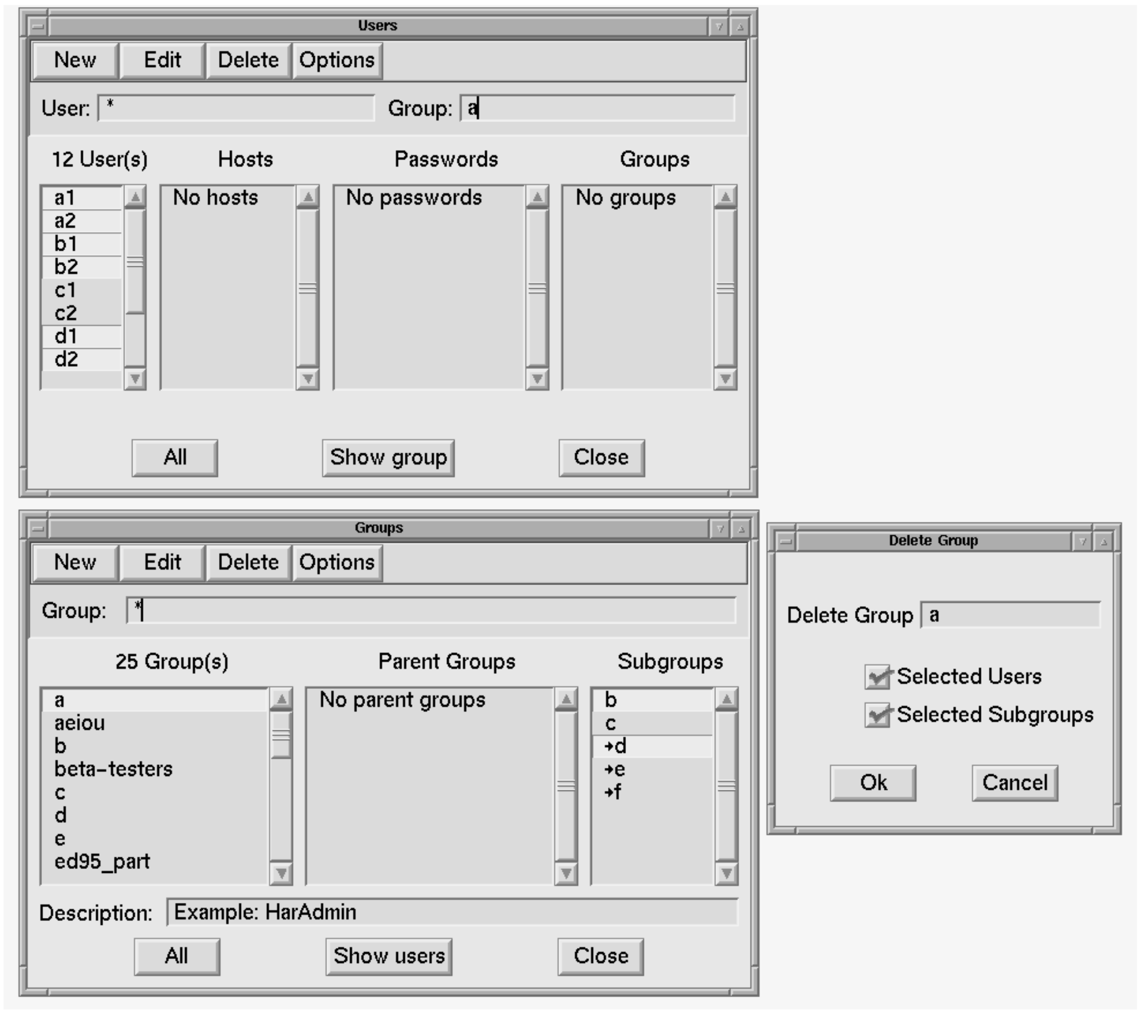
Description: Example: HarAdmin

Delete Group a

✓ Selected Users

✓ Selected Subgroups

Figure 6.17: Example: User Window, Group Window, and Delete Dialog



User: *
Group: a

12 User(s): a1, a2, b1, b2, c1, c2, d1, d2
Hosts: No hosts
Passwords: No passwords
Groups: No groups

Group: *

25 Group(s): a, aeiou, b, beta-testers, c, d, e, ed95_part
Parent Groups: No parent groups
Subgroups: b, c, →d, →e, →f

Description: Example: HarAdmin

Delete Group a
✓ Selected Users
✓ Selected Subgroups

Figure 6.18: Example: Windows for Deleting Group “a”, “b”, “d” , and all direct users

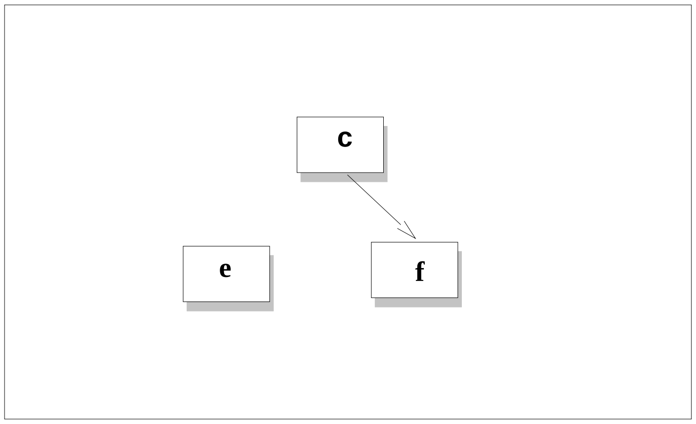


Figure 6.19: Example: Group Hierarchy after Deleting Groups “a”, “b” and “d”

The procedure for deleting groups `a, b` and `d`:

1. All selected users are deleted
2. Group `d` is deleted if no direct users of group `d` exist.
3. Group `e` is unlinked from group `b`.
4. Group `c` is unlinked from group `a`.
5. Group `b` is deleted if no direct users of group `b` exist.
6. Group `a` is deleted if no direct users of group `a` exist.
7. Group Window and User Window are updated with selection strings = "*".

