Diplomarbeit in Telematik, TU Graz

Institut für Informationsverarbeitung und Computergestützte neue Medien

Object-Oriented Implementation of a Multiprotocol Hyper-G Client for Ms-Windows

Thomas Dietinger

Gutachter: O.Univ.-Prof. Dr.phil. Dr.h.c. Hermann Maurer

Betreuer: Dipl.-Ing. Dr.techn. Frank Kappe

Graz, im Juli 1995

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

Die Arbeit ist in englischer Sprache verfaßt.

Danksagung

Ich möchte mich an dieser Stelle bei allen Mitarbeitern des IICM und IHM, die mich bei der Durchführung dieser Arbeit unterstützt haben und stets bemüht haben, meine Fragen zu beantworten, herzlich bedanken.

Spezieller Dank gebührt auch meinen Eltern und meiner Verlobten für ihr Verständnis und ihre aufopfernde Hilfe.

Abstract

The subject of this thesis is to describe the implementation and the usage of an MS Windows ™ based Hyper-G Client called Amadeus that not only features accessing information but also makes it possible to insert data into the system.

Wide spread acceptance of Amadeus pointed out high user friendliness and the availability of a powerful set of functions.

To bring the whole thematic in a better context chapter 2 will describe the technical prerequisites and the history of the Internet. Chapter 3 will explain basic Internet services like Telnet, FTP, E-Mail and Usenet-News. Traditional first generation information systems and their advantages and disadvantages will be weighed in chapter 4.

In chapter 5 the concept of Hyper-G will be discussed in detail specifying its underlying data model, the Client-Server architecture and other design goals.

Finally, chapter 6 will give a profound description of Amadeus explaining its implementation, all functions and its usage.

Table of Contents

1. Introduction

Nowadays everybody is talking about the Internet and wants to utilize it. Unfortunately expectations are often set to high due to exaggerated reports in the media about video on demand, Virtual Reality etc.

But indeed, the Information Highway is a very large library where terabytes of data are stored and can help us not only in science but also in everyday life. The problem is just how to find the information we need and how to access it. First generation information systems like WWW, Gopher and WAIS tried to face the problem and developed different strategies. But reality showed that these technologies have not been sufficient to organize the chaos of data and services in the Internet.

That's why Hyper-G, a second generation information system, has been developed at Graz University of Technology. The main goal of Hyper-G is to provide information stored in this system in a structured way based on an object oriented database. With a Hyper-G Client the data can also be retrieved very easily using powerful search capabilities and viewed with a user friendly front end.

The subject of this thesis is to describe the implementation and the usage of an MS Windows ™ based Hyper-G Client called Amadeus that not only features accessing information but also makes it possible to insert data into the system.

Wide spread acceptance of Amadeus pointed out high user friendliness and the availability of a powerful set of functions.

To bring the whole thematic in a better context chapter 2 will describe the technical prerequisites and the history of the Internet. Chapter 3 will explain basic Internet services like Telnet, FTP, E-Mail and Usenet-News. Traditional first generation information systems and their advantages and disadvantages will be weighed in chapter 4.

In chapter 5 the concept of Hyper-G will be discussed in detail specifying its underlying data model, the Client-Server architecture and other design goals.

Finally, chapter 6 will give a profound description of Amadeus explaining its implementation all functions and its usage.

2. Internet

At the moment there are perhaps 50 million people using the Internet on a regular basis, but the current growth rate is about 15% per month! [Fenn94] This could well continue until almost all of those in the ‘developed’ world are connected. If this increase would continue, in theory everybody within the human population could participate short after the millennium. In reality of course the growth will decrease because of not fully developed infrastructure in some countries and limited financial resources.

Nevertheless I am very positive that later in the next century using the Internet or one of its successors will be as common as making a phone call or watching TV today and will have a tremendous impact on everyday life.

2.1 History

The Internet was born about 20 year ago, out of an effort to connect together a U.S. Department of Defense network called the ARPAnet. It was an experimental network designed to support military research - in particular, research about how to build networks that could withstand partial breakdowns and still function.

Communication always occurs between a source and a destination computer. The network itself is assumed to be unreliable. Any portion of the network could disappear at any moment caused by natural disasters or by war. It only guaranteed that the data might be transmitted somehow, somewhen - the communicating computers were given the responsibility for ensuring the correct transmission of the data. To accomplish this several communication protocols had been invented. The two most important of them TCP and IP will be described later.

In about the same time the Internet was coming into being, local area networks (LANs) were developed. But due to the very expensive hardware LAN technology matured quietly and slowly until the early eighties when desktop workstations became available and local networking exploded. Most of these workstations came with Berkeley UNIX, which included IP networking software. This created a new demand: rather than connecting to a single large timesharing computer per site, organizations wanted to connect their entire local network to the ARPAnet.

At the same time other companies started building private networks using the same communication protocols. It became obvious that if these networks could talk with each others, users on one network could communicate with those on another, everyone would benefit - Internet the network of networks has been formed!

Nowadays not only universities and research centers, even most colleges, high schools and bigger corporations are connected to the Internet. It will not be too long, until primary schools, smaller companies and more private individuals will participate.

2.2 TCP/IP and the fundamentals of the Internet

TCP/IP is often used as a synonym for the Internet although that is not completely correct because they are only two important protocol types out of three (the third one is UDP) the Internet and all of its services are based on.

The Internet is based on a model ("*catenet model*") that assumes that there are a large number of independent networks connected together. The user should be able to access computers or other resources on any of these networks. *Datagrams* (parts of information) will often pass through several different networks before getting to their final destination. The *routing* needed to accomplish this should be completely invisible to the user. All the user needs to know is the *Internet address* identifying the partner system. It is a 32-bit number normally written as 4 decimal numbers with a dot in-between (e.g., 129.27.191.26). Of course this number has to be

unique within the Internet. To simplify the correctness of these numbers a hierarchical structure called '*classes*' has been developed: A network administrator who wants to connect new PCs to the Internet does not get a special Internet address for each of the PCs but an address range for the whole subnet. This range can vary depending on the size of the site and on the number of computers that have to be connected. We distinguish 5 different classes: (The 'xxx' are a number the network administrator receives for the subnet, the '*' are numbers (1-254, because 0 and 255 are reserved) the network administrator can assign to each PC)

- Class A network: xxx.*.*.*, (xxx between 1 and 126), for 254*254*254 = 16.387.064 PCs on the site
- Class B network: xxx.yyy.*.*, (xxx between 128 and 191), for 254*254 = 64.516 PCs
- Class C network: xxx.yyy.zzz.* , (xxx between 192 and 223), for 254 PCs
- Class D network: xxx.yyy.zzz.* , (xxx between 224 and 239), for 254 PCs (not available for general use)
- Class E network: xxx.yyy.zzz.* , (xxx between 240 and 255), for 254 PCs (not available for general use)

(Note: You can request numbers for a new subnet by sending electronic mail to *Hostmaster@INTERNIC.NET* , it is operated by the Department of Defense Network Network Information Center (DDN NIC) - see [WashEvan93] for a detailed description)

Of course you can also divide a class B network into 254 class C networks by yourself, but it will be up to you to ensure that these numbers are assigned correctly.

The information about the ip number and the physical place of the subnet is stored in quite complex machines called '*routers*' because these network junctions also try to route datagrams into the right direction. They are so complicated because there are many ways an information packet could go through a big network and it is not very easy to choose the best (fastest and most secure) one. The circumstances can also be very dynamic. If a huge datagram is splitted into several smaller packets (this is done by the Internet Protocol), each of them could be routed over a different path, because the surroundings (network traffic, breakdown of parts of the network etc.) could change so fast.

Internet numbers are difficult to remember by human beings, that's why each computer is also assigned an *Internet name*. Usually these names are also structured hierarchically:

<computername>.<department>.<organization>.<type of organization>.<country>

Example: hyperg.iicm.tu-graz.ac.at, where

- **'at'** means Austria
- **'ac'** means academic
- **'tu-graz'** means Graz, University of Technology
- **'iicm'** means institute for informatics and computer supported new media
- **'hyperg'** is the name of the computer at last

There exist also some variations, especially for smaller companies (where you do not have departments) or older sites (where the country field is missing). These fields are also called domains and subdomains - see [FreyAdam94] for a detailed description.

Again the relation between the Internet numbers and names has to be handled somehow. This is done by the (domain) name service (see [RFC 882] for details). If somebody wants to send some data over the Internet specifying the destination by name, the local name server checks whether the target belongs to the same domain. If this is the case, it has to find out the correct ip number using its lookup tables stored in the name server. If the destination belongs to a different network then it has to ask a name server that is higher in the hierarchy.

Let's say, we want to connect to 'ftp.hmu1.auckuni.ac.nz' that is an ftp server at the 'HyperMedia Unit' at the University of Auckland' in New Zealand. Then the name server of 'iicm' has to ask the name server of 'tu-graz', this one has to ask the name server of the academic network in Austria and that one has to ask one of the global name servers that know the ip addresses of other country specific name servers, in our case the address of a name server in New Zealand. Now the query continues until we find a name server that knows the exact address of our target.

Of course it is not always that complicated, because name servers have integrated caches, so that they can store information about already found name-number relations for a certain amount of time.

Now we know those name servers and routers help us in finding the partner for our communication, but we still do not know how it is going to be controlled and who assures that our data is transmitted correctly. That's up to *TCP*, the transmissions control protocol, and *IP*, the Internet protocol.

TCP is responsible for breaking up messages into datagrams, reassembling them at the other end, resending anything that gets lost, and putting things back together in the right order. IP is responsible for routing individual datagrams.

Suppose we have a single data stream (e.g. a file) which we are trying to send to some other computer:

.....

TCP breaks it up into manageable chunks.

....

Then TCP puts a header in front of each datagram. This header actually contains at least 20 bytes, but the most important ones are a source and destination '*port number*' and a '*sequence number*'. The port numbers are used to keep track of different conversations. Suppose 3 different people are transferring files. Your TCP might assign port number 1024, 1025 and 1026 to these transfers. When you are sending a datagram, this becomes the source port number, since you are the source of the datagram. Of course TCP at the other end has assigned a port number of its own for the conversation. Your TCP has to know the port number used by the other end as well. (It finds out when the connection starts.) It puts this in the destination port field. Of course if the other end sends a datagram to you, the source and destination port numbers will be reversed since it will be the source and you will be the destination.

Each datagram has a sequence number. This is used so that the other end can make sure that it gets the datagrams in the right order, and that it has not missed any. Finally, also of great importance is the checksum which is put at the end of the header and ensures the correctness of this datagram.

If we abbreviate the TCP header as "T", the whole file now looks like this:

T.... T.... T.... T.... T.... T.... T....

Note: The TCP header also contains some items that are not mentioned here mainly for managing the connection, look at [RFC793] for details.

TCP sends each of these datagrams to IP. Of course it has to tell IP the Internet address of the computer at the other end. Note that this is all IP is concerned about. It doesn't care about what is in the datagram, or even the TCP header. IP's job is simply to find a route for the datagram and get it to the other end. In order to allow gateways or other intermediate systems to forward the datagram, it adds its own header. The main things in this header are the source and destination Internet address, the protocol number, and another checksum. The protocol number tells IP at the other end to send the datagram to TCP. Although most IP traffic uses TCP,

there are also other protocols that can use IP, so you have to tell IP which protocol to send the datagram to. Note that IP has its own checksum, to be able to verify that the header didn't get damaged during transmission. If we represent the IP header by an "I", your file now looks like this:

IT.... IT.... IT.... IT.... IT.... IT.... IT....

Again the header contains some additional fields that have not been discussed. See at [RFC791] for a detailed description of the Internet Protocol.

At this point, the datagrams will be passed to the physical layer of your network. This could be your network card based on the Ethernet protocol or your Modem that uses SLIP or PPP. Again headers and checksums have to be added to the datagrams, but they are too specific and will not be described here. (See [RFC894] and [RFC826] for a description for sending IP over Ethernet and [RFC1055] for SLIP)

3. Internet Services

Internet Services simplify the usage of the network. Nobody would like to send single TCP packages over the net, but complete files (FTP), messages (EMail) or more sophisticated things. There are also many services that run in the background without notice from the user. The Domain Name Service, I mentioned before, is just one of them.

In this chapter I will describe four of the most common Internet services:

- *Telnet* for remotely logging into other computers on the net. It's used for a lot of public services, including libraries and other kinds of databases.
- *FTP* for file transfer between two computers on the net. It's most useful for retrieving files from public archives that are scattered around the Internet.
- *E-Mail* for sending electronic mail within seconds to every computer on the Internet
- *Usenet News* a public bulletin board for people who want to discuss with other Internet users.

3.1 Telnet

In former times people didn't have a personal computer or workstation but a simple terminal consisting of a keyboard and a screen capable of only displaying characters and no graphics. These terminals were connected to a mainframe over a serial line, a modem or over the network and were used to remotely control and utilize it. This was the only possibility to grant access to these expensive mainframes to a broader range of users, mainly scientists and students.

Using the Telnet service is quite similar: A user may log into another computer on the net by identifying with the users account name and password. Afterwards nearly every keystroke the user types into the keyboard is sent over the network to the destination computer. This computer in turn reacts to them and sends back some characters as the result that are visualized on the users screen.

The great advantage of this is that you don't need a special hardware to use this services, just a screen capable if displaying characters and a keyboard to enter your commands. That's why especially libraries and big databases that want to offer their data to a broad range of people often support the Telnet service. Of course the user interface is quite poor due to the lack of colorful graphics and mouse input (although your terminal might have these capabilities you won't be able to use them with the Telnet service).

3.2 FTP

Often you will find information on the Internet that you don't want to examine on a remote system. You want to have a copy, for example of a recent research paper or some free software, for yourself. Then the *FileTransferProtocol* is exactly what you need!

As the name implies, it's primary purpose is to exchange files between two computers over the network. It does not matter where the two computers are located, how they are connected, or even whether or not they use the same operating system. The only requirement is that one of the two is an ftp-*server* and the other an ftp-*client*.

The client can be as simple as a command line tool, where you type in commands like get (to fetch a file) and put (to upload a file) or it can have a sophisticated graphical user interface with mouse support and drag and drop facilities.

An important point to notice here is the fact that FTP is the first real *client/server protocol*. This means that not every single keystroke you type in will be

transmitted over the network, but instead the client will gather enough information to form a command and send it to the server (where it will be computed). Things like editing the command because you misspelled it or a different user interface will be processed completely by the client and won't increase the network load (versus Telnet, where every keystroke will be transmitted!).

3.3 E-Mail

Electronic Mail is one of the most important Internet services, that even unexperienced people might use and need. It's the modern way to communicate and will replace telephone and fax in some areas quite soon. It has not only the capability to send messages within second around the whole world, a user may also deliver files like images, sounds, spreadsheets etc. over E-Mail. Another advantage is that it's quite cheap, because many Internet providers include E-Mail in their cheapest package and you only have to pay for a local call (if you use a modem).

Unfortunately E-Mail has one big disadvantage: it is not very secure ! There is no real guarantee that your mail reaches its addressee nor that nobody else has read or changed it. That's why you should not order or buy things using E-Mail.

3.4 Usenet News

Whenever you have a question nobody you know can answer take a look at some of the newsgroups and post it there, maybe someone can help you then.

Usenet News is the Internet equivalent of a discussion group or a "*bulletin board system*" (BBS) like those on CompuServe or private dial-up facilities. To the user, network News organizes discussions under a set of headings called *Newsgroups*. Newsgroups are organized hierarchically, with the broadest grouping first in the name, followed by an arbitrary number of subgroupings. The name of its group is separated from its *parent* and its *subgroups* by a period (.) (like 'comp.os.ms-windows.win32.programmers', for programmers that code using the Win32 API of MS-Windows™).

Most of the newsgroups come as part of *Usenet*, a set of newsgroups generally considered to be of interest globally, and free. Usenet is a set of voluntary rules for passing and maintaining newsgroups. There are some well managed newsgroup categories (like 'comp', 'news', 'rec', 'sci', 'soc', 'talk' and 'misc'), where you can only create a new subgroup after some sort of democratic voting. On the other hand there are many *alternative* or private categories that everybody can declare without a rule, but which are not necessarily mirrored to other news servers.

Now everybody who has a *news-reader* (the client) and is permitted to connect to a *news-server* may follow numerous discussions or actively participate through posting questions or answering them. If you are in a certain group you also have the possibility to search for an article specifying the author or the subject. Unfortunately there is no possibility to search in fulltext or over several groups.

4. Information Systems

Now we know some of the basic Internet services and we see that they are quite difficult to use: Each of the services has different user interfaces and we have to remember lots of server names. To ease usage it would be great to have something like an Internet directory where we can browse or even better search for things we need from the net. It would also be very comfortable to have a common user interface to minimize learning costs.

This is exactly what *information systems* want to provide: A structured access to all kind of data (multimedia: text, image, sound, movie, ...) over different operating system platforms and different Internet services (including the basic ones, to gain most benefit of the net).

One important prerequisite for this is the implementation of *the client-server concept*. Here we have a server that grants access to its data using a protocol as powerful as possible. The client and thus the complete user interface to the server on the other hand runs on the users computer. Things like dialog boxes, cursor movements etc. are done completely local (compare versus Telnet sessions where every keystroke is transmitted over the net). This has not only the advantage of less network load, but also of fast interaction responds and best adaption to the properties of the operating system of the client computer. It also helps to keep the server small and efficient because it is not burdened with user interface considerations.

For more detailed information look at [Krol94] for the user's point of view and at [Liu94] for the server administrators point of view.

4.1 Gopher

Gopher has been developed at the University of Minnesota primarily as a distributed campus information system. The service was designed so that each piece of bureaucracy could have control over its own server and data. Gopher's developers then created a special application that could guide students to the information, with no training required. To do this, they organized the system hierarchically with topics, so that it looks like one large database, rather than hundreds of smaller databases. Every piece of information is represented by a menu line describing the content in short form (with a title).

The power of Gopher is that those menu items can actually point to files or directories on other Gopher servers, to FTP archives, to Telnet services, and more. Gopher was the first service to offer this kind of transparency where traversing different machines and different services on the Internet is as easy as browsing a local directory.

In addition to the hierarchical way of navigation Gopher offered another quite interesting feature: Searching. First it was only possible to search within one server but this was extended quite soon with the development of *Veronica*, a search robot that moves from one server to another and collects information about every menu item. Afterwards the user may put a question in form of a search query to Veronica and it answers with a list of menu lines (containing the required string) pointing to several different Gopher servers.

Another important feature to mention is that Gopher supports several different types and formats of documents right out of the box. In a Gopher server you can store text documents, images, sounds etc.

Unfortunately one very important feature was missing till recent times: *HyperLinks*. These are *links* within documents pointing to other documents, servers or basic Internet services. This was the reason for the tremendous success of the next information system, the *WorldWideWeb* (*WWW*, or W^3) that included this facility.

After all Gopher is not a dead system, development is going on and new features are developed frequently (Gopher+: see [Gopher+], GopherVR: see [GopherVR]).

4.2 WWW

The birthplace of the World-Wide-Web was the European Laboratory for Particle Physics (CERN) located in Geneva, Switzerland. For a long time there has been a special need for wide-area hypertext to support the research activities of particle physicists. The need arose from the geographical dispersion of research institutions working together on the same projects. Very often researchers like visiting scientists and students only work for a limited period of time before leaving for some other location. In fact most of the information they needed to familiarize themselves with the new work is already available on-line, but retrieval always needed some special knowledge of the network. One of the original aims of the initiative at CERN was to provide a system with a "point and click" interface. (For detailed information about the history of the WWW, look at [WWWHist].)

The striking new concept of WWW was the usage of *hypertext* over a distributed network. Hypertext does not only contain text but also hyperlinks. Hyperlinks are especially marked areas of text, where users can click on, which leads them to other hypertexts, the links are pointing to. Later a special type of links were created, called inline images. These links are automatically executed during the display of the text. They don't point to other text documents but to images that are inserted in the text document on the fly. What we get now isn't a boring simple text document any more, but a text involving colorful images and graphics.

This was the first time information systems also got the possibility to present the data they stored in a way so that it can be used for advertisement!

One of the unsolved problems of the WWW is that users might soon get ‘lost in hyperspace’ during their journey through the Internet due to the lack of a hierarchical structure of the data. There is no location feedback where and how deep they are currently in the web. In addition to that the concept of WWW does not directly include search capabilities. Instead many WWW servers are combined with some other search engine (like *WAIS*), but this again has the disadvantage that they have to be used differently and that there is no standard for more sophisticated search capabilities.

Further design weaknesses are that links are not stored in a separate link database but in the documents. This is the reason why currently only links in text documents are supported (there are no real links possible in images). Furthermore the underlying http (HyperTextTransferProtocol) is connectionless which means that all the information a server has to know about the client has to be retransmitted on every request, because there is no permanent connection between client and server. This is not very efficient and results in heavy network load. Because of this access rights and user management are also difficult to implement (they would also have to be retransmitted every time!). The lack of access rights makes it impossible to distributively provide information to the server.

Nevertheless WWW is the leading information system technology today and has widespread acceptance all over the world. The main reasons for this are certainly really well designed clients like Mosaic or NetScape with a very easy to use graphical interface combined with capabilities to use other information systems (like Gopher) and Internet services (like Usenet News, EMail, FTP).

4.3 WAIS

The main purpose of the *WideAreaInformationService* (*WAIS*) is to find things users are looking for on the Internet. There is no familiar way to navigate within this system, like browsing through a hierarchy or following links. That's why WAIS is often combined with other information systems.

WAIS doesn't really look at the data in the process of a search, but at an index. It's most common to see indexes for various kinds of texts (articles and so on), but you can also build an index for anything else (e.g. index images with a description of each image).

When you request a search from a WAIS client, it contacts a server that handles the libraries you suggested. This server in turn asks the other servers to search their index for a set of words (WAIS is based on a standard named called that describes a way for one computer to ask another to do searches for it.) The server then sends you a list of documents that may be suitable, and a 'score' telling how appropriate it thinks each one is. The scores are normalized, so that the document that matches your search criterion best (e.g. has most occurrences of your keywords included) is given a score of 1000; others get proportionally less. After all the libraries have been searched , WAIS gives you the titles of the documents that received the highest score. There's a limit to the number of documents it reports - usually between 15 and 50, depending on which client you use. You can then pick which documents to view, and the WAIS client will display it for you.

5. Hyper-G - a Second Generation Information System

WAIS, Gopher and WWW belong to the first generation information systems on the Internet. They work well in particular contexts but run into difficulties when applied to hundreds of thousands of documents distributed over many thousands of servers. They provide no graphical navigation aids, only rudimentary access control, little support for automatic database maintenance, and no support for multiple languages. *Hyper-G* has been designed to be a general-purpose, large-scale, distributed, multi-user, second generation hypermedia information system that supersedes these problems.

Based on an analysis of the strengths and weaknesses of existing systems, the following primary design goals were formulated:

- Distributed server concept
- Standardized search capabilities
- Anchors stored in a separate link database and not embedded in the documents
- Object oriented database where documents are stored in a hierarchical structure
- Support of user identification and access control
- Support of multilinguality
- Maintain interoperability with existing systems

The Hyper-G project has been in progress at the Institute for Information Processing and Computer Supported new Media (IICM) of Graz University of Technology and the Institute for HyperMedia Systems (IHM) of Joanneum Research for a number of years.

5.1 Client-Server / Server-Server Architecture

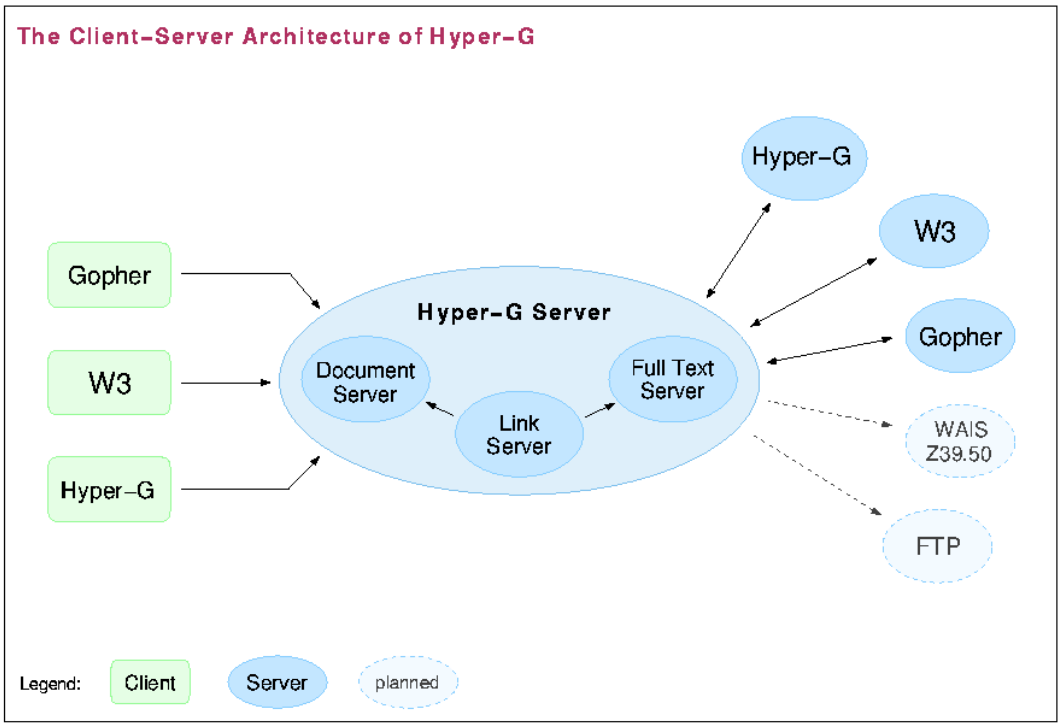

Figure 1 -The client server architecture of Hyper-G

The Hyper-G server concept consists of three server parts:

- The *Link Server*: for storing object (including anchors and documents) attributes and the relations between them
- The *Document Cache Server*: for storing the data of the documents and caching remote documents
- The *Fulltext Server*: for indexing text documents and doing fulltext queries

Also included in the concept are several gateways for different clients and servers. These are currently (July 1995, some more will follow in the near future):

- A gateway to WWW clients, so that clients like Mosaic and Netscape can also access a Hyper-G server
- A gateway to Gopher clients
- A gateway to WWW servers, to allow Hyper-G clients access to WWW servers (without the need that Hyper-G clients know the http protocol)
- A gateway to Gopher servers (for the same purpose)
- Gateways to WAIS and FTP servers are currently under development.

To support distributed operations between several Hyper-G servers, also a *server-server protocol* has been implemented.

5.2 Data Model

The Hyper-G concept includes an object oriented database especially designed for the requirements of an information system. All data types are objects containing particular functions and attributes that can be inherited by derived objects.

This a short overview of the most important objects:

- *document*: is the smallest data entity from which all other document types are derived, such as (in brackets the supported data format):
  - *text* (currently only HTF: Hyper-G text format, see [HTF] for a description of the format), but HTML 3.0 (HyperTextMarkupLanguage, see [HTML]) will be supported very soon
  - *image* (GIF, JPEG, TIFF, XBM, ...)
  - *movie* (MPEG)
  - *sound* (au, WAV)
  - *3D scene* (VRML: VirtualRealityModelingLanguage, and a format derived from Wavefront)
  - *Postscript* (PostScript 1 and 2, PDF will be supported very soon). This document type is quite interesting if you want to do electronic publishing, *J.UCS* (“*Journal of Universal Computer Science*”, an electronic scientific journal, based on Hyper-G) is one example for this.
  - *remote* (remote connections to other Internet services and information systems like WWW, Gopher, WAIS, FTP, Telnet, ...)
  - *generic* (userdefined document types like an Excel spreadsheet, a PowerPoint presentation, a Midi sound file, etc.)
- *Cluster*: this is a group of documents that belong together in some manner and form a unity. A typical example for this is a cluster containing the same text in English and in German (thus two documents), a sound document and an image. If this cluster is visualized the user will see one text (English or German, depending on what the user selected as the preferred language) and the image and hear the sound document.

- *Collection*: contains other documents, clusters or sub-collections. Collections are quite similar to directories in an ordinary file system, with one difference: The structure behind collections is an *acyclic tree*. This means that a subcollection (and other documents) can be a member of several collections, thus it has more than one parent. Suppose we have two collections ‘chemistry’ and ‘biology’, then the subcollection ‘biochemistry’ would have to be added to both of them. Of course it isn’t duplicated for this purpose, because the relations between collections and subcollection are stored and treated like links. This means that the physical location of a collection is irrelevant for the hierarchical structure. This also has the benefit that collections (and also documents) can easily be moved from one place to another (e.g. disk) without the requirement to change the hierarchical structure of the database!
- *HyperLinks*: are also stored as separate database objects and are not embedded in the documents (like in WWW HyperText). An *Anchor* object bears only a relation to a (source or destination) document. This technique has several very important advantages:
  - we can have links between every type of document (e.g. movies, sound, PostScript, etc.). It does not matter whether we know the exact data format of the document nor, whether we may modify it or not (because the document might be stored in a write protected part of the database, e.g. a CD-ROM).

- links are *bi-directional*: This means that we can retrieve links that are pointing **to** a certain document. This makes it possible to implement navigational aids like the '*local map*' which displays the surrounding (other objects that are in context with this one) of a selected document.

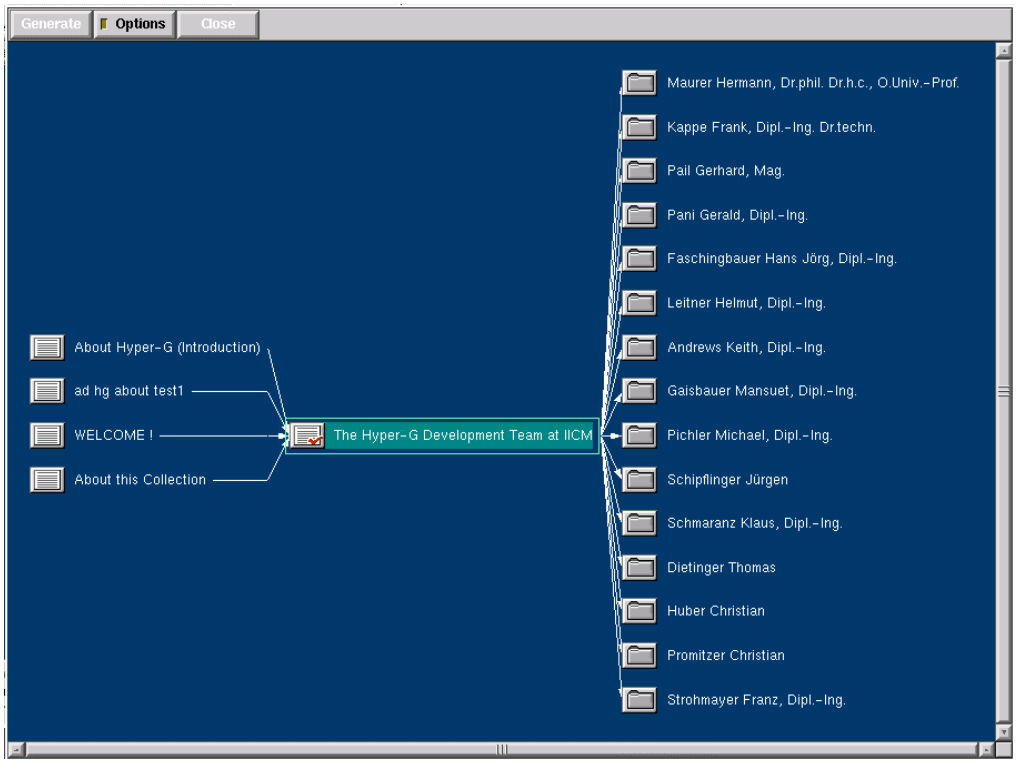

Figure 2 - The 'local map' from Harmony

- A document to which links are pointing to may be removed from the database and replaced by a new version (containing the same title). The old links will automatically point to the new document although it might be stored in a completely different physical location now! Meanwhile if the (destination) document is missing, all links referring to it will be deactivated. That's why it is impossible to have *dangling links* (links that are pointing to nowhere) in Hyper-G.