If a group cannot be deleted (usually direct users exist), an error message is displayed.

6.6 Getting And Changing Information without HarAdmin

It is possible to use other Hyper–G utilities to obtain information about users and groups direct from the database

For example, to display all existing users on the current Hyper–G server using the *hginfo* utility:

```
hginfo -hghost yourserver -hgport yourport \
       -formatted -key 'UName=*' -attr UName -mult
```

To display all existing groups of the current Hyper–G server:

```
hginfo -hghost yourserver -hgport yourport--Gport \
       -formatted -key 'UGroup=*' -attr UGroup -mult
```

To change an attribute of a certain user on the current Hyper–G server you can use the *hgmodify* utility:

For example, to add a new description for user `testperson` to the user object:

```
hgmodify -hghost yourserver -hgport yourport -formatted \
         -key 'UName=testperson' -comm 'add Descr=New Description'
```

To see all descriptions of user `testperson` using *hginfo*:

```
hginfo -hghost yourserver -hgport yourport -formatted\
       -key 'UName=testperson' -attr Descr
```

As in HarAdmin only the system administrator or members of the group `system` receive the whole information.

For more detailed information about *hginfo* and *hgmodify* see [Mau96].

Chapter 7

The Implementation of HarAdmin

Hyper–G maintains user accounts and user groups in the server as objects. It is essentially the same mechanism as for storing document meta information in Hyper–G. This makes it possible to insert, manipulate and delete users and user groups.

7.1 The Hyper–G User Object

The user object supports the attributes:

- `UName`: The account name of the user.
- `Group`: A user is a member of at least one group.
- `Passwd`: The password is in encrypted form. There can be more than one `Passwd` entry for a user.
- `Host`: The form `user@host` is used for automatic identification.
- `Home`: Name of the user's personal home collection.
- `Descr`: Short description of user, e.g. full name.
- `Account`: An amount of virtual money.

Not all attributes must be present for a user, only `UName` and `Passwd` are necessary for a user object (for automatic identification `Host` is also needed).

7.2 The Hyper–G Group Object

The group object supports the attributes:

- `UGroup`: The Hyper–G group name

- `Group`: A parent group of the user group. There can be more than one `Group` entry since a group can inherited from more than one parent group.
- `Descr`: Short description of the group, for example the purpose of the group

Not all attributes are necessary for a group object, only `UGroup`.

7.3 The Hyper–G Client/Server Protocol

The material in this section has been adapted from the technical report "Hyper-G Client/Server Protocol (HG-CSP): Version 7.05" written by Gerald Pani and Frank Kappe. As mentioned at the beginning of this chapter, HarAdmin is based on the Hyper–G Client/Server Protocol (HG–CSP). The Hyper–G Client/Server Protocol has the following features:

- it is based on TCP/IP
- it is a connection-oriented protocol, stays open for duration of a session
- it is asynchronous
- and (almost) stateless.

The client sends messages to the server which initiate server transactions, and the server returns messages with the results of transactions and/or error codes. The client opens the connection by opening a TCP connection to the server's port (generally port 418). If any error occurs, HarAdmin displays the error in a message box.

The frequently used HG–CSP message is

◇ GetObject

```
boolean GetObject(const int ObjectID,ObjectRecord&)
```

HarAdmin uses this message to retrieve the full object record. Send the `ObjectID` and if no error occurs the `ObjectRecord` contains a list of attributes of this object.

Possible errors:

RSERV_NRESP Remote server not responding

NOTFOUND The object could not be found

NOACCESS No read permission for the object

Further HG–CSP message used by HarAdmin are:

- GetObjByQuery
- ObjectByIdQuery
- Identify
- ChangeObject
- GetAndLock
- UnLock
- InsertObject
- DeleteObject

A detailed description of the above messages is given in Appendix B.

7.4 The C++ Class Structure of HarAdmin

HarAdmin is written in C++. The three main classes are `Haradmin`, `User`, and `Group`, as shown in Figure 7.1.

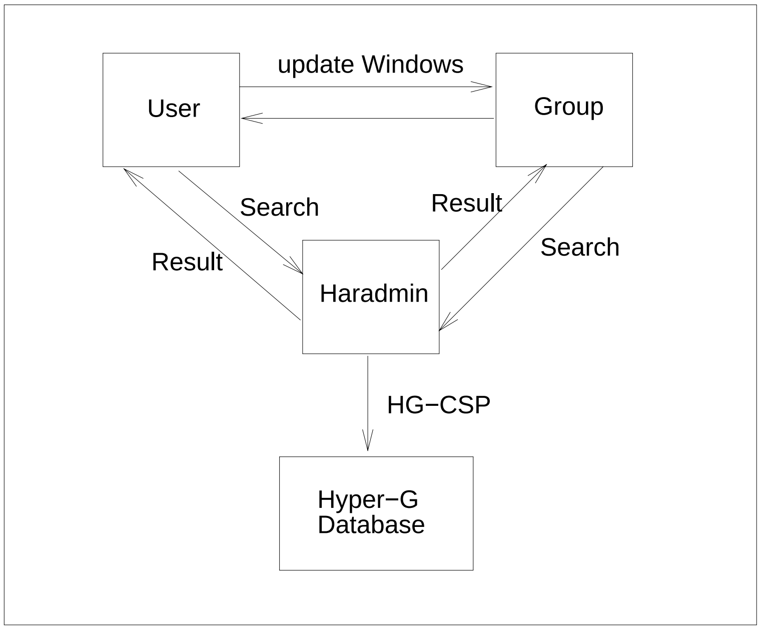


Figure 7.1: The Three Main Classes of HarAdmin

These three classes communicate with each other to send or retrieve messages (including error messages), to send or retrieve search requests, and to retrieve a list of matching objects or a list of attributes of a certain group or user object.