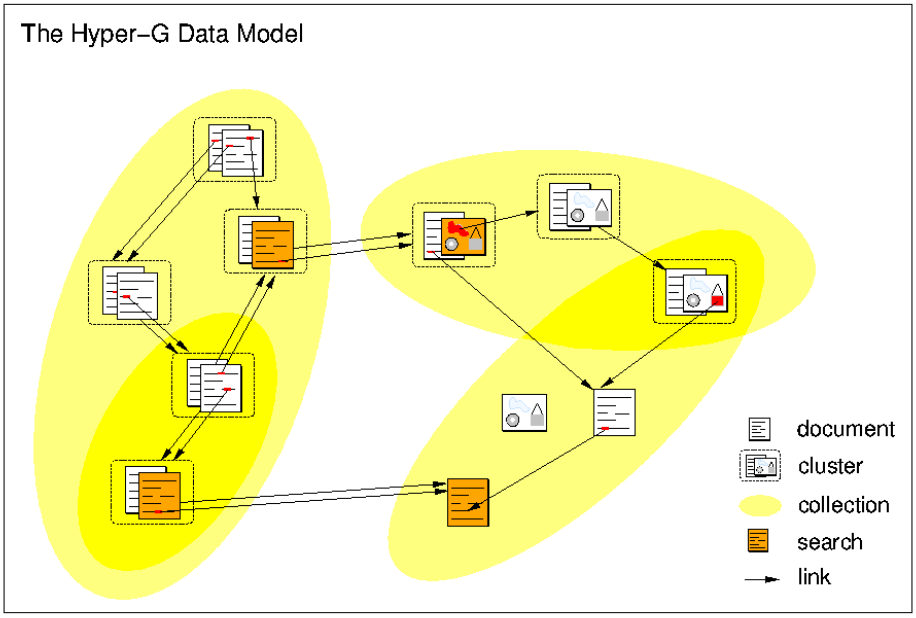

Figure 3 - The Hyper-G data model

5.3 Search Capabilities

Hyper-G has a built in a rich set of search capabilities.

Generally a user can specify **what** to search for:

- the object *title* (regular expressions may also be used)
- special object *attributes* like the author, creation date, keywords etc.
- *fulltext*: The fulltext server of Hyper-G is able to search for every word (except *stop-words*) in every text document stored in the database. This is done by indexing the text document during insertion into Hyper-G. This index also adds a score to every entry depending on the importance of the words (e.g. a heading is more important and gets a higher score than ordinary text). If the search query is matched several times in a text document it also gets a higher score. The scores are considered when users get a list of search

results. Very similar to WAIS these scores are also normalized. Apart from the usual regular expression possibilities, fulltext search offers even more sophisticated capabilities like fuzzy search etc..

In addition to the selection of the type of search users may also specify **where** to search. Basically we distinguish three types:

- Searching in the *whole home database*: The search goes over the whole Hyper-G server the user is currently connected to.
- Searching in *activated collections*: The search is restricted to certain parts (collections and their sub-collections) of the Hyper-G server that the user marked to be of special interest.
- Distributed search in several Hyper-G servers: In addition to collections on the local server, also collections on other Hyper-G servers may be activated. With regard to this it is quite interesting to note that users may get a list of all Hyper-G servers that are currently running (*Hyperroot*). With this list it is quite easy to mark additional servers to be included in a distributed search.

5.4 Multi-lingual Concept

An information system that makes demands to be of general purpose has to support multiple languages. In Hyper-G it is possible to assign every document a certain language or make it multi-lingual. Users may specify to the client what their preferred language(s) are and whenever possible they will only get those documents that are assigned to these languages. In a next step even the user interface of the clients will be adjusted to the preferred language (Harmony already supports English and German user interface, Amadeus will do so quite soon).

Currently Hyper-G supports the following languages:

- English
- German
- French
- Italian
- Spanish
- Styrian (an Austrian dialect)

Of course additional languages can be added to the system quite easily. In principle Hyper-G supports nearly every language because it supports the *UniCode* standard. Only the full text server's stop word list and the text-viewer of the clients have to be adjusted to handle new characters.

5.5 User Administration and Access Rights

One important key to an easy to use and manage large scale information system is to include a user management and good access rights to keep an overall view of what happens with the database. Without these prerequisites it is also impossible to distributively (over several different users and clients) add new data to the system.

In Hyper-G every document may have with several access rights (currently *read*, *write* and *unlink* rights) which may be assigned to certain users or groups of them (Users may belong to particular user groups, which automatically gives them certain access rights that are common for these groups).

One disadvantage with identified users is that the information system may store information about every single command the individual user issues when using the system, which is possibly referred to as the "*Big Brother*" scenario. The approach taken in Hyper-G allows not only anonymous and identified users, but support instead *four modes of identification*:

- *Anonymous*: In this mode no monitoring of individual users is possible. On the other hand, no write permission is granted to anonymous users.
- *Anonymously identified*: Anonymous users may choose pseudonyms and passwords by which they are identified. However, the system does not know their real identity, but can recognize the user in later sessions and can restore user configurations and preferences. Write operations are only allowed for documents that are not visible to the public.
- *Semi-Identified*: In this mode the real identity of the user is known to the system, but to nobody else. This assures that users are prevented from making slanderous comments, contributions violating some pornography law, etc. (since they can be identified, if necessary) yet their real identity is shielded from other users: they can choose an arbitrary pen-name. This mode has proven to be particularly valuable for discussions in groups to assure that the usual barriers of job-hierarchy are broken down, and a discussion in a "space free of power and fear" can be conducted. Semi-identified users are allowed to create both private and publicly accessible documents. A user may also have more than one pseudonym.
- *Identified*: The identity of the user is known to the system and also to other users when looking at the "who-is-online" list, receiving mail, or reading annotations from identified users.

In connection with user identification another feature has to be mentioned. Hyper-G supports so-called *home collections*. They work as an individual entry point to the system for each user. If users identify themselves they can go to their home collection (which is a private property of the user).

The user can copy references to other documents or other collections to the home collection using it as a personal bookmark or hotlist for items of interest. This allows faster navigation to these documents when needed in subsequent sessions.

6. Amadeus - a Hyper-G PC Client

Amadeus is the first Hyper-G PC client for Microsoft Windows. It is especially designed to make most of the features Hyper-G is capable of available for a broad range of customers. Because it should not only be used as a viewer but also as an authoring tools for people that are not highly educated in computer science one of the key features had to be user friendliness. Another aspect was that it should also be usable on PCs typical for ordinary office equipment (and not only high-end machines). Last but not least it should be implemented very modular (in C++) to ease maintenance and extension of the application.

Some of the key features of Amadeus are:

- native 32 bit-application (Windows 95 ready!), runs under Windows 3.1x with the help of the Win32s extension.
- comfortable installation program
- typical Windows like features:
  - MDI (multiple document interface),
  - dockable-toolbar with toolbar-tips,
  - context-sensitive popup-menus,
  - extensive online-help etc.
- numerous build-in document viewers for several data formats:
  - text (HTF and HTML)
  - raster-images (GIF, JPEG, TIFF, PCX, BMP, XBM, ...)
  - movies (MPEG)
  - PostScript-documents (PostScript 1 and 2)

- 3D-scenes for virtual reality (currently implemented as an external viewer that only runs under Windows NT 3.5x)

- build in converters from RTF (rich text format) to HTF and from HTML to HTF. With this you can use ordinary word processors (that are capable of saving text in RTF) to create new text documents for Hyper-G. Especially for Word 6.0 we have a separate document template which, in conjunctions with another converter, makes it very ease to create good looking texts.

The following chapters should give a concise description of Amadeus beginning with the requirements and the installation, followed by a narration of the components and at last by a detailed reference of all functions and features of Amadeus.

Note: Amadeus is free software (for educational institutions as well as commercial organizations) and may be downloaded via anonymous-ftp from our ftp server '*iicm.tu-graz.ac.at*' in the directory *'/pub/Hyper-G/Amadeus'*.

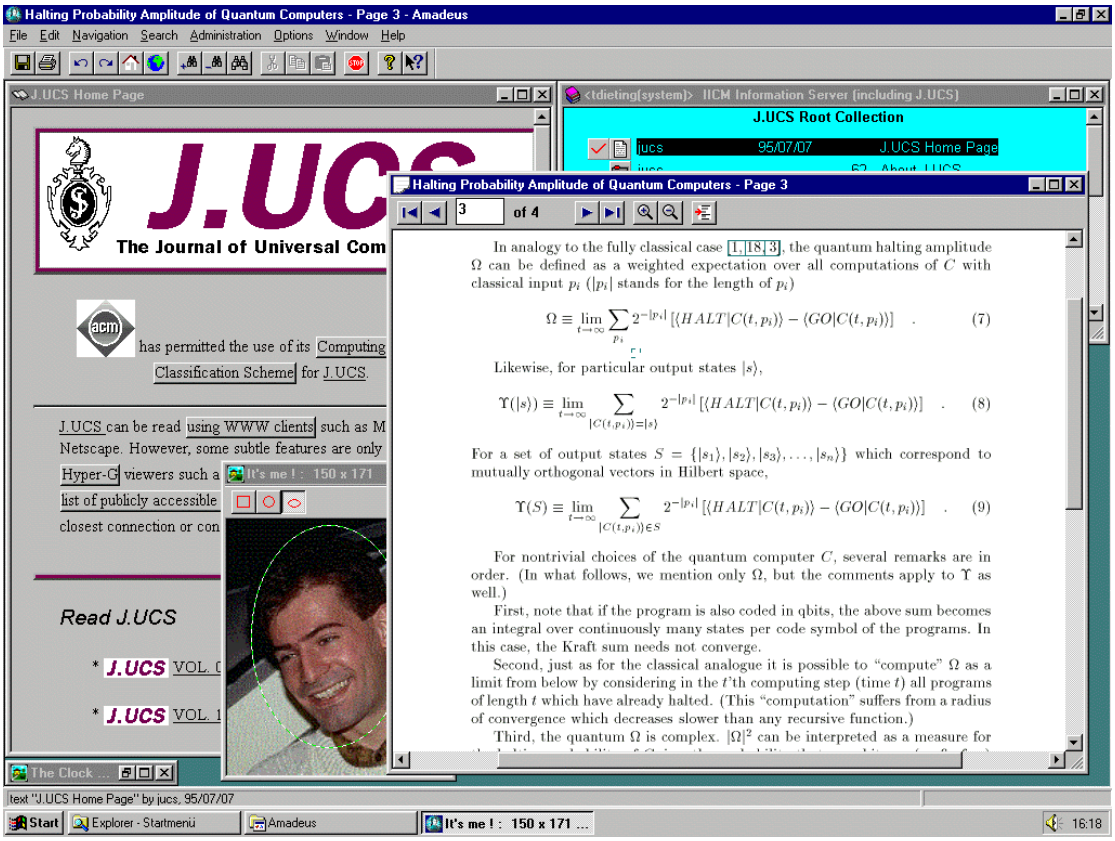

Figure 4 - Amadeus - a typical screenshot

6.1 Implementation and design concepts of Amadeus

The basic idea behind the concept of Amadeus was to implement a well designed structure of independent module layers that can easily be extended, reused in other applications and substituted by completely new code without the need to understand the whole project. To support a broader range of platforms it was clear, that I had to use C++ as the primary programming language.

Generally Amadeus consists of three layers with numerous classes:

- The *protocol and Hyper-G object* layer provides the basic functionality for accessing and handling various data objects available with different Internet protocols (such as Hyper-G documents, FTP-directories, News groups, etc.)
- The *command layer* is the control center of Amadeus and works as the connection between the user interface and the core functions.
- The *viewer layer* provides an application interface to independently develop and test a new document viewer without the requirement to have and understand the complete source code of Amadeus.

Note: I will release a complete tool-kit for implementing new clients for Hyper-G and other Internet services, that is based on the classes and layers described in this section.

6.1.1 The Protocol/Object concept

Initially it was planned that Amadeus should only be a native Hyper-G client and every access to the Internet should go over a Hyper-G server. But soon the demand to access other Internet services, that are not yet supported by the server, arose. Especially after we decided to implement a local database engine in Amadeus, it was clear that I had to split the logical representation of database objects (like text, images, collections, anchors, ...) and the physical methods to access them. That led to two (C++) base classes with virtual functions, from which all other important classes are derived:

- HGObject (for the logical representation of objects, the 'HG' stands for Hyper-G for historical reasons, in spite of being the basis for all objects, not only Hyper-G objects)
- BaseProt (provides the access functions for all kinds of objects and protocols)

6.1.1.1 The 'HGObject' classes

This and all derived classes should represent database objects from a logical point of view and should encapsulate physical differences (e.g. for accessing document data). All objects are characterized by a certain set of properties and virtual member functions (like inserting, deleting, moving, etc.). Every object knows its original protocol via a pointer to the BaseProt class (see next chapter) and thus how to physically perform operations on this object using the virtual member functions of BaseProt. This makes it possible to use the same types of documents (e.g. images) based on different protocols (e.g. Hyper-G and Gopher). This has the advantage that functions that are unique for this type of document must only be coded once. There is no need to provide a HyperGImageDocument and a WWWImageDocument, just an ImageDocument. Furthermore new protocols may be easily added without the requirement to change the HGObject classes or other parts of the Amadeus source code.

Only one very basic function has to be extended: The '*makeObjectById*' function. It takes a unique object identifier (e.g. a URL or a virtual Hyper-G object id) as the input parameter, evaluates it, creates the required protocol class (from BaseProt derived) and uses the protocol to get the object attributes. The result is an object derived from HGObject. Of course this function also considers the behavior of connection oriented protocols like Hyper-G and FTP, where no new connections to the same host must be established (this is done by keeping a list of open connections) and some additional features like using Hyper-G as a proxy-server for WWW and Gopher.

Note: Future versions of this function will also support dynamical loading of protocol libraries (DLLs), so that it is guaranteed to keep hardware requirements low when only a few protocols are used.

Look at “Figure 5 - The HGObject class hierarchy” for an overview of currently supported object types:

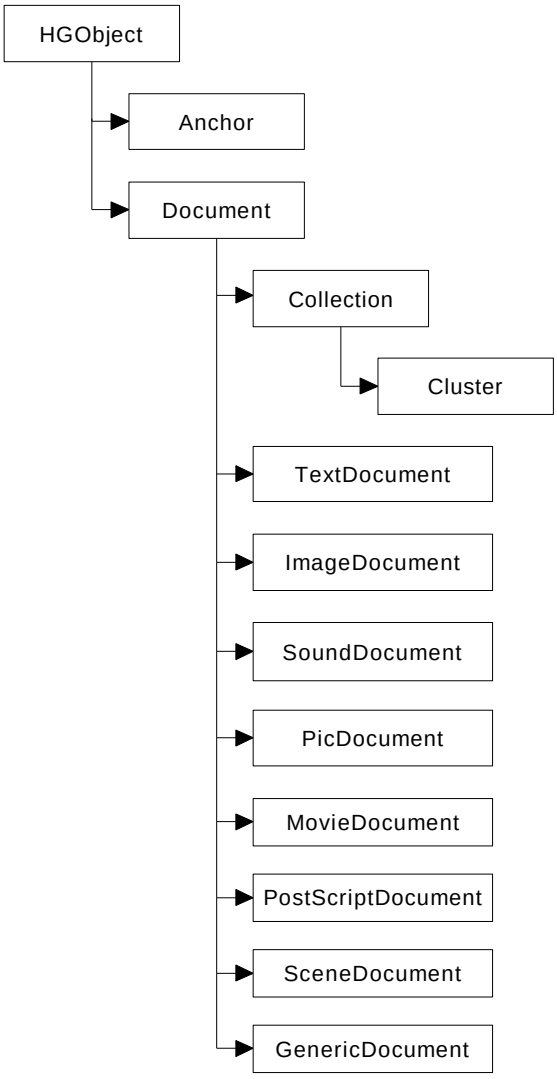

Figure 5 - The HGObject class hierarchy

Here is a short insight into (parts of) the header file to get a feeling for the implementation of HGObject:

```
class HGObject : public Resource
{
public:
  static ProtocolList conOrProtList_; // list of connection oriented
                                      // protocols
  HGObject(const Object& o, ProtPtr pPtr = ProtPtr() );
  virtual ~HGObject() { }             // is called by unref

  virtual boolean update();           // updates object attributes to
                                      // current state in database
  const RString& Type() const             { return _type; }
  const RString& Author() const           { return _author; }
  const RString& Rights() const           { return _rights; }
  const RString& Created() const          { return _created; }
  const RString& Modified() const         { return _modified; }
  const RString& Expires() const          { return _expires; }
  const RString& Opens() const            { return _opens; }
  const RString  Title();
  virtual RString IDString() const        { return _id.IDString(); }
  virtual RString Info() const;
  virtual RString ClientAttributes() const;
  HGObjectEnum   getType() const          { return objType_; }
// ...

  virtual ObjectList* myParents()
  virtual ObjectList* myChildren()
  virtual ObjectList* linksTo()
  virtual ObjectList* linksFrom();
  virtual ObjectList* keyQuery(RString& query,
                               ObjectList* activeCollList = 0);
// ...
  virtual boolean erase();
  virtual boolean insert(ObjectPtr toColl);
  virtual boolean moveTo(ObjectPtr fromColl,
                         ObjectPtr toColl);
  virtual boolean copyTo(ObjectPtr toColl);
  virtual boolean unlinkFrom(ObjectPtr fromColl);
  virtual boolean lock();
  virtual boolean unlock();
// ...
}
```

Note: *ObjectPtr* is a special class that includes a pointer to an HGObject including a resource management for this object (reference counter) to ensure correct usage of the pointer. Similarly *ObjectList* is a list of ObjectPtrs.

As you can see, there are many attributes like the title, the author, access rights and so on, describing the properties of an object and a lot of member functions to perform some actions on it. Additional attributes and functions are included in derived classes.

6.1.1.2 The ‘BaseProt’ class philosophy

The main goal of this class is to define a set of functions as complete as possible that might be required to make full use of all features a certain protocol can provide. If only these functions are used for generating the logical representation of the database objects (DBOs), it is quite easy to implement a new protocol that immediately works within Amadeus.

First I had to include basic operations on DBOs like

- retrieving
- inserting
- deleting
- moving
- duplicating (physically copying)
- linking (making a new relation between two objects)
- unlinking (deleting a relation between two objects)
- locking
- unlocking

Then there are some functions that deal with two different types of data:

- data (content) stream
- list of DBOs

To keep the class simple (to implement and use) this is done by only six different functions:

- opening with a special opening mode (specifying the type of operation)
- retrieving a data stream (getData)
- putting a data stream (putData)
- retrieving a list of objects (getList)

- putting a list of objects (putList)
- closing the stream or list

Similar to ordinary file operations I also use a special kind of file descriptors (that is internally implemented as a C++ class, called *BaseFD*, from which other classes may be derived) as a common element of control between these functions.

Currently I implemented the following modes for the open operation (the list may be enlarged if required):

- getDataMode
- putDataMode
- getChildrenMode
- getParentsMode
- getReferencesMode (returns a list of anchors pointing to this object)
- getLinksMode (returns a list of all anchor objects pointing FROM this object to others. In Hyper-G source anchors, as well as the corresponding destination anchors are included)
- keyQueryMode
- fullTextQueryMode

These are only the basic functions a new protocol *might* support. A (HGObject derived) class that deals with them has to be aware that some of them may also return ‘not implemented’ as an result! This makes it possible to create new protocols that only include a subset of these operations (e.g. FTP which doesn’t support links, or http which doesn’t know anything about a hierarchy). Functions that are special features of a certain protocol may be included in a derived class, but must be type checked and casted in order to have access to them.

Future versions of this class may also include some high level methods for *actions* on objects (e.g. clicked on a certain area within an object (if there is no link?), etc.).

To clarify what I explained above, you can look at an excerpt of the header file of BaseProt:

```
class BaseProt : public Resource
{
public:
  BaseProt(RString connectStr)
  ~BaseProt();

  // get object information by destination ID/URL
  virtual Object getObjectInfo(RString dest);

  // ...

  //
  // placeholders for protocol functions
  //
  virtual BaseFD*     open(ObjectPtr objPtr,           // pointer to object
                                                       // of interest
                           OpenMode mode,              // opening mode
                           ObjectList* objList = 0,    // pointer to an
                                                       // additional list
                                                       // (e.g. list of
                                                       // activated
                                                       // collections)
                           RString helpStr = ""        // additional string
                                                       //(e.g. query string)
                           );
  virtual boolean     close(BaseFD* fd);

  virtual long        getData(BaseFD* fd,
                              char* dataBuf,
                              long buflen);
  virtual long        putData(BaseFD* fd,
                              char* dataBuf,
                              long buflen);

  virtual ObjectList* getList(BaseFD* fd,
                              long count,
                              ObjectList* oldList = 0);
  virtual boolean     putList(BaseFD* fd,
                              ObjectList* newObjs);

  virtual boolean     insert(ObjectPtr object,
                             ObjectPtr toColl);

  virtual boolean     erase(ObjectPtr);
  virtual boolean     moveTo(ObjectPtr,
                             ObjectPtr fromColl,
                             ObjectPtr toColl);
  virtual boolean     duplicateTo(ObjectPtr,
                                  ObjectPtr toColl);
  virtual boolean     linkTo(ObjectPtr,
                             ObjectPtr toColl);
  virtual boolean     unlinkFrom(ObjectPtr,
                                 ObjectPtr fromColl);

  virtual HGObject*   lock(ObjectPtr);
  virtual boolean     unlock(ObjectPtr);
  // ...
}
```

The design also has the advantage that Amadeus may very easily be extended to support other non-Internet services like access to the local file system, to other databases (e.g. over ODBC), etc..

6.1.2 The command stack and history list

Amadeus does not immediately execute a command when it is initiated by the user but collects it on a stack. This has the advantage that it may be executed asynchronously when time is convenient and the operation is allowed.

Every information required for performing the command is stored in this *command object*. This makes it very easy to reuse the command at a later time and store it on a history stack in the meantime. Again commands are C++ classes derived from a base class (called '*Command*').

Currently these classes are tied to the foundation classes and development environment Amadeus is based on (Visual C++ 2.1 and MFC 3.1) because many Windows dialogs are included in them. But I will also try to divide these classes in Windows specific and universal useable (C++) code till the release of the tool-kit.

6.1.3 The Viewer concept

Amadeus can deal with external and built-in Viewers. However currently only viewers that are directly integrated in Amadeus can support link handling. To prevent successive growth of the code size, future versions (under development) will load internal viewers dynamically and thus make it possible to execute Amadeus even on a smaller machine.

External viewers are controlled via settings in the Amadeus.ini file and magic numbers. Magic numbers are a special sequence of numbers or characters that are unique for a specific data format and are used to find out the document type (in Hyper-G Amadeus already knows the basic type, e.g. whether it is an image or a movie, but it does not know whether it is JPEG or a GIF etc.). As soon as Amadeus knows the exact format it looks in its configuration file to find out which viewer is assigned for visualizing it. If there is an entry it tries to launch an external application or to open an internal viewer window (after loading the required DLLs). This makes it possible to have different viewer applications for several data formats (e.g. AU and WAV) even if they are the same basic document types (e.g. sound).

Currently internal viewers are based on the document-view architecture of the Microsoft Foundation Classes (this will change in the future to support also other development environments). To simplify development of new viewers I designed a collection of base classes and some dummy classes that provide the same interface as the Amadeus classes but don’t include network functionality. These classes also contain simulation for link management to fully test new viewers in a stand-alone project (thus without the real Amadeus sources).

The most important classes are:

- *class CApplication : public CWinApp*: base class for all viewer applications (includes things like accessing the configuration file, providing several types of cursors etc.)
- *class CMDITemplate : public CMultiDocTemplate*: extended class for handling the MFC template macro (serving as the connection between documents, frames and views)
- *class CViewerDocument : public CDocument*: base class for all documents that want to access HGObjects and derived classes. Includes basic HGObject dummy classes and link emulation.
- *class GUIControls*: provides (platform independent!) functions for handling interruption of lengthy operations and displaying progress indicators (visualized by derived classes).
- *class CMDIMainFrame : public CMDIFrameWnd*: base class for frames in an MDI application (like Amadeus)
- *class CSDIMainFrame : public CFrameWnd*: base class for frames in an SDI application (like the Public Terminal Viewer)

6.2 Amadeus Requirements and Notes

6.2.1 Required Hardware

Amadeus requires as an absolute minimum of an 80386-based machine with 4 MB RAM. Our recommended configuration is a 33-MHz, or faster, 80486 with at least 8 MB of RAM (>=256 color display is also recommended.). The Amadeus binaries require min. 2 MB / max. 7 MB (depending on the configuration, + 2 MB for Win32s) of disk space, and you should have about 10 MB free in your TEMP directory when running Amadeus.

6.2.2 Required Networking Environment

In order to use AMADEUS you must have direct access to the network, i.e. your machine has a network card and is connected to the network or you use your modem to call a terminal server that allows SLIP or PPP connections. This direct access must support the WinSock networking specification. To run Amadeus, you will need a system running Microsoft Windows 3.1 in enhanced mode, a 1.1 compliant winsock.dll and Microsoft's win32s software. Windows NT 3.5x and Windows 95 users have these DLLs built in.

The winsock.dll provides the TCP/IP networking under Windows. Amadeus is a winsock 1.1 compliant program. If you are using a commercial TCP/IP stack such as PC-TCP, PC-NFS or running a local area network such as Novell in addition to TCP/IP, you need to contact your network vendor directly about obtaining the WinSock DLL.

If you are accessing the Internet with a stand-alone PC running Windows, you may use a shareware winsock called the "Trumpet Software International Winsock". A copy of the Trumpet software can be found at ftp.utas.edu.au in the pc/trumpet/winsock directory.

Note 1: Our PC/TCP 2.20 does not work with multimedia documents, but version 2.30 works perfectly

Note 2: The hostname of your PC must be listed in the TCP/IP host file if aliases are used.

6.2.3 Microsoft's Win32s software

AMADEUS is a 32-bit application and therefore requires the win32s software when run under Windows 3.1 or Windows for Workgroups. Win32s allows you to run 32-bit Windows applications which do not use Windows "NT-specific features" under Windows 3.1 and Windows for Workgroups.

Win32s was developed by Microsoft and it is freely distributed to licensed users of Windows 3.1 or Windows for Workgroups. If you already use other 32-bit Windows applications then you should already have this software installed in your /windows/system/win32s directory. If AMADEUS is your first 32-bit application then the setup program of Amadeus will install win32s before installing AMADEUS.

6.2.4 Bugs

This is a pre-release software, that means there will be bugs. We are very interested in bug reports. Please send electronic mail to '***amadeus@iicm.tu-graz.ac.at***' with a concise description of the error, INCLUDING what you did to generate it. Please try to re-produce the error.

Note: Be sure to read 'reame.1st' and 'amadeus.faq', before sending us the report, this might help you solve your problem or prevent sending us reports about known bugs.

6.3 Installing AMADEUS

Note: If you haven't got Amadeus on floppy disks, but have downloaded the zip-files you first have to unzip amadeus?.zip (? stands for 1,2, ..) in a temporary directory or on a set of floppy disks (each zip on one floppy).

There are two setup programs: 'setup.exe' on disk #1 which allows you to choose full control over all setup features and 'setup2.exe' on disk #2 which only works for local installation on Windows NT, Windows 95, and on Windows 3.xx systems that already have installed Win32s (but do not require disk #1 at all, so you don't need to download it !). Disk #4 and Disk #5 are only required if you want to install the PostScript viewer. Disk #6 includes the 3D scene viewer and an external version of the PostScript viewer to be used with other applications or stand alone.

Suppose you started ‘setup.exe’ from disk #1 (which works for all platforms), then you will get the following dialog box:

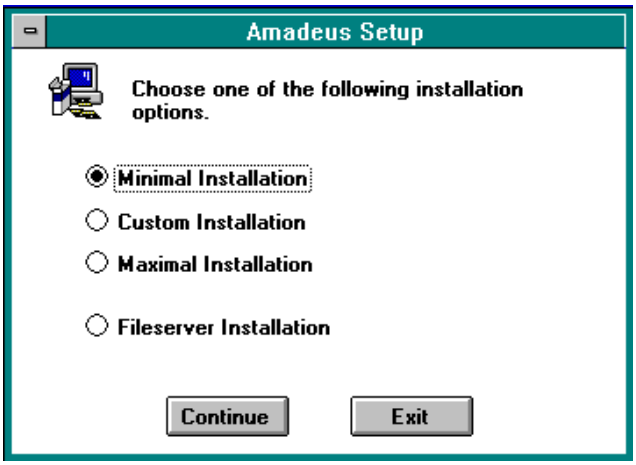

Figure 6 - Installation option dialog

6.3.1 Local installation

All Amadeus binaries will be installed locally (directly on your PC), if you choose one of the first three options. For testing purposes, or if you don’t have enough space left on the harddisk of your PC, you can click on ‘Minimal Installation’, this one requires about 2 MB (+ 2 MB, if you need Win32s and don’t have installed it in your system yet). ‘Custom installation’ will be the best choice if you don’t know what’s going to be installed and want to look first at the possibilities you have. ‘Maximum installation’ will make a ‘full-blown’ Amadeus system, but be careful, some things might only work on Windows 95 and/or Windows NT 3.5x (e.g. the upcoming 3D-scene viewer), that’s why it is better to use custom installation, if you only have Windows 3.xx !

Note (only for Windows 3.xx users): After you have finished your installation, Windows 3.xx will restart if the setup-program installed Win32s or WinG (a high speed graphics library). If you start Amadeus for the very first time, a graphic speed test will be made (test pattern) to check what will be the best display methods for your video driver. If you have difficulties during this step, please read the file ‘amadeus.faq’, which should come with the Amadeus files. (There are some known problems with an older S3-SVGA driver !)

6.3.2 Fileserver installation

This kind of installation is a very good choice if you are going to install Amadeus on a number of PCs and have a fileserver which is accessible by the PCs. Here all Amadeus binaries (except Win32s and WinG for Windows 3.xx systems) reside on the fileserver, only the Amadeus configuration file (‘amadeus.ini’) which stores your personal settings, will be copied to the local PC. The great advantage of this is not only that you need very little harddisk space on the local PC, but it is also very easy to update to a new Amadeus version by simply installing a new version on the file server. Then all Amadeus clients will run the new version without any change on these PCs !

After you have copied the Amadeus installation files to your file server (with the fileserver option), you need to install Amadeus on the local PCs, using the Amadeus setup program **on the fileserver** ! The install program will copy Win32s, WinG (if necessary) and amadeus.ini to the local disk and generate an icon to access Amadeus. (Note: You have to do this only once, and not after every upgrade of the Amadeus fileserver files !)

6.4 The document viewers of Amadeus

6.4.1 The collection browser

The collection browser displays the Hyper-G database using its hierarchical structure. Thus it displays the content of a collection as a list showing some short information about its objects.