The classes `User` and `Group` manage the user interfaces and class `Haradmin` uses the Hyper–G Client/Server Protocol (HG–CSP) to communicate with the Hyper–G server. To communicate with the Hyper–G database an instance of the class `HgStub` is used.

7.4.1 Class Haradmin

The class `Haradmin` uses the previously discussed HG-CSP messages for maintaining user accounts and user groups.

```
class Haradmin
{
  Haradmin();  // opens the connection to the Hyper-G Server

  int getGroups(RString& filter, RStringArray& groups);
  int getParSubDes(RString& filter, RStringArray& dpa, RStringArray& ipa,
                 RStringArray& dsub, RStringArray& isub, RString& des);
  int getLockedParentsDess(RString& filter, RStringArray& parents,
                            RStringArray& dess);
  int insEditGroup(RString& groupname, RStringList pgrps, RStringList ds);
  int insNewGroup(RString& groupname, RStringArray pgrps, RStringArray ds);
  int remGroup(RString groupname);
  void remSubGroups(RStringArray& subgrps, RString& groupname);
  void deleteSubGroups(HgStub*, RStringArray& subgrps, RString groupname);
  int getUsers(RString& ufilter, RString gfilter, RStringArray& duser,
             RStringArray& iuser);
  int getHPGDHA(RString& filter, RStringArray& hs, RStringArray& pws,
               RStringArray& grps, RStringArray& dgrps, RString& des,
               RString& h, RString& a);
  int getLockedHPGDHA(RString& filter, RStringArray& hs, RStringArray& pws,
         RStringArray& gs,RStringArray& ds, RString& h, RString& a);
  int insNewUser(RString& un, RStringArray pws, RStringArray hs,
         RStringArray gs,RStringArray ds, RString h, RString a);
  int insEditUser(RString& un, RStringList pws, RStringList hs,
         RStringList gs, RStringList ds, RString h, RString a);
  int remUser(RStringArray& users, RString& error);
  void getSubGroupsAndUsers(RString groupfilter, RStringArray& dgrps,
         RStringArray& igrps, RStringArray& du, RStringArray& iu);
  ...
}
```

◊ getGroups

```
int getGroups(RString& filter, RStringArray& groups)
```

This command is used to retrieve all groups with selection string `filter`. First the HG–CSP message `GetObjByQuery` is used to get all matching ObjectIds (searches on indexed attributes, here UGroup). Then the HG–CSP message `ObjectByIdQuery` is used to get all object records for the previous retrieved ObjectIds. The names of the matching group objects are stored in `groups` and sent back to the `Group` class for displaying in the Group Window. If an error occurs, the error message is stored in `filter` and sent back to the `Group` class to display the error.

Possible error codes are:

1: Database busy

◇ getParSubDes

`int getParSubDes(RString& filter, RStringArray& dpa, RStringArray& ipa, RStringArray& dsub, RStringArray& isub, RString& des)`

This method is called to show all attributes of a certain group in the Group Window. It sends back all direct and indirect parent groups (`dpa, ipa`), all direct and indirect subgroups (`dsub, isub`), and the (last) description of the group with name `filter`. It first gets the object Id with the HG–CSP message `GetObjByQuery` and then uses the message `GetObject` to retrieve the full object record. After all attributes are received, the corresponding RStringArrays are filled and sent back to the `Group` class. To get indirect subgroups and indirect parent groups, first all direct and then recursively all indirect ones are searched. Remember Hyper–G uses a group hierarchy, which is an acyclic directed graph. If no error occurs all attributes are displayed in the Group Window.

Possible error codes are:

1: Database busy

2: GetObject failed

◇ getLockedParentsDess

`int getLockedParentsDess(RString& filter, RStringArray& parents, RStringArray& dess)`

This command is used to get all attributes of a group for editing. The procedure works in the same way as `GetParSubDes` but sends back only direct parent groups and an array of all found descriptions. The group must be first locked (HG–CSP: `GetAndLock`) to avoid that other persons can edit the group at the same time. The object is locked until the method `insEditGroup` is called.

Possible error codes are:

1: Database busy

2: Group is already locked

◊ insNewGroup

`int insNewGroup(RString& groupname, RStringArray pgrps, RStringArray ds)`

It is called if the `Commit` button in the New Group Window is clicked. It checks if the parent groups exist. If not, the insert process is stopped and an error message is sent back to the `Group` class. The `Group` class displays an error box and sets the focus to the line, which contains the wrong value. If no error exists the new group is inserted (HG–CSP: `InsertObject`) and "no error" is sent back to the `Group` class.

Possible error codes are:

1: Database busy

2: Parent Group does not exist

3: InsertObject failed

◊ insEditGroup

`int insEditGroup(RString& groupname, RStringList pgrps, RStringList ds)`

This command is called if the `Commit` button in the Edit Group Window was clicked. It checks if the parent groups really exist. If not, the insert process is stopped and an error message is sent back to the `Group` class. The `Group` class displays an error box and sets the focus to the line, which contains the wrong value. For each change it uses the HG–CSP command `ChangeObject` and after all changes are made, it sends back "no error" to the `Group` class. The differences between `insNewGroup` and this message are: it gets a list of parent groups instead of an array, and it has to unlock the group object after changing it (HG–CSP: `UnLock`).