In general, the lines look like this:

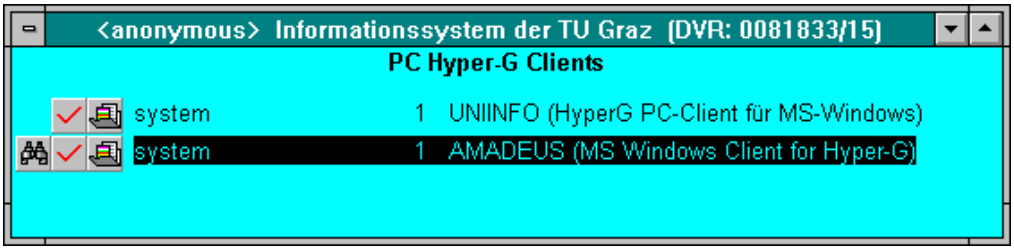

Figure 7 - The Collection browser

The first icon () tells you that this cluster is included in the search list (see The „Activate“ and „Deactivate“ command, page 87), the next one () tells you that you have already seen this cluster. Then the type of the object is displayed (explanation: see below). After that the author, the number of objects in the collection/cluster or the creation date (if it’s an ordinary document) and at last the title is listed.

6.4.1.1 Types of Objects

Icon	Object Type
	Collection (similar to a directory)
	Cluster (special collection, where the system selects members for display)

 Text Document

 Image Document

 Sound Document

 Movie Document

 3D - Scene

 Generic Document

 Unknown Document

 Destination anchor

 Remote File (anonymous FTP)

 Telnet Session (on-line connection to interactive service on remote host)

 Unknown Remote Connection

 Gopher Collection (like collection, but from remote Gopher server)

 Gopher Text (like text, but retrieved from remote Gopher server; no links)

 Gopher Image

 Gopher Index (searchable Gopher collection)

 Gopher Sound

 WorldWideWeb Document (hypertext from remote server, sometimes searchable)

 WAIS Source ("Wide Area Information Server"; remote full text database)

 WAIS Text (text received from WAIS server, otherwise like Gopher Text)

 WAIS Image (image from WAIS server)

Note: After a search command, a short name associated with the parent collection of the object is displayed instead of author and creation date. Should the object belong to more than one collection, one of it is displayed followed by a '+'. After a full text search, a score value is also displayed.

The list is sorted according to the currently set sort options, except for search results which are always sorted by score and parent collection, and some collections where the information provider wants to maintain a certain order of objects.

6.4.1.2 Choose an Object

To choose from the list you can either double click on the line (use the scroll bar to browse through the list) or use one of the following keys:

„down arrow“:	Moves cursor one line down. When the end of the page is reached, the list will scroll up. When used on the last line, the cursor jumps to the first line.
„up arrow“:	Moves cursor up a line (if possible). From the first line you jump to the last line.
„page down“:	Scrolls lines one page down (if possible).
„page up“:	Scrolls lines one page up (if possible).
„Home“:	Jumps to the first line.
„End“:	Jumps to the last line

RETURN, „right arrow“:	Selects the item under the cursor and issues a command that depends on the type of object under the cursor. In case of a collection the members of the collection are shown (the list is updated), whereas when applied to a document the document is shown.
BACKSPACE, „left arrow“:	Equivalent to entering the back command.

Note: In addition to that you can get a ‘context sensitive popup menu’ when you click the right mouse button on an object. Context sensitive means that you get only those options that are valid for the current object, e.g. you get only a ‘children’ menuline, if you click on a collection or a cluster.

6.4.2 The text viewer

The text browser shows text coded in HTF (Hyper-G Text Format), an SGML-compliant format that allows formatting of the text at run-time. This enables you to change the window size in text mode resulting in immediate reformatting of the text.

The text may contain so-called links to other documents that are usually displayed with a 3D-effect border, making the link look like a button. If the link is surrounded by a different color then this means that the target document the link points to has already been visited during the current session.

Note: In older versions of this viewer an asterisk (*) immediately after the link indicates this state.

If you double click on a link the text browser will follow this link and therefore display a new document or a new collection list in the collection browser.

6.4.2.1 Key Bindings

To read the text, the following keys may be useful:

„down arrow“:	Scrolls text one line down (if possible).
„up arrow“:	Scrolls text one line up (if possible).
„page down“:	Scrolls text one page down.
„page up“:	Scrolls text one page up.
„TAB“:	select next link (for following it or deleting it), if any
„Shift TAB“:	select previous link (for following it or deleting it), if any
„ENTER“:	follow selected link
BACKSPACE, „left arrow“:	Returns to the previous state (equivalent to entering the back command).

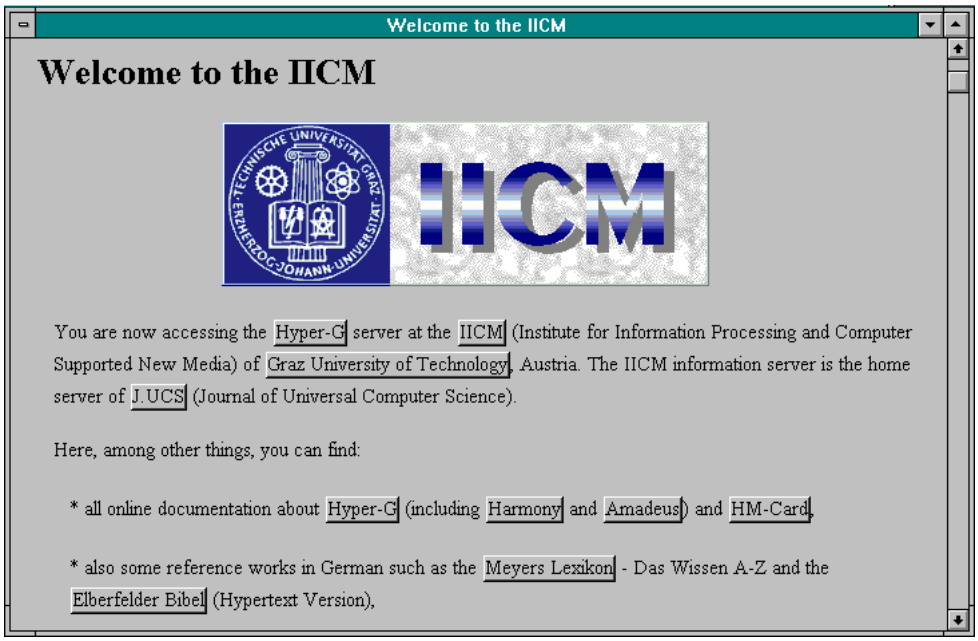

Figure 8 - The Textviewer

6.4.2.2 Mouse Navigation

Apart from the usual mouse controls (like adjusting the scroll bar, sizing and positioning the window etc.), there are two additional very important functions:

- *Marking* parts of a text for link creation: you can mark a single word by double-clicking on it or you may *drag* (moving the mouse with left mouse button pressed) over the text that you want to select. Afterwards it will be displayed inverted and you can create a link with the selected text as the source anchor using 'Edit/Make Link' (see
- The „Create Link" Command, page 73) or a context sensitive popup menu (see below).
- Inline Images: can be created by positioning the *caret* (blinking vertical bar) on the place where you want the image to be displayed. Then call 'Edit/Make Link' to specify the target image. (Note 1: Thus the image has to be inserted in the database **before** you create the inline link !) Note 2: No piece of text must be selected (indicated by the inversion) during this operation because then Amadeus would think that you want to create a real link but not an inline (image) link ! (automatic link type detection)
- Opening a *context sensitive popup menu* when clicking on the right mouse button: this will be a very useful shortcut to important functions of the text viewer and includes things like: making/deleting of links, changing the font you clicked on, viewing document properties etc.

6.4.3 The image viewer

- is used to display various image formats and associated links (if any). New links are displayed with a dotted rectangle/circle/ellipse, ‘seen’ links (means the target document the link points to has already been visited during the current session) are visualized with a solid border. (Note: this may change in future versions of Amadeus !). Links can be followed by double clicking inside the marked area. New links are created by choosing the desired link type with the toolbar buttons (rectangle, circle, ellipse), selecting the link area and calling the create link command (see The „Create Link“ Command, page 73).

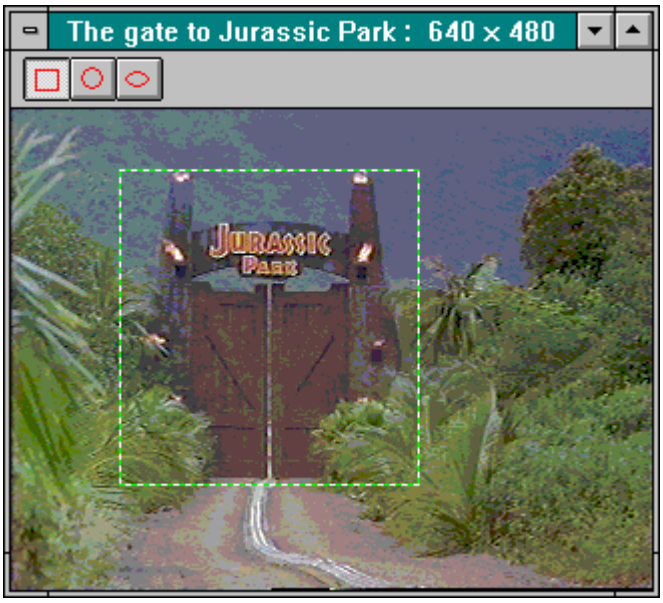

Figure 9 - The Amadeus image viewer

6.4.4 The PostScript viewer

- is used to display PostScript documents and works only if you installed it during setup using the ‘custom’ or ‘maximum’ option (see Installing AMADEUS, page 46)!

You may navigate through a postscript document of several pages either by entering the desired page number in the toolbar, going forward/backward, to the beginning/end using the toolbar buttons or by following links (areas marked by a colored rectangle) by double clicking on them.

Additionally you may alter the zooming factor by pressing (zoom in) or (zoom out).

If you press the right mouse button you will get a magnifying glass to temporarily enlarge interesting areas. (This behavior may slightly change in future versions of Amadeus because this functionality conflicts with the use of the right mouse button for context sensitive menus !)

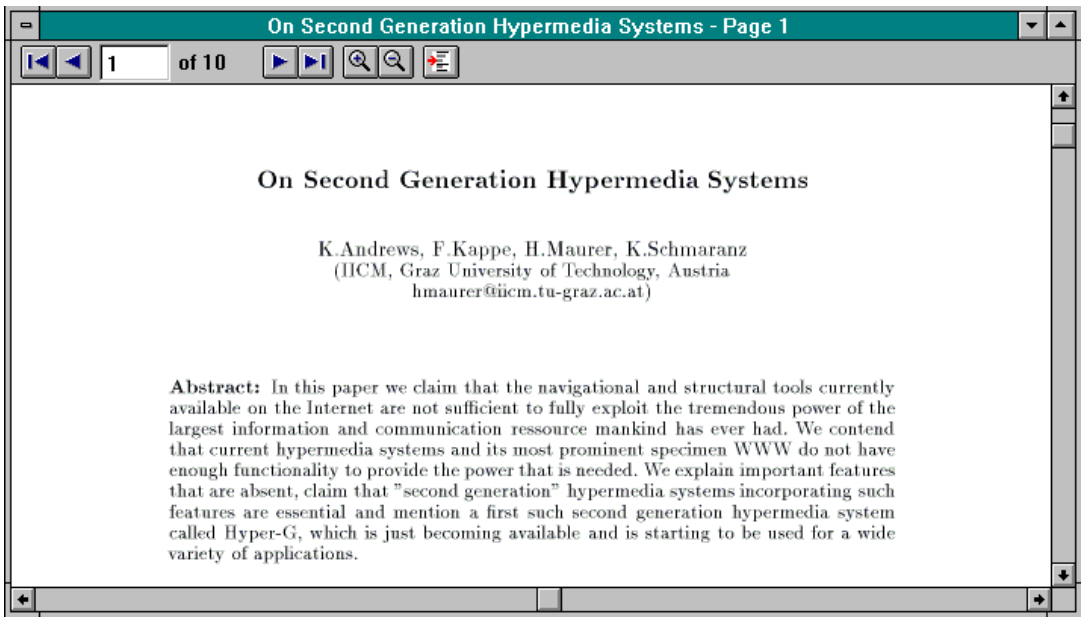

Figure 10 - The PostScript viewer

Note: You may also print out the postscript document even if you don't have a PostScript printer !

6.4.5 The movie viewer

- is used to display animations (technically ‘MPEG movies’) and is currently under construction.

You can start playing a movie with the ‘File/Play’ command (see The „Play“ command, page 70) and stop it with the ‘File/Stop’ command (see The „Stop“ command page 70).

Figure 11 - The Amadeus movie player

6.4.6 The sound player

- is used to play sounds. There is currently no special viewer directly build in Amadeus, instead you can configure an external viewer (see The „AMADEUS.INI“ file, page 100 on how to do this) or take the default ‘soundrec.exe’ which comes with every Windows system (Note: On Windows NT systems, it’s called differently (sndrec32.exe), so you have to copy this file to ‘soundrec.exe’ (located in the system32 directory) to make it work with Amadeus !)

6.5 Detailed description of Amadeus commands

6.5.1 The „File“ menu

6.5.1.1 Overview

These are the commands that are available in the File menu:

Commands	**Shortcut Key**	**Purpose**
New	Ctrl+N	creates Hyper-G documents (such as collections, clusters, texts, images, movies, ...)
Save	Ctrl+S	saves the active document (actually only working with text, image, postscript documents and the collection browser)
Save Anchors		saves source and destination anchors of active document (dumps out link attribute information)
Modify		modifies the active document (currently only working for text documents !)
Copy		copies the active document to a different collection
Move		moves the active document to a different collection
Unlink from collection		unlinks the active document from its collection
Delete		deletes the active document from the Hyper-G database.

Play		plays a multimedia document (if it is playable, e.g. a movie)
Stop		stops a multimedia document from playing
Print	Ctrl+P	prints active document
Page Setup		changes page layout settings
Print Preview		makes a print preview of the active document
Print Setup		calls the print setup dialog box
Exit	Alt+F4	closes Amadeus

6.5.1.2 The „New documents“ commands

The New command in the File menu allows you to insert different documents into the Hyper-G database. Before you choose this command you have to select where (in which cluster or collection) you want to insert the new document (using the collection browser). Then you may click on ‘New’ and a pull down menu will open, where you can decide which kind of document you want to insert. You can select between inserting:

- a text document,
- an image document
- a movie document
- a sound document
- a 3D-scene document
- a Postscript document
- a generic document
- a remote connection

- a new collection
- a new cluster
- an annotation

After that one of four dialogue boxes will appear. (The commands ***Image***, ***Movie***, ***Sound***, ***Scene***, ***Postscript*** and ***Generic*** will always call the same dialog box, see Figure 12, just the dialog box title changes.)

What they have all in common is that you have to specify a title and you may apply access rights to the document.

Access Rights:

Currently there exits ‘read access rights’, ‘write access rights’ and ‘unlink access rights’.

The syntax is:

< read access rights >[;< write access rights >][;< unlink access rights >]

whereas

< read access rights >: **R:<*spec*>{,<*spec*>}**

< write access rights >: **W:<*spec*>{,<*spec*>}**

< unlink access rights >: **U:<*spec*>{,<*spec*>}**

<spec>:

a ... only the Author

u user{ user2} ... one or more users

g group{ group} ... one or more usergroups

By default everybody may read your document, write access is limited to the owner of the document and to system administrators.

Unlink access right: If you add this access right only **the specified user(s)** may delete a document (users with the write access right may only add new, but not delete other documents which they do not own !)

Examples:

‘R: g iicm; W: u tdieting; U: a’

Means: The author, the user ‘tdieting’ and all members of the ‘iicm’ group may read the document, the author and the user tdieting may modify it, but only the author may delete it.

‘R: a’

Here, only the author (and the system administrators) may access this document.

6.5.1.3 Inserting an Image, a Movie, a Sound, a Scene, a Postscript or a Generic document:

In the right part of this dialog box you will find a small file chooser. Use it to select the document you want to insert into the database. On the left side you have to enter some information about the document. Author, creation time and language will be taken from the present settings but can also be changed. In the first input line you have to insert the title of the document and in the input line called ‘Rights’ you can specify the access rights (see ‘Access Rights:’).

If you select ‘*Generic*’ in the ‘New’ menu, then two additional fields will be enabled: „***subtyp“***‘ and „***arguments***“.

At ‘subtype’ you write the generic user type, e.g. ‘excel’ if you want to insert an Excel-spreadsheet document into the database. At ‘Arguments' you type now the true file-name of your Excel file (e.g., 'budget94.xls' , you can use the file chooser on the right to select the file and then copy and paste the file name to this field), without a path in this field. (Note: You may also specify a different filename in the argument-entry, but that's the name your document gets, when it is downloaded)

In this example we don't need any further arguments, thus we click on 'OK' and the document will be uploaded to the database (I suppose that you have also filled out correctly the rest of the entries!)

Ok, now we go to the 'Amadeus.ini' file and look what we have to do there (this has to be done only once for every new generic document type):

Use your preferred editor to open amadeus.ini and scroll down to the [Generic] section. Then make a new line beginning with the text you specified in the 'Subtype' entry before, in our case 'excel' (Note: It MUST NOT be different in any way, otherwise it will not work) and place a '=' afterwards. Then you write the complete path of your Excel-version on your PC followed by '$arg0'.

Now it should read like this: (for example)

[Generic]

excel=c:\excel\excel.exe $arg0

Note: There is also a short description of the syntax in the ini-file.

Ok, you've got it - that's all !

If you want to upload additional Excel files to the database be sure to specify always 'excel' in the 'Subtype'-field !

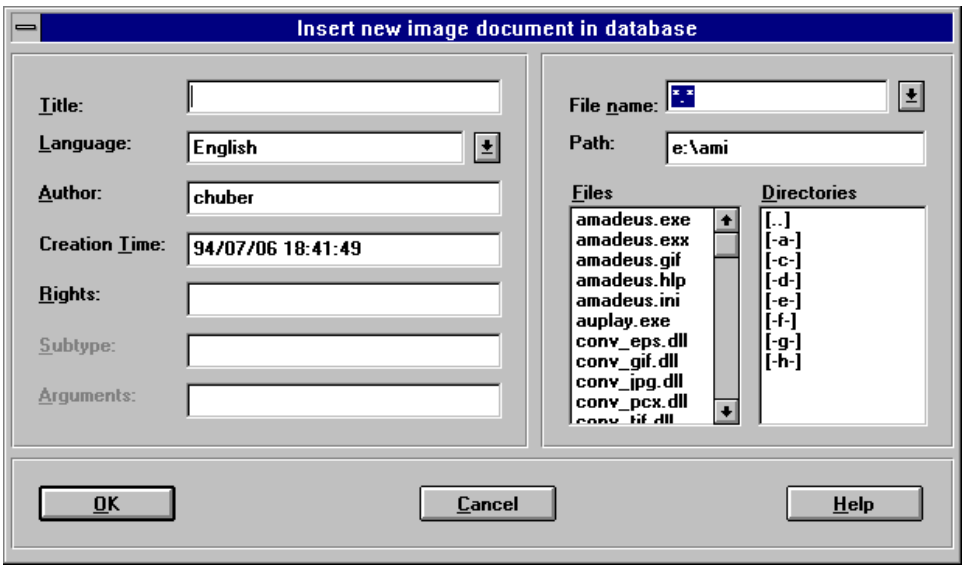

Figure 12 - The 'Insert new image document in database' dialog

6.5.1.4 Inserting a Text document:

If you select the ***Text*** command a dialog box will open where you have to enter the collection name, the title of the document, choose the document in the file chooser and set the correct language and the correct type of object (RTF, HTF or TXT). You may also apply an expiration date, that specifies when a document is no longer visible in the database.

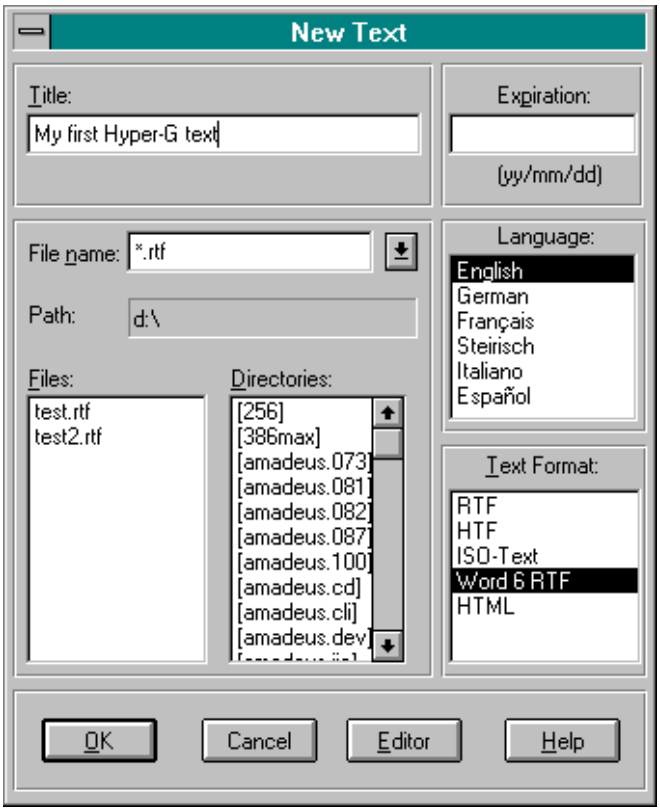

Figure 13 - The 'New text' dialog

Hint: Having a collection or cluster selected as current object (e.g. by positioning the cursor on the parent collection in list mode) the name of this collection is taken as default. Otherwise, you would have to find the name or the ObjectID of the parent collection with the info command.

Note: Specifying the correct language is important for the display of the title and for searching!

6.5.1.5 Inserting a new collection or a new cluster:

When you insert a new ***collection*** or a new ***cluster*** you get nearly the same dialog box. Title, language, author, creation time and access rights have to be filled out by the Amadeus user and appear in both dialog boxes. Description and unique name are two more input lines which appear when you select the 'new collection' command.

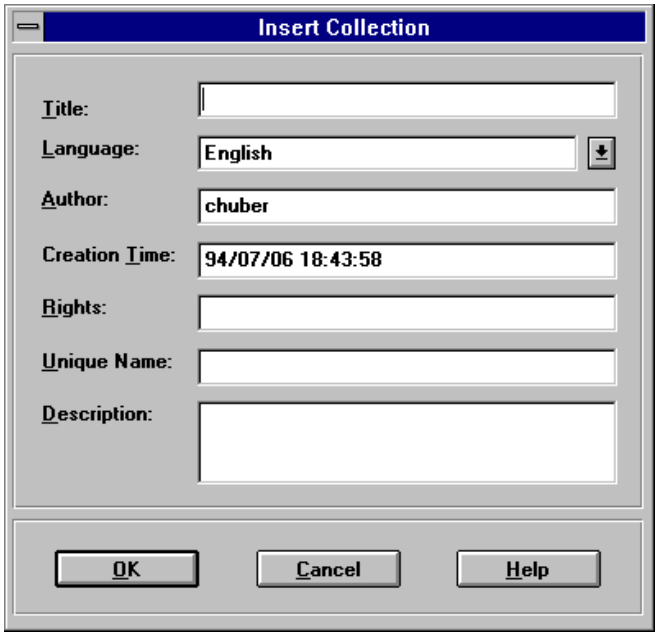

Figure 14 - The 'Insert collection' dialog

6.5.1.6 The „New Remote" command

With this command you can insert a special document which provides links to other remote databases (host field).

Valid types are (currently): Telnet, Gopher, http (WWW), FTP and WAIS. Additionally you can also specify a port number and a startup directory (Gopher, http), a path (FTP), a user (Telnet) or the database name (WAIS).

Of course you also have the possibility to specify the usual Hyper-G document attributes like title (should always be added), title language, author, creation time (set to server time, if left empty) and access rights (see Access Rights: page 60)

Note: additional attributes (e.g. a second title) can easily be added using ‘Edit/Properties’ (see The „Properties“ Command, page 74)

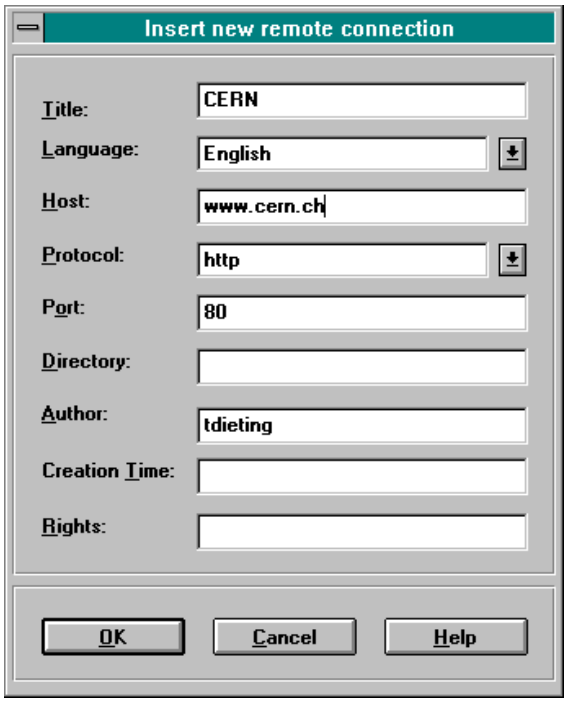

Figure 15 - 'Insert new remote connection' dialog

6.5.1.7 The „New Annotation“ command

This command allows to interactively enter a (possibly public) annotation to a certain document. You need to have selected the document to be annotated as current document (e.g. by reading it in or having positioned the cursor on it in the collection browser).

After entering the command, a dialog box similar to the ‘new text’ command (see Figure 13, page 63) will come up that asks you about the name of the parent collection of the new text (defaults to the parents collection of the document to be annotated, if only one) and the title of the annotation. Then you have to enter the language of your comment, defaulting to the language of the document to be annotated.

Note: Specifying the correct language is important for the display of the title and for searching!

Then you may click on the ‘Edit’ button to enter a few lines with the simple built-in editor or select a prepared file with the file chooser.

Note: Be sure to specify the correct text format: E.g. ‘RTF’, if you want to insert an RTF-document, ‘Txt’ if you want to insert plain ISO-text (e.g. from EMail, or if you use the built-in editor because you don’t know anything about the HTF-Format !).

If you open the text-editor it will contain a few lines that were generated automatically, including a link specification pointing to the document to be annotated. Do not edit these lines!

It is possible to insert arbitrary links into the text. The description of the make link command tells you how.

6.5.1.8 The „Save“ command

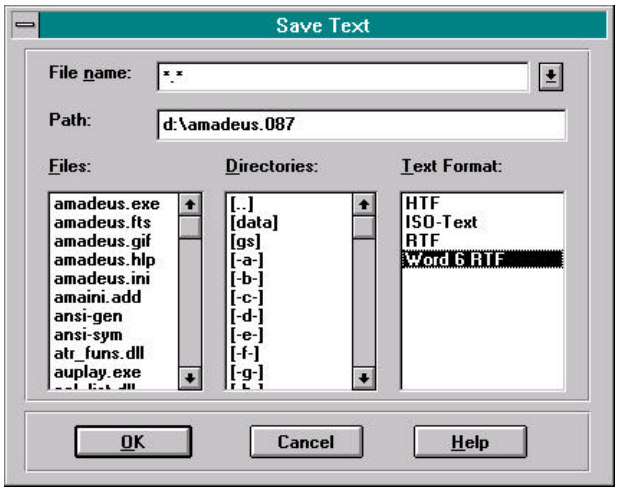

Figure 16 - The ‘Save text’ dialog box

With this command you can save the selected document to disk. Currently only texts, images and postscript documents are supported. (See Figure 16 as a sample dialog for saving a text document)

6.5.1.9 The “Save Anchors” command

This command is only for internal purpose. E.g.: In addition to a document you may save all attached anchors. If you have a stand-alone Hyper-G viewer, you may also follow internal links using this viewers. An example is the stand alone PostScript viewer, released very soon.

6.5.1.10 The „Modify“ command

Currently, only modifying of text document (in the original HTF-style) is available. There is a very simple text editor built in that allows you to change the content of the document.

Note: You have to be identified and have write access rights to modify the document !

6.5.1.11 The „Copy“ command

This allows to copy the current object (document or collection) to another collection/cluster (a Hyper-G object may be member of a number of collections). You will get a dialog box which asks you for the source object (initially the currently active object) and the destination collection. Use the collection browser to move to the desired object and then click the ‘source’/’destination’ button to fix your selection (see Figure 17).

The only restriction is that no circles in the collection hierarchy can be made (e.g. collection A being a member of B being a member of A).

Note: The object is not physically copied (thus resulting in requiring additional space in the document server), instead only a new link from the specified collection/cluster to the object is generated.

You can use this function to generate ‘bookmarks’ in your home collection.

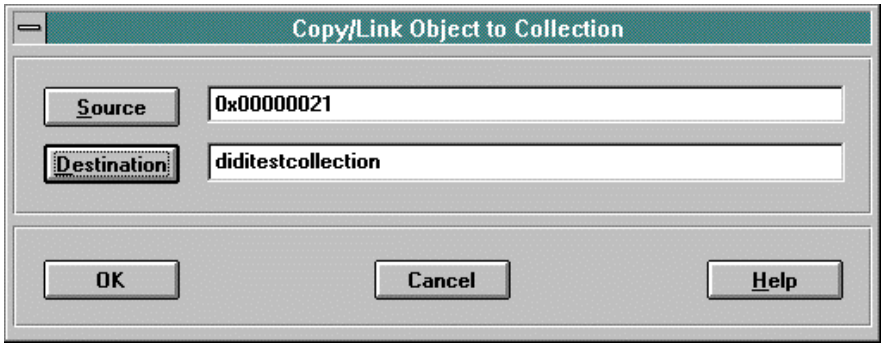

Figure 17 - The copy/link object to collection dialog

6.5.1.12 The „Move“ command

This command moves the selected object to a different collection. After the command has been entered the viewer prompts you with a dialog box for the target collection (similar the ‘copy object’ command, see Figure 17). This can be specified by either directly entering the collection name or the collections ObjectID (which can be found with the ‘Edit/Attributes’ command), or by selecting the target collection with the collection browser.

You must have write permissions for both the target collection and the collection which contains the object you are moving. Otherwise the message *** Error: Access denied *** will appear.

The message Database modified is displayed if the command completed successfully. The message *** found nothing *** indicates that the target collection was incorrectly specified.

6.5.1.13 The „Unlink“ command

With this command you can unlink a selected object from a collection. In this case it is **not** physically deleted if there is more than one parent referring to it, only the link between the parent collection and the object is removed. (Thus you may still find the object using search, or following a link to this object from a different collection)

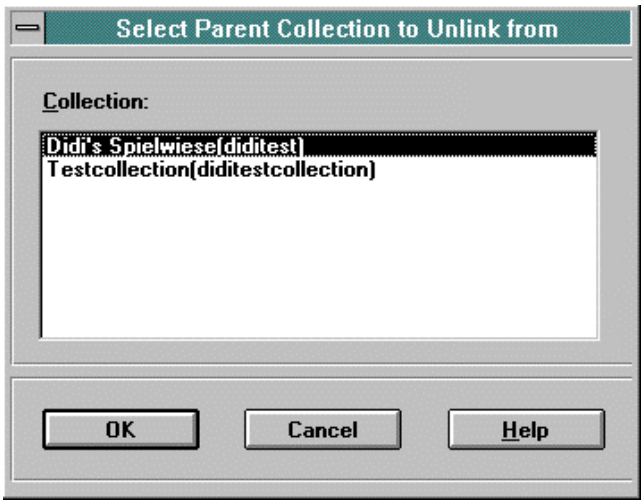

Figure 18 - The ‘select parent collection’ dialog

If there are more than one parent collections, you will get a dialog box (see Figure 18) which asks you to select a particular collection to unlink from.