Possible error codes:

1: Database busy

2: ChangeObject failed

◊ remGroup

`void remGroup (RString groupname)`

It is used to remove one group from the Hyper–G database. If the object is not locked, no subgroups, and no members exist the group is deleted with the HG–CSP message `deleteObject` and "no error" is sent back to the Group Window to be updated (see `groupWindowUpdate`). If members or subgroups exist the error code "3" is returned to the `Group` class.

Possible error codes are:

1: Database busy

2: Object is already locked

3: DeleteObject failed (subgroups or members exist)

◊ remSubGroups

`void remSubGroups(RStringArray& subgrps, RString& groupname)`

It is called after the `Ok` button in the Delete Dialog is pressed. `remSubGroups` first recursively unlinks all subgroups from their parent groups, if they should be deleted (HG–CSP: `ChangeObject`). Then it deletes all selected subgroups and the group selected in the column Group(s) in the Group Window (HG–CSP: `deleteObject`).

◊ getSubGroupsAndUsers

```
void getSubGroupsAndUsers(RString groupfilter, RStringArray& dgrps,
RStringArray& igrps, RStringArray& du, RStringArray& iu)
```

This command returns all users (direct and indirect) and all sub groups (direct and indirect) for a certain group `groupfilter`. To find indirect users and sub groups, the group hierarchy is searched recursively.

The functions for maintaining users have the same mechanism as the ones previously discussed for groups.

◊ insNewUser

Possible error codes are:

1: Database busy

2: InsertObject failed

30: Specify account (user@host)

31: Host doesn't exist

32: Host already exists for user

4: Group doesn't exist

◊ insEditUser

Possible error codes are:

1: Database busy

2: ChangeObject failed

30: Specify account (user@host)

31: Host doesn't exist

32: Host already exists for user

4: Group doesn't exist

7.4.2 Class User

The class `User` manages all activities of the User Window, the New User Window, and the Edit User Window. It sends requests to classes `Haradmin` and `Group`.

```
class User
{
public:
  User(Haradmin*);    // only initialize variables
  void userMenu();
  void removeUser();
  void newUserWindow();
  void editUserWindow();
  void FiltersAccept(FieldEditor31*, char);
  void deleteUsersFromGroup(RStringArray dusers,
                 RStringArray iusers, RString group);
private:
  void userWindowUpdate();
  void showAll();
  void showGroup();
  void insNewUser();
  void insEditUser();

  ....
}
```

◇ userMenu

`void userMenu()`

It is used to build the User Window. If the User Window already exists the User Window is only mapped again. At the beginning the user filter is set to the identified user and the group filter to "*". These filters are used to search the database for existing users. (`getUsers`). To send the filters to the class `Haradmin` the method `FiltersAccept` is used, which builds the selection string, sends it to `Haradmin`, and after receiving the matching objects, updates the User Window.

◊ userMenuUpdate

`void userWindowUpdate()`

It is activated when the user double–clicks on a certain user in the column User(s) to display all attributes of the selected user.

◊ newUserWindow

`void newUserWindow()`

This command builds and initializes the lines in the New User Window.

◊ editUserWindow

`void editUserWindow()`

This command builds and initializes the lines in the Edit User Window.

◊ showAll

`void showAll()`

It is called when the All button in the User Window is pressed. It changes the user and group filters to "*" and calls the method `FiltersAccept` to update the User Window.

◊ showGroup

`void showGroup()`

It is activated when the Show Group button in the User Window is pressed or double clicked on a certain group in the column Group(s) in the User Window. It checks if a group is selected and then activates the method `showGroupDetails` of the class `Group` to display all attributes of the selected group.

◊ insNewUser

`void insNewUser()`

It is called when the Commit button in the New User Window is pressed. It stores all given attributes in arrays and sends the arrays and the user name to the class `Haradmin` for insertion. (`insNewUser`). If an error occurs, the method displays the error in a message box and sets the focus to the line, which contains the wrong value. If no error occurs it calls the method `userWindowUpdate` to display the new user and the user's attributes in the User Window.

◇ deleteUsersFromGroup

```
void deleteUsersFromGroup()
```

This command is called from the class `Group` when a group has members and therefore cannot be deleted. It is used to update the UserWindow, displays all direct and indirect users, highlights all direct users, and changes the user filter to "*" and the group filter to `group`.

7.4.3 Class Group

The class `Group` manages all activities of the Group Window, the New Group Window, and the Edit Group Window. It sends requests to classes `Haradmin` and `User`.

```
class Group
{
public:
  Group(Haradmin*, User*);     // only initializes variables
  void groupMenu();
  void usersForGroup();
private:
  void showSubGroup();
  void showParentGroup();
  void newGroupWindow();
  void editGroupWindow();
  void insNewGroup();
  void insEditGroup();
  void removeGroup();
  void removeSubGroups();
  void groupWindowUpdate();
  void ggroupFilterAccept(FieldEditor31*, char);
  ....
}
```

◇ groupMenu

```
void groupMenu()
```

It is used to build the Group Window. If the Group Window already exists the Group Window is only mapped again. At the beginning the group filter is set to "*". These filters are used to search the database for existing groups. (`getGroups`). To send the filters to the class `Haradmin` the method `ggroupFilterAccept` is used, which builds the selection string, sends it to `Haradmin`, and after receiving the matching objects, updates the Group Window.

◇ groupWindowUpdate

`void groupWindowUpdate()`

It is activated when the user double–clicks on a certain group in the column Group(s) to display all attributes of the selected group.

◇ showParentGroup, showSubGroup

`void showParentGroup()` and `void showSubGroup()`

They are called when the user double–clicks in the columns Parent Group(s) or Sub Group(s) to display all attributes of the selected group.

◇ newGroupWindow

`void newGroupWindow()`

It builds and initialize the lines in the New Group Window.