Note: If the object is going to be physically deleted, you will get a warning informing you about this operation !

6.5.1.14 The „Delete“ command

This command can be used to physically delete the selected object (the active text document or the highlighted object in the collection browser, if this one is active) from the Hyper-G database. You can only delete objects if you have write permission on the object. Naturally, you will have to identify yourself before you issue this command.

Note: this command erases the current object from all of its father collections. In order to remove it only from specific father collections use the 'unlink from coll.'-command (this is much safer !)

6.5.1.15 The „Play“ command

This command plays a multimedia document that is handled by an internal viewer. (If it’s playable, such as movie.) Currently only MPEG movies are supported by this command.

6.5.1.16 The „Stop“ command

This command stops a multimedia document (that is handled by an internal viewer) from playing. (If it’s playable, such as movie.) Currently only MPEG movies are supported by this command.

6.5.1.17 The „Print“ command

Use this command to print a document. This command presents a Print dialog box, where you may specify the range of pages to be printed, the number of copies, the destination printer, and other printer setup options.

Shortcuts

Toolbar: 

Keys: CTRL+P

6.5.1.18 The „Page Setup“ command

Use this command to change the page layout settings.

6.5.1.19 The „Print Preview“ command

Use this command to display the active document as it would appear when printed. When you choose this command, the main window will be replaced with a print preview window in which one or two pages will be displayed in their printed format. The print preview toolbar offers you options to view either one or two pages at a time; move back and forth through the document; zoom in and out of pages; and initiate a print job.

6.5.1.20 The „Print Setup“ command

Use this command to select a printer and a printer connection. This command presents a Print Setup dialog box, where you specify the printer and its connection.

6.5.1.21 The „Exit“ command

With this command you can quit ‘Amadeus’

6.5.2 The „Edit“ menu

6.5.2.1 Overview

These commands are available in the Edit menu:

Commands	**Shortcut Key**	**Purpose**
Cut	Ctrl+X	cuts parts of a document into the clipboard. (not yet implemented !)
Copy	Ctrl+C	copies parts of a document into the clipboard.
Paste	Ctrl+V	inserts clipboard content into the document. (not yet implemented !)
Create Link	Ctrl+L	inserts a link into the selected document (currently only working for text documents)
Edit Link		edits several attributes of the link (not yet implemented !)
Delete Link		delete the currently selected link
Link Properties		gets/edits properties of selected anchor
Properties		gets/edits properties of selected document

6.5.2.2 The „Cut“ command

Use this command to remove the currently selected data from the document and put it on the clipboard. This command is not available if there is no currently selected data.

Cutting data to the clipboard replaces the contents previously stored there.

Shortcuts

Toolbar: 

Keys: CTRL+X

6.5.2.3 The „Copy“ command

Use this command to copy selected data onto the clipboard. This command is not available if there is no currently selected data.

Copying data to the clipboard replaces the contents previously stored there.

Shortcuts

Toolbar: 

Keys: CTRL+C

6.5.2.4 The „Paste“ command

Use this command to insert a copy of the clipboard contents at the insertion point. This command is unavailable if the clipboard is empty.

Shortcuts

Toolbar: 

Keys: CTRL+V

6.5.2.5 The „Create Link“ Command

If you have write permission to a text document you will be able to insert clickable links into it. After you called this command a dialog-box will appear (see Figure 19 - The Create Link dialog). Then you may select the word (in the text document) you want to make a link to by double-clicking on it. The word will get marked. Now click on the „Source“-button in the dialog box to fix the „*source-anchor*“. Next to that you have to specify what will happen when the user clicks on this link. Thus you have to specify the „*destination anchor*“ by selecting the object (e.g. in the collection browser) and then clicking on the „Destination“-button.

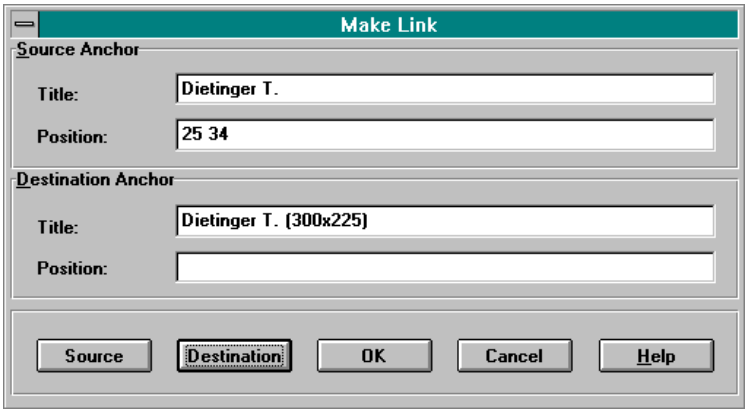

Figure 19 - The Create Link dialog

Finally you press the „Ok“-button to confirm your link creation or the „Cancel“-button to abort the operation.

Note: If you don’t select a word, the current cursor position will be used to insert an inline image (you specify with destination) there !

6.5.2.6 The „Delete Link“ Command

With this command you can physically delete the selected link.

6.5.2.7 The “Link Properties” Command

See 6.5.2.8 (The „Properties“ Command) for a detailed description of properties.

6.5.2.8 The „Properties“ Command

This command provides detailed information about the currently selected object (more than fits into the title line) For the "ordinary" user this information is not of extreme importance, however. The number and kind of attribute names and values depends on the type of the object.

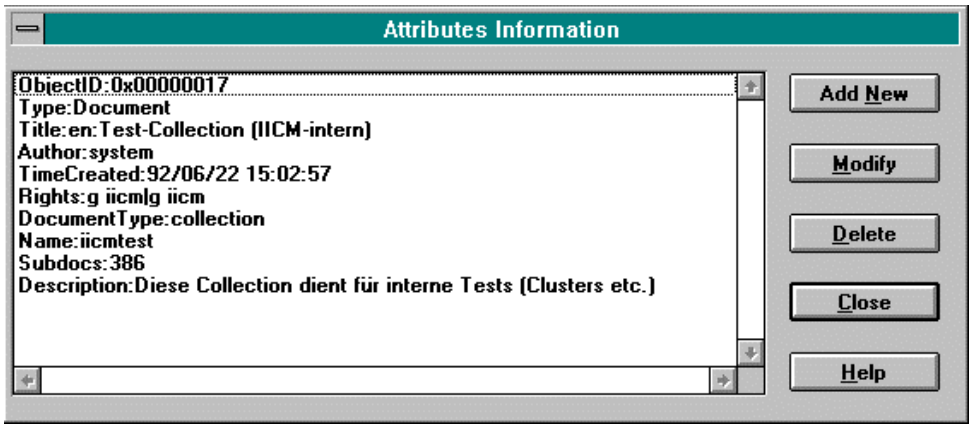

Figure 20 - The ‘Attributes information’ dialog box

Some of the attribute fields displayed by the property command may be changed using this command, provided that you have write permission for the current object. It is rather obvious that this command should be used with care.

Adding/Modifying attributes

If you click on the „Add New“-button you will get a new dialog box (see Figure 21) that allows you to add a new attribute type (out of the list in the type-field) to the current document.

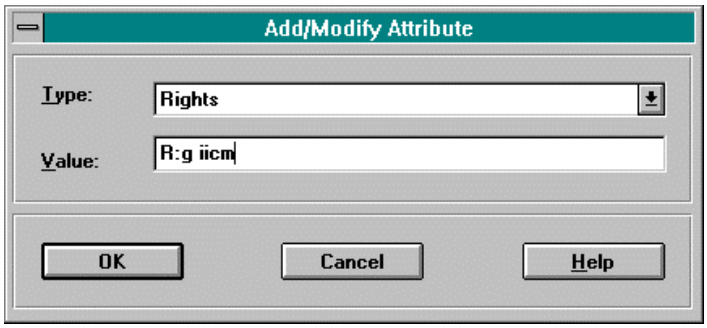

Figure 21 - The ‘Adding/Modifying’ dialog box

The same dialog box will also appear if you click the „Modify“-button or double-click on one attribute in the information dialog box and enables you to modify the value of an attribute.

Note: Usually you use this command to add/modify access rights or titles/descriptions.

E.g. By adding a second title to a multimedia document, you can make it „*language independent*“.

Explanation of „language independent/dependent“: Suppose you want to make a cluster with two sound documents with the same contents but in two different languages (e.g. a spoken explanation). All you have to do is to specify the titles of the sound documents in the respective language.

The algorithm to ensure correct viewing of these documents works like this: If a cluster contains more than one documents of a certain type (e.g.: sound) with only ONE title only the document that matches the desired language best will be displayed. Documents that have more than one title (= language independent) will be displayed in any case (and in addition to the language dependent documents). If you want to show several language dependent documents (of the same type) in a cluster, you have to make „*sub-clusters*“. (E.g.: If you want support two languages you will create a cluster that contains two text documents in different languages

and two sub-clusters, each containing two language dependent images. The viewing-result will be a text and two images in the correct language.)

Deleting of attributes

If you have write access for this document you may also delete certain attributes (where it makes sense, e.g. you may not delete the document type) by clicking on the „Delete“-button in the Attributes Information dialog box (see Figure 20).

A description of the meaning of attributes of Hyper-G objects can be found in the documentation on the Hyper-G Data Model. In particular, it contains a description on how to specify access rights.

6.5.3 The „Navigation“ menu

6.5.3.1 Overview

These are the commands that are available in the Navigation menu:

Commands	**Shortcut Key**	**Purpose**
History		shows a list of previously executed commands
Forward		navigates forward in the history list
Back	Ctrl+Z	navigates back in the history list
Home		jumps to home page
Hyperroot		shows all Hyper-G servers all over the world
Goto		goes to a specified collection or stored bookmark
Set Bookmark		stores active collection in bookmark list
Children	F12	shows children of current document
Parents	F11	shows parents of current document

Links		shows links of current document
References		shows references of current document
Annotations		shows annotations of current document
Break		stops each action

6.5.3.2 The „History“ command

This command displays a list of recently issued commands. You may select any command from this list and re-execute it.

Technical description of commands "history", "back" and "forward"

Whenever a viewer command is executed, it is appended to the history list, which is organized as a stack.

- The Command "back" (or simply BACKSPACE) removes the top element from the stack (but remembers it in a second one) and re-executes the new top-level command of the stack.
- In contrast, the "forward" command removes the top-level command from the second stack, puts it on the history stack again, and re-executes it.
- Execution of an ordinary command clears the second stack and puts the command on top of the history stack. Commands which cannot be undone are not stored on the stack.
- The "history" command displays both stacks (with elements from the second stack marked with '*').

In most cases, repeating an old command works faster the second time because of caching in the client.

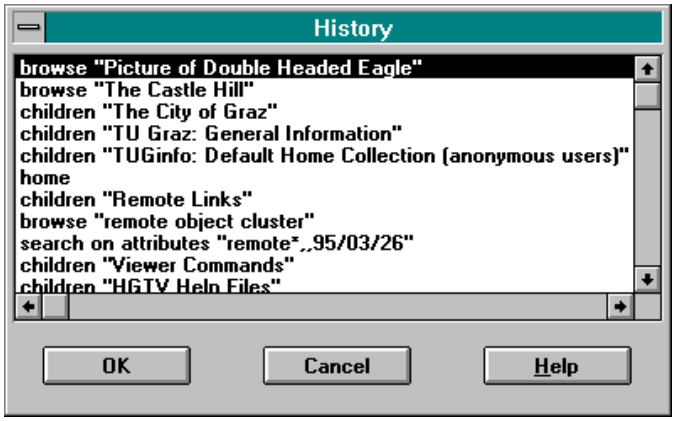

Figure 22 - The 'History' dialog box

6.5.3.3 The „Forward" and „Back" commands

With these two commands you can move forwards and backwards in the history list (thus executing a previous command).

6.5.3.4 The „Home" command

This command puts you to your home collection. If you are an anonymous user, or if your system administrator has not created a personal home collection for you, you will see the "anonymous root collection" (a collection that is named either ~anonymous or rootcollection).

The home command is automatically executed when the terminal viewer is started.

You may use the parents command to go "up" from this point, especially if you want to insert a new document into the home (root) collection.

As a Hyper-G system administrator, you may use the hgadmin (currently only on UNIX-platforms supported) command to supply users with a personal home collection.

6.5.3.5 The „Hyperroot command

This command lists you all Hyper-G Servers all over the world in the collection browser. You can connect to one of the servers by selecting its menu line in the browser.

6.5.3.6 The „Goto“ command

Currently this command allows you only to jump directly to a collection by specifying the unique collection name (not the title) and a remote connection by specifying the path.

Collection name:

The command will prompt you for the name of the collection. The name may be abbreviated (if the prefix is unique) with an asterisk. You will then see the children of this collection, or an error message if it is empty.

Note: The (unique) name of a collection can be found by applying the info command to it.

Remote connection:

The format is: <protocol>://<host>{:port}/path

where <protocol> is one of the following:

'hyperg:', 'gopher:', 'http:', 'wais:', 'telnet:', 'ftp:'

E.g.: http://www.tu-graz.ac.at/

gopher://gopher.tu-graz.ac.at/1~anonymous

gopher://gopher.bmwf.gv.at

For 'hyperg':hyperg://<host>{:port{:<binary>{:<distributed>}}}/path

E.g.: hyperg://hyperg.tu-graz.ac.at:418;0;1/~anonymous

6.5.3.7 The „Children“ and „Parents“ commands

With ‘parents’ you get the previous object (according to the hierarchical structure of Hyper-G), with ‘children’ all objects contained in the selected collection/cluster.

Note: The ‘Parents’ command is the only way of getting to the very top of the document hierarchy to add items into the root collection.

6.5.3.8 The „Links“, „References“ and ‘Annotations’ commands

‘Links’ displays a list of all links emanating from the current object (more precisely, the destination documents of the links), sorted according to the current sort options.

The call also works on WorldWideWeb documents, however, only the URL (Universal Resource Locator) and not the title of the target document can be shown.

‘References’ is the opposite of the links command, so to say. It answers the question: "What other documents reference to the current document"?

The result of this command is a list of all documents that have a link pointing to the current document.

‘Annotations’ shows a list of annotations referring to the selected document.

6.5.3.9 The „Break“ command

This command allows to interrupt some lengthy operations like file loading etc...

6.5.4 The „Search“ menu

6.5.4.1 Overview

These are the commands that are available in the Search menu:

Commands	Shortcut Key	Purpose
Find	Alt+F3	searches for documents in the database
Find in text		searches for strings in the active text document
Activate		includes this collection in the search list
Deactivate		deletes this collection from the search list
Show list		shows current search list
Sort order		changes sort order of search result list
Full text option		changes some fulltext query options

6.5.4.2 The „Find“ command

One of the most important functions in Hyper-G. Currently you have three different possibilities **how** (search type) to search and two **where** to search in the database.