◇ editGroupWindow

`void editGroupWindow()`

It builds and initialize the lines in the Edit Group Window.

◇ usersForGroup

`void usersForGroup()`

Is activated when the Show Users button in the Group Window is pressed. It checks if a group is selected and then activates the method `usersForGroup` of the class `User` to display all users of the selected group in the User Window.

◇ insNewGroup

`void insNewGroup()`

It is called when the Commit button in the New Group Window is pressed. It stores all given attributes in arrays and sends the arrays and the group name to the class `Haradmin` for insertion. (`insNewGroup`). If an error occurs, the method displays the error in a message box and sets the focus to the line, which contains the error. If no error occurs it calls the method `groupWindowUpdate` to display the new group and the group's attributes in the Group Window.

◊ removeGroup

`void removeGroup()`

This command is used to delete one group, it checks if a group is selected and calls the method `remGroup` of the class `Haradmin`. If an error occurs eg. if the group has sub groups and/or members the methods `getSubGroupsAndUsers` (of class Haradmin), then `deleteUsersFromGroup`, and `removeSubGroups` are called.

Chapter 8

Future Work

HarAdmin has extended the functionality of *hgadmin* (the VT100 terminal–based administration tool) with graphical components to make the management of user accounts and user groups easier, but it is not finished, yet. There can be done much more to help the system administrator and other users to manage their servers.

Ways in which the current version of HarAdmin could be extended include:

- In this version of HarAdmin, you cannot change the name of the user or group easily. You must delete the user or group object and create it new again with the desired name. The implementation of changing the names with HarAdmin is too difficult, because of the dependences of certain attributes of the objects. For example: if you would change a group name, the name also would have to be changed in all subgroups, parent groups, direct users, and access right fields.

- Deleting a group and direct members can also be extended. If you do not want to delete some direct users you have to change the group object of each user using the Edit User Window. Direct users are not unlinked automatically like direct subgroups. This is deliberate so as to make sure that no users exist who do not belong to any group.

- It is maybe confusing that you can create user accounts and user groups, but only can create a link to the home collection. You have to create a home collection by using Hyper–G clients such as *Harmony*, *Amadeus* or *hgtv*.

- It can be tedious to create multiple users, especially when you already have a list of users in a text file. You do not have an opportunity to import such users (like you have it in the Netscape Commerce Server).

- In this version only the system administrator and users of group system can create, delete, and manipulate other user accounts and user groups. It would be useful to be able to delegate certain functions to "subgroup administrators".

- If an organization has a complicated group hierarchy it might be useful to see the complete hierarchy, like the *local map* facility of *Harmony*.

HarAdmin is nevertheless a great step forward in maintaining user accounts and user groups under Hyper–G.

Chapter 9

Concluding Remarks

Simple and efficient access to information is the key to development. Twenty years ago a great step was made: The *Internet*, the network of networks, was born. Because of the huge amount of information available on the Internet, the user has difficulty gaining an overview and finding desired information. Many different systems have been built to help the user to overcome these problems of disorientation: *Archie*, a directory server for locating files by name, *WAIS*, a search engine for full text searching, *Gopher*, the first system to provide browsing and full text searches, and *WWW*, the first generation hypermedia information system. But they all together do not have the power to exploit the large information and communication resources available on the Internet.

Hyper–G, the first second generation hypermedia system, was built to overcome some of the problems of distributed hypermedia systems. It provides search, navigational, and orientational facilities, structures its information in so called collections, and builds a hierarchical structure called collection hierarchy. It differs in many ways from previous systems: it stores links in a separate database, it supports links not only in text documents, also MPEG–video clips, audio, and 3D–scenes. *Harmony*, the Unix client for Hyper–G, has structuring and retrieval facilities to provide navigational and orientational facilities and information feedback about the location of information. One of the most important differences is that Hyper–G supports anonymous and identified access, user accounts, and a hierarchical scheme of user groups to grant or deny access rights to certain parts of the Hyper–G information space.

HarAdmin, the graphical administration tool for Hyper–G servers, was discussed in detail in this thesis. Chapter 6 described in which ways user accounts and user groups can be inserted, manipulated, and deleted from the Hyper–G database. Hyper–G supports structures of groups and subgroups, which form an acyclic directed graph. Therefore deleting a group is complicated, and certain preconditions must be fulfilled to delete a group. HarAdmin has taken the place of the now more or less obsolete *hgadmin* administration tool, which is terminal–based and considerably less appealing to use.

HarAdmin is based on the Hyper–G Client/Server Protocol. The Hyper–G Client/Server protocol messages used by HarAdmin and the implementation of it are described in detail in this thesis for extending HarAdmin in the future. HarAdmin is now the definitive administration tool for maintaining user accounts and user groups under Hyper–G.

Appendix A

HarAdmin X–Defaults

Default Hyper–G host and port:

```
Harmony.Haradmin.hghost:  hgiicm.tu-graz.ac.at
Harmony.Haradmin.hgport:  418
```

Geometry options:

```
Harmony.Haradmin.Main.geometry:  300x115+10+5
Harmony.Haradmin.Identify.geometry: +20+30
Harmony.Haradmin.User.geometry:  680x375+10+160
Harmony.Haradmin.NewUser.geometry: +720+180
Harmony.Haradmin.EditUser.geometry: +720+160
Harmony.Haradmin.UserOptions.geometry: +720+160
Harmony.Haradmin.Group.geometry:  680x360+10+580
Harmony.Haradmin.NewGroup.geometry: +720+600
Harmony.Haradmin.EditGroup.geometry: +720+580
Harmony.Haradmin.GroupOptions.geometry: +720+600
Harmony.Haradmin.DeleteDialog.geometry: +720+580
```

Options to enable or disable columns and lines in the User Window:

```
Harmony.Haradmin.UserOptions.showhosts: on
Harmony.Haradmin.UserOptions.showpasswds: on
Harmony.Haradmin.UserOptions.showgroups: on
Harmony.Haradmin.UserOptions.showdescription: on
Harmony.Haradmin.UserOptions.showhome: on
Harmony.Haradmin.UserOptions.showaccount: on
```

To show host by domain name (domain) or ip address (ip) or
alias name (alias).

```
Harmony.Haradmin.UserOptions.showhostby: domain
```

Options to enable or disable columns and lines in the Group Window:

```
Harmony.Haradmin.GroupOptions.showparents: on
Harmony.Haradmin.GroupOptions.showsubgroups: on
Harmony.Haradmin.GroupOptions.showdescription: on
```

Appendix B

HG–CSP Commands used by HarAdmin

The material in this section has been adapted from the technical report "Hyper-G Client/Server Protocol (HG-CSP): Version 7.05" written by Gerald Pani and Frank Kappe.

This chapter contains a short description of messages used by HarAdmin to maintain users and user groups (implemented in files `haradmin.[Ch]`). The Hyper–G Client/Server Protocol messages are implemented in the C++ class `HgStub`.

◇ GetObject

`boolean GetObject(const int ObjectID,ObjectRecord&)`

HarAdmin uses this message to retrieve the full object record. Send the `ObjectID` and if no error occurs the `ObjectRecord` contains a list of attributes of this object.

Possible errors:

RSERV_NRESP Remote server not responding

NOTFOUND The object could not be found

NOACCESS No read permission for the object

◇ GetObjbByQuery

`int GetObjByQuery(const RString& indexquery,ObjIdArray&)`

This command is used for searching on *indexed* attributes of Hyper–G objects. The search is performed on a set of attributes over the whole database of the connected Hyper–G server. Use `GetObjectByIdQuery` command to match a set of

objects returned by this command against other, non-indexed, attributes. If no error occurs, it returns the IDs of the matching objects in `ObjIdArray`. If nothing is found the count of `ObjIdArray` is 0.
Possible errors:

CMD_SYNTAX `indexquery` is invalid

Q_OVERFLOW too many intermediate results

Indexed Hyper–G attributes important for HarAdmin:

Attribute	Description
Group	Parent group
Host	is used for automatic identification
UName	account name of the user
UGroup	Hyper–G user group

The syntax of `indexQuery` is as follows:

```
<EXPR> ::= "(" <EXPR> ")" |
           <EXPR> "||" <EXPR> |  /* OR */
           <EXPR> "&&" <EXPR> |  /* AND */
           <EXPR> "&!" <EXPR> |  /* AND NOT */
           <INDEXFIELD> "=" <VALUES> |
           <INDEXFIELD> ">" <SIMPLEVALUE> "<" <SIMPLEVALUE>  /* BETWEEN */

<INDEXFIELD> ::= "UName" | "UGroup" | "Host" | "Group" |

<VALUES> ::= <VALUE> |
             <VALUE> <VALUES>  /* value AND values must be true */

<VALUE> ::= <SIMPLEVALUE>      /* ordinary word */
            <SIMPLEVALUE>"*"   /* all words with prefix simplevalue */
            <SIMPLEVALUE>"^"   /* all words >= simplevalue */
```

◊ ObjectByIdQuery

```
int ObjectByIdQuery(ObjectIdArray&, const RString query,
ObjectField&)
```

Like `GetObject`, this command is used to retrieve the object records (with the object's attributes) from object IDs of `ObjectIdArray`. This command is more powerful in that it lets you retrieve a number of object records with one command. In addition, a query may be performed on the object records to narrow down the set of objects retrieved. The query is not limited to the indexed attributes, and is more powerful than the index query of the `GetObjByQuery` commands, but slower as it is performed directly on the attribute values.

Send the ObjectIDs returned from `GetObjectByQuery` and retrieve the matching object records `ObjectField`.

The syntax of `query` is as follows:

```
<EXPR> ::= "(" <EXPR> ")" |
           "!" <EXPR> |          /* NOT */
           <EXPR> "||" <EXPR> |  /* OR */
           <EXPR> "&&" <EXPR> |  /* AND */
           <ATTRIBUTE> <OPERATOR> <VALUE>

<ATTRIBUTE> ::= /* any attribute name (Author, DocumentType ...) */

<OPERATOR> ::= "=" |    /* equal */
               "<" |    /* LESS THAN (STRING COMPARE) */
               ">" |    /* greater than (string compare) */
               "~"      /* regular expression matching */
```

It is here (and at the simple `GetObject` command) where Hyper-G's read access rights are checked. This is because access rights to objects are contained within the *Rights* attribute, so that the object record has to be fetched from the low-level database in order to determine whether a user is allowed to access it. This is why it is normal that this call may return fewer object records than have been requested. The commands that return object IDs always return all object IDs, even if the user is not supposed to access them.

◇ Identify

```
void Identify(const RString& name, const RString& passwd,
boolean encrypt)
```

This command is used to identify the user to the Hyper-G database. This command must be used before any of the commands that modify the database can be used (otherwise, the NOACCESS error will occur when such a command is used). By default, the user is identified as "anonymous" or automatically identified if configured.

The `name` specifies the Hyper-G user name, while `passwd` is the password. If `encrypt` is set to true, the client supplies `passwd` in encrypted form (according to modified DES, same as UNIX passwords), otherwise it is passed in clear text. The server will return the object id of the users UserObject and the name of the user itself (e.g., "0x0000000 anonymous", if the identification failed).

No errors can occur.