The search region

You may search in the ‘Whole Database’ or restrict the search to ‘Activated Collections’ (see *The „Activate“* and „Deactivate“ command, page 87)

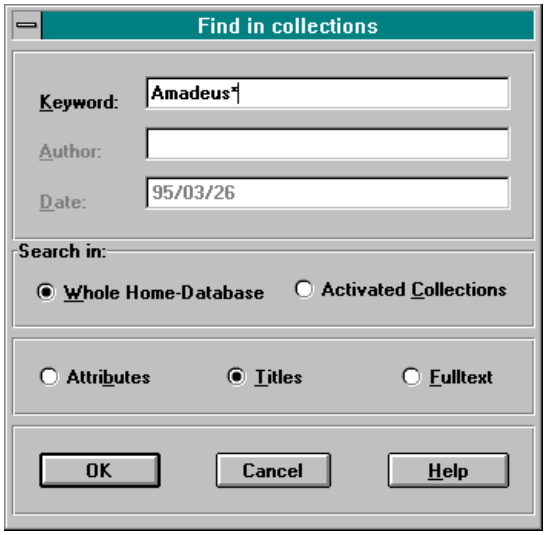

Figure 23 - The ‘Find in collections’ dialog

Search type Titles

With this search type you may perform Boolean searches in the Hyper-G information network.

Note 1: This is not a full text search. Rather, an object is found when the keywords entered appear in the title of the object or have been attached to it as additional keywords. When used on a WAIS source or Gopher index, the exact effect is determined by the remote database implementation (but in most cases it will default to a full text query).

Ordinary Search

For example, entering 'foo' as a keyword will find all objects (documents, collections, destination anchors) with the word 'foo' in the title or as a keyword. Upper- and lowercase are ignored. National character sets can be entered as 8-bit codes (ISO Latin 1) or in SGML notation.

The output will be a list of matching objects, sorted by score (for WAIS databases, first parent collection, and title in reverse order. The name of the first parent collection is displayed instead of date and author in list mode.

When entering more than one word, they are ANDed (in other words, a blank is the AND operator). E.g., 'foo bar' will match only objects that contain both 'foo' and 'bar'. The OR operator is '|'. E.g. 'foo|bar' will match all objects that have either 'foo' or 'bar'.

You may use the following operators (and combinations of them):

- & or && or ‘AND’
- | or || or ‘OR’
- ! or ‘NOT’
- &! or ‘ANDNOT’

If you want to look for a search query that is an operator, you have to put it under quotes: e.g: mother “and” father would search for titles containing ‘mother’ and ‘and’ and ‘father’.

Prefix Search

E.g. 'foo*' would match everything that contains a word that starts with 'foo' (like foobar). You may use AND and OR operators like above (like in 'uni* california).

Due to a (deliberate) limitation in the server, you cannot use '*' to match all objects.

Search type Attributes

This search type offers the full functionality of the search type ‘Titles’. In addition, you may restrict the search to certain authors and objects inserted into the database after a certain date.

Note: Unlike the search titles command this command cannot be used to search through remote databases (WAISes and Gopher Indices).

An Example: To find all document of author "foo" in the set of active collections that have been created after April 1, 1993, you would enter:

Keyword: (empty)

Author: foo

Date: 93/04/01

At most 2 out of the 3 fields can be left empty. As a special case, if the first two fields are left blank, the system will offer the current date as a default. This is a convenient way to search for the objects that have been created today.

Search type Fulltext

This search type allows you to perform searches on the contents of text documents stored in the Hyper-G server. All text documents inserted into the server are automatically indexed for full text search.

Technically speaking, the Hyper-G full text engine is able to perform a combination of ranked, Boolean, weighted, fuzzy, and nearest-neighbor searches on the documents (see the formal syntax description at the end of this text). The following examples explain the meaning of these terminis a more intuitive fashion.

Examples:

„computer“ (primitive search)

When you enter only one word, you will receive a ranked list of the documents in the set of active collections that contain this word. The list will be truncated after 40 matches (this number can be changed with the Full Text Option command), and normalized such that the top-ranking document gets a Score value of 100%. The ranking is computed based on the number of times the word appears in the document, where it appears (title, headings, and subheadings count more), and the size of the document.

As the Hyper-G full text engine employs a so-called *stemmer*, the above example would also match documents containing "computers", "computing", "compute", and the like.

There are certain words (so-called *stopwords*) that usually appear very often (e.g., "is", "was") that you cannot search for, as they are not indexed. Whether you use uppercase or lowercase characters is irrelevant.

„tell me about computer conferences“ (nearest neighbor)

You may also enter a search string like this example. The first 3 words are stopwords and are ignored. The system will search for documents that contain most of the other (2) words, but usually will also deliver those that contain a high frequency of one of the words only.

„computer & conference“ (Boolean search)

Sometimes you want to specify that the documents really contain all the search terms, like the example above. The & operator means "AND" (you may also use &&), while | means "OR" (you may also use ||), ! means “NOT” and &! means “ANDNOT”. You may of course combine these operators to more complex constructs, like e.g. "(personal & computer) | (ibm & pc) | macintosh | amiga" to match a variety of personal computers. In this example, the brackets are not really necessary as the AND operator has precedence over OR.

Note: You may also write the keywords (OR, AND, NOT, ANDNOT instead of the operators)

„computer &[f] conferencing“ (fuzzy Boolean search)

The 'f' makes the AND operator a fuzzy AND, i.e. it will also match documents that contain only one of "computer" and "conferencing", but with lower ranking.

„computer &[f] conferencing{3.5}“ (weighted Boolean search)

Similar to the example above, but allows you to make "conferencing" 3.5 times as important as "computer" in the ranking.

Note

Your may combine above examples to arbitrary complex queries. Most of them are not particularly useful, however.

Some of the features (e.g. stemming and stoplists) are of course language dependent. While ,in principle, the Hyper-G indexer allows to index words in any language that is based on the ISO Latin-1 alphabet, stoplist and stemmer are currently available for English only.

Query Syntax

This is a more formal description of the syntax of full text queries, where expressions in '[]' are optional, '{}' means 0 or more occurrences, and '|' means "or":

```
expr       ::= term { orop term }
term       ::= factor { andop factor }
factor     ::= node ['{'float'}']
node       ::= word { word } |  '('expr')'

orop       ::= '||' | '|'
andop      ::= '&&' [ optionlist ]  |  '&' [ optionlist ]

optionlist ::= '[' option { ',' option } ']'
option     ::= 'F' | 'f'
```

The optionlist is extensible, to foresee proximity searches in the future.

6.5.4.3 The „Find in text“ command

With this command you may find a string in the active text browser. Currently only exact matches are found (no query syntax).

6.5.4.4 The „Activate“ and „Deactivate“ command

Use this commands to include/activate or exclude/deactivate a collection from the search list. It only works with ordinary Hyper-G collections (), WAIS sources () and gopher indices (). Activated collections are visualized with a '  ' in front of in the collection browser.

If you want to restrict the search scope with ‘search in Activated collection’(see Figure 24) using the ‘Search’ command, you need to activate at least one collection first. However, there is a little shortcut: When using the above command on a collection, WAIS server or gopher index (e.g. when the cursor points to it in list mode), the corresponding collection is temporarily activated, the search is performed, and it is deactivated afterwards.

Important: Searches are performed on all subtrees of the activated collection(s), recursively (except for remote collections and remote databases; they have to be activated "by hand"). In order to activate the root collection (i.e. the whole Hyper-G database), you may perform

the ‘Home’ command,

the ‘Parents’ command and

the ‘Activate’ command

Note: You may also use the toolbar buttons  to activate, and  to deactivate a collection from the search list.

6.5.4.5 The „Show list“ command

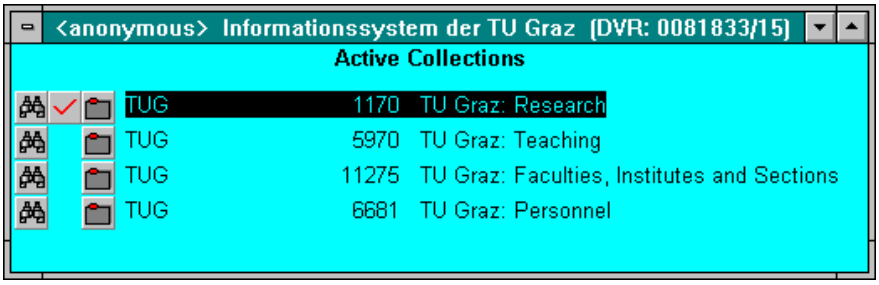

Figure 24 - The list of activated collections in the collectionbrowser

Displays a list of all activated collections (including remote WAISes and Gopher indices). Useful to deactivate some of them.

6.5.4.6 The „Sort Order“ command

Use this command to change the sort order of collections browser items (e.g. the search result list after a search).

The sort order is defined by sorting on the object's attributes. You will be prompted to enter a string composed of the following characters:

- **-**: on 1st position: sort descending (default ascending)
- **#**: Sequence Number
- **A**: Author
- **C**: Creation time
- **E**: Expiration time
- **O**: Opening time
- **P**: Parent (after search only)
- **S**: Score (after full text search only)

T: Title

t: Type

For example, „**AT**“ means sort by author first, and then (if authors are equal) by title.

The default sort order is "#T", which means look at a sequence number first (the Sequence attribute, if specified), and then by title.

After a search, however, the default sort order is "-SPT", which means sort by descending score value (after fulltext search or WAIS search), then by parent collection, then by title.

Bugs: The "descending" flag works for all sort attributes. It is not possible to specify it for individual attributes.

6.5.4.7 The „Full Text Option“ command

This command lets you set the maximum number of items that is returned by the search full text command, and by using the search titles command on WAIS sources. The default value is 40, which means that only the 40 "best-matching" items are displayed.

6.5.5 The „Administration“ menu

6.5.5.1 Overview

These are the commands that are available in the Administration menu:

Commands	**Shortcut Key**	**Purpose**
Identify		identifies the user using a password
Change		change current password

password		
Status		shows current system status
Send Message		sends a message to one or all active users

6.5.5.2 The „Identify“ command

Use this command to identify yourself to the Hyper-G server. After successful identification (the user name is displayed in the title bar of the collection browser) you may modify part of the information network, add documents, etc., depending on your access rights.

The identification is done by entering your user name plus a password. In order to receive a Hyper-G account, you have to contact the Hyper-G operator(s) of your organization.

6.5.5.3 The „Change password“ command

This command allows you to change the password associated with your Hyper-G user record. You will have to enter the old password, and the new one (twice). Of course, you have to be identified to use this command.

6.5.5.4 The „Status“ command

This command allows you to retrieve information about the Hyper-G server you are connected to, including performance statistics and the user currently connected to it.

The command displays the protocol version of the client (Px) as well as the connection (Cx), and a copyright message in the top right corner of the screen. Following are the identification of the server, the local server time, the time it was started and the time that has elapsed since then. The next line contains a performance (or load) measure, i.e. the number of objects per second that have been retrieved from the low-level database. An average value is given for the last minute, the last 15 minutes, the last hour, and since the server was started. What

follows is a list of users that are currently connected to this server, with yourself marked by an asterisk in the first column. The list contains the user number, the user's name, the host of the client, when the session was started, and the user's idle time (i.e. the time that has elapsed since the last transaction with this server).

Note that some Hyper-G servers will close the connection after a certain period of inactivity (usually 12 hours, but could be significantly less).

6.5.5.5 The „Send Message“ command

The command allows you to send a message to all or to a special user (specified by his user number). Use the ‘status command’ in order to find the user number which belongs to the user.

After the message has been received the other user can reply to your message easily.

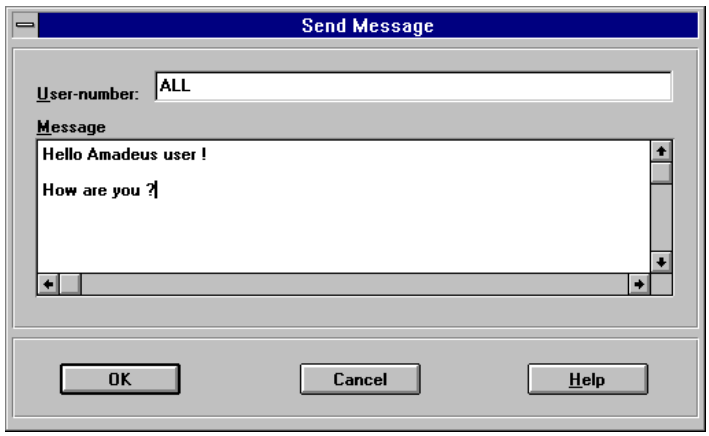

Figure 25 - The ‘Send message’ dialog box

Note: If you are a system administrator you may also specify „ALL“ instead of a user number, so that all users that are currently logged in the system will receive your message !

6.5.6 The „Options“ menu

6.5.6.1 Overview

These are the commands that are available in the Options menu:

Commands	Shortcut Key	Purpose
Language		changes preferred language
Fonts		changes various fonts (e.g. textbrowser)
Colors		changes various colors (e.g. background)
Viewer Options		configures some viewer specific options
Toolbar		enables or disables toolbar
Statusbar		enables or disables statusbar
Save Settings on Exit		enables or disables saving of changed configuration settings
Save Settings Now		saves changed configuration settings now

6.5.6.2 The „Language“ commands

With this command you can set your preferred language. Then the user interface texts, the titles of documents as well as the documents themselves are displayed in your selected language, if available.

6.5.6.3 The „Fonts“ commands

With this command you can change the fonts (and their attributes) of the fonts used in the collection browser (currently only one) and the text browser.

6.5.6.4 The „Colors“ commands

Here you can change the background colors of the Amadeus desktop, of the collections browser and of the text viewer.

Note: The text background in the collection and text browser must be a solid color. If you select a dithered color, Amadeus will automatically select the nearest solid color, thus preventing unreadable text.

6.5.6.5 The „Viewer Options“ commands

With this command you can change various settings that affect the appearance and behavior of the viewers directly implemented in Amadeus.

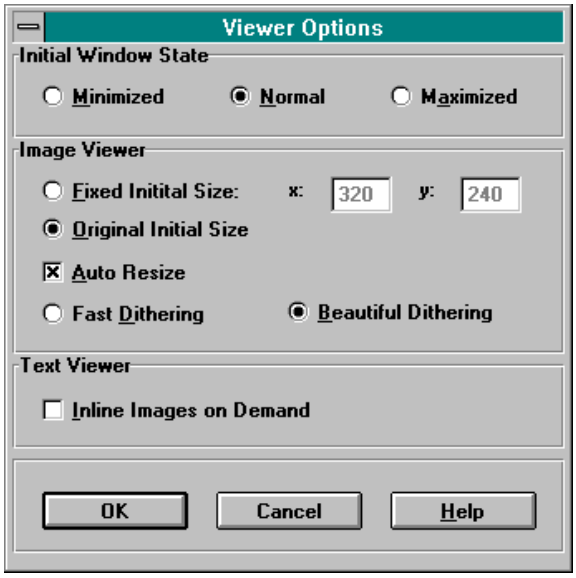

Figure 26 - The ‘Viewer options’ dialog box

Initial Window State

This influences how a multimedia document (image, movie, sound, postscript, ..) will be opened after download. Possible options are:

- Minimized: The document will be shown as an icon
- Normal: The document will be immediately opened after download
- Maximized: The document will be immediately opened with a maximized window after download

Image Viewer

- Fixed Initial Size: The image viewer will display images with a fixed size no matter what the real size of the image might be. This should prevent displaying very huge images with their full size. You may also specify the fixed size in the two edit boxes for x (width) and y (height).
- Original Initial Size: If you check this radio button, the image will be shown in its original size when you open the window.

- Auto Resize: If this option is checked, images will be automatically resized if you change the size of the window.
- Fast Dithering: On 16 or 256 color displays images must be ‘dithered’(technique to mix colors) to show all colors. If this option is enabled the calculation will be quite fast but the quality will not be very high.
- Beautiful Dithering: Dithering will be slower but with a better quality.

Text Viewer

- Inline Images on Demand: If you enable this option, inline images will not be fetched (and thus shown) automatically. Instead you will see an Amadeus-head-icon indicating the inline image. You may fetch the inline image by double-clicking on this icon. This feature is mainly used when you have a very slow connection to your database server and you want to look at the text first and decide afterwards whether it is interesting for you.

6.5.6.6 The „Toolbar“ and „Statusbar“ commands

With these commands you can toggle the appearance of the toolbar and the statusbar.

The left area of the status bar describes