◊ ChangeObject

`boolean ChangeObject(ObjectID, const RString& command)`

This command allows to remove, add, or modify individual attributes of an object record. The object is specified by `ObjectID`; commands adhere to the following syntax:

```
<COMMAND> ::= <REMCMD> |
              <ADDCMD> |
              <REMCMD> "\" <ADDCMD>

<REMCMD>  ::= "rem " <ATTRIBUTE> "=" <VALUE>

<ADDCMD>  ::= "add " <ATTRIBUTE> "=" <VALUE>
```

Note that in order to delete or remove an attribute the object has to be locked (`GetAndLock` and its old value has to be supplied (some attributes are allowed more than once). A command like
`''rem Account='' + oldvalue + ''\nadd Account'' + newvalue`
allows to modify attributes in one operation, here changes the value of the Account Attribute..

Errors: If *error* is not zero, an error has occurred, e.g.:

NOACCESS No permission to modify object eg. the object is already locked.

CMDSYNTAX Error in command syntax.

CHANGEBASEFLD Change of base attribute (i.e., attributes that cannot be changed, like *ObjectID* or *Type*).

NOTREMOVED Attribute not removed (e.g., because the value was wrong).

FLDEXISTS Attribute already exists and may exist only once.

Errors are listed in Appendix C.

◊ GetAndLock

`boolean GetAndLock(const ObjectID, Object&)`

Similar to `GetObject`: returns object record from object ID (provided that access permissions are OK). However, the object is also marked as “locked” in the database. That means that another user trying to use GetAndLock on the same object would fail, until the object has been unlocked again (with the `Unlock` command), or the connection of the user that locked the object is closed.

Clients must use this mechanism to implement write-locks, to ensure that no two users can modify the same object at the same time. Specifically, the `ChangeObject` commands have to be surrounded by `GetAndLock` and `UnLock`. Unlike `GetObject`, this command guarantees that the attributes returned are the current state of the object and cannot be modified by somebody else between operations.

Possible errors:

NOACCESS No permission to access object.

LOCKED Object is locked by another user; try again later.

See appendix C for a list of all possible error codes.

◇ UnLock

`boolean UnLock(const ObjectID)`

Removes the lock flag from object `ObjectID` after a previous `GetAndLock` operation on that object. The lock flag is also removed when the connection that locked the object is closed. Note that this command does not return a response.

No errors can occur.

◇ InsertObject

`boolean InsertObject(const RString& objRecord, const RString& params, ObjectID)`

Allows to insert an arbitrary object into the Hyper-G database. `objRecord` is the object record to insert, while `params` allows to specify parameters that are not part of the object record.

On successful return `ObjectID` which is the new object ID that has been assigned for this object.

See appendix C for a list of all possible error codes.

◇ DeleteObject

`boolean deleteObject(const ObjectID)`

This command deletes the object with ID `ObjectID`. Before you can delete a object, you must lock the object, see `GetAndLock()`.

Possible errors:

NOACCESS No permission to delete object.

NOTEMPTY Attempt to delete non-empty collection, e.g you try to delete a group which contains members or subgroups.

See appendix C for a list of all possible error codes.

Appendix C

HG-CSP Error Codes

The material in this section has been adapted from the technical report "Hyper-G Client/Server Protocol (HG-CSP): Version 7.05" written by Gerald Pani and Frank Kappe.

An error code of 0 means that no error occurred.

The following table [Mau96] shows the link server error codes with their associated mnemonic names and a short (human-readable) description that could be output on a terminal, for example .

Code	Mnemonic	Description
1	NOACCESS	Access denied
2	NODOCS	No documents
3	NONAME	No collection name
4	NODOC	Object is not a document
5	NOOBJ	No object received
6	NOCOLLS	No collections received
7	DBSTUBNG	Connection to low-level database failed
8	NOTFOUND	Object not found
9	EXIST	Collection already exists
10	FATHERDEL	parent collection disappeared
11	FATHNOCOLL	parent collection not a collection
12	NOTEMPTY	Collection not empty
13	DESTNOCOLL	Destination not a collection
14	SRCEQDEST	Source equals destination
15	REQPEND	Request pending
16	TIMEOUT	Timeout
17	NAMENOTUNIQUE	Name not unique
18	WRITESTOPPED	Database now read-only; try again later
19	LOCKED	Object locked; try again later
20	CHANGEBASEFLD	Change of base-attribute
21	NOTREMOVED	Attribute not removed
22	FLDEXISTS	Attribute exists
23	CMDSYNTAX	Syntax error in command

24	NOLANGUAGE	No or unknown language specified
25	WRGTYPE	Wrong type in object
26	WRGVERSION	Client version too old
27	CONNECTION	No connection to other server
28	SYNC	Synchronization error
29	NOPATH	No path entry
30	WRGPATH	Wrong path entry
31	PASSWD	Wrong password (server-to-server server authentication)
32	LC_NO_MORE_USERS	No more users for license
33	LC_NO_MORE_DOCS	No more documents for this session and license
34	RSERV_NRESP	Remote server not responding
35	Q_OVERFLOW	Query overflow
36	USR_BREAK	Break by user
37	N_IMPL	Not implemented

Bibliography

[AAL$^+$92] Alberti B., Anklesaria F., Lindner P., McCahill M., and Torrey D.: "The Internet Gopher Protocol: A Distributed Document Search and Retrieval Protocol". Available at `ftp://boombox.micro.-umn.edu/pub/gopher/gopher_protocol`, March 1992.

[AK94] Andrews K. and Kappe F.: "Soaring Through Hyperspace: A Snapshot of Hyper-G and its Harmony Client". In Herzner W. and Kappe F. (editors), *Proc. of Eurographics Symposium on Multimedia/Hypermedia in Open Distributed Environments*, pages 181–191, Graz, Austria, June 1994. Springer.

[AKM95a] Andrews K., Kappe F., and Maurer H.: "The Hyper-G Network Information System". *Journal of Universal Computer Science*, 1(4):206–220, April 1995. Available at URL `http://-hyperg.iicm.tu-graz.ac.at/jucs`.

[AKM95b] Andrews K., Kappe F., and Maurer H.: "Serving Information to the Web with Hyper-G". *Computer Networks and ISDN Systems*, 27(6):919–926, April 1995. Proc. Third International World-Wide Web Conference, WWW'95, Darmstadt, Germany.

[AKMS95] Andrews K., Kappe F., Maurer H., and Schmaranz K.: "On Second Generation Network Hypermedia Systems". In *Proc. of ED-MEDIA '95*, pages 75–80, Graz, Austria, June 1995. AACE.

[BLCGP92] Berners-Lee T., Cailliau R., Groff J.-F., and Pollermann B.: "World-Wide Web: The Information Universe". *Electronic Networking: Research, Applications and Policy*, 2(1):52–58, Spring 1992.

[BLCL$^+$94] Berners-Lee T., Cailliau R., Luotonen A., Nielsen H. F., and Secret A.: "The World-Wide Web". *Communications of the ACM*, 37(8):76–82, August 1994.

[Cai95] Cailliau R.: "About WWW". *Journal of Universal Computer Science*, 1(4):221–231, April 1995.

[DH95] Dalitz W. and Heyer G.: *Hyper-G: Das Internet-Informationssystem der 2. Generation.* dpunkt, October 1995. In German, see `http://www.dpunkt.de/`.

[DR94] December J. and Randall N.: *The World Wide Web Unleashed.* Sams Publishing, Indianapolis, USA, 1994. Available at `http://www.-rpt.edu/ decemj/works/wwwu.html`.

[ED92] Emtage A. and Deutsch P.: "archie – An Electronic Directory Service for the Internet". In *USENIX Winter 1992 Technical Conference Proceedings*, pages 93–110, San Francisco, CA, 1992. USENIX Association.

[Eyl95] Eyl M.: "The Harmony Information Landscape Interactive, Three-Dimensional Navigation Through an Information Space". Master's thesis, University of Technology, October 1995. Available at `ftp:-//ftp.iicm.tu-graz.ac.at/pub/Hyper-G/doc/eyl.ps`.

[Flo95] Flohr U.: "Hyper-G Organizes the Web". *BYTE*, pages 59–64, November 1995.

[Kap93] Kappe F.: "Hyper-G: A Distributed Hypermedia System". In Leiner B. (editor), *Proc. INET '93, San Francisco, California*, DCC-1–DCC-9, August 1993. Internet Society.

[KP94] Kappe F. and Pani G.: "Hyper-G Client/Server Protocol (HG-CSP): Version 7.05". Technical report, IICM, Graz University of Technology, December 1994. Available at `ftp://ftp.iicm.tu-graz.ac.at /pub/Hyper-G/papers/HG-CSP-7.05.ps.gz`.

[KPS93] Kappe F., Pani G., and Schnabel F.: "The Architecture of a Massively Distributed Hypermedia System". *Internet Research: Electronic Networking Applications and Policy*, 3(1):10–24, Spring 1993.

[Kro94] Krol E.: *The Whole Internet: User's Guide and Catalog.* O'Reilly & Associates, second edition, April 1994.

[LCV87] Linton M. A., Calder P. R., and Vlissides J. M.: "The Design and Implementation of InterViews". In *Proc. USENIX C++ Workshop, Santa Fe, New Mexico*, November 1987.

[LPJ$^+$94] Liu C., Peek J., Jones R., Buus B., and Nye A.: *Managing Internet Information Services: World Wide Web, Gopher, FTP and more.* O'Reilly & Associates, Sebastopol, USA, December 1994.

[LVC89] Linton M. A., Vlissides J. M., and Calder P. R.: "Composing User Interfaces with InterViews". *IEEE Computer*, 22(2):8–22, February 1989.

[MA95] McCahill M. P. and Anklesaria F. X.: "Evolution of Internet Gopher". *Journal of Universal Computer Science*, 1(4):235–246, April 1995.

[Mar95] Marshall B.: "Integration of Digital Video into Distributed Hypermedia Systems". Master's thesis, University of Technology, March 1995. Available at `ftp://ftp.iicm.tu-graz.ac.at/pub/Hyper-G/-doc/marschall.ps`.

[Mau92] Maurer H.: "Why Hypermedia Systems are Important". In *Proc. ICCAL '92*, pages 1–15. LNCS 602, Springer Pub. Co., 1992.

[Mau96] Maurer H.: *Hyper-G: The Second Generation Web Solution.* Addison-Wesley, 1996.

[Mey86] Meyrowitz N. K.: "Intermedia: The Architecture and Construction of an Object Oriented Hypertext/Hypermedia System and Applications Framework". In *Proc. OOPSLA 86*, pages 186–201, Portland, OR, September/October 1986.

[Nel87] Nelson T. H.: *Literary Machines.* The Distributors, South Bend, IN 46618, USA, 87.1 edition, 1987.

[Sch95] Schmid A.: "A Mail Archive Server on Top of Hyper-G". Master's thesis, University of Technology, November 1995. Available at `ftp:-//ftp.iicm.tu-graz.ac.at/pub/Hyper-G/doc/schmid.ps`.

[Ste91] Stein R. M.: "Browsing Through Terabytes – Wide-area Information Servers Open a New Frontier in Personal and Corporate Information Services". *BYTE*, 16(5):157–164, May 1991.

[Str91] Stroustrup B.: *The C++ Programming Language.* Addison-Wesley, Reading, Massachusetts, second edition, 1991.

[The95] *The Internet Unleashed.* Sams.net Publishing, 2nd edition, May 1995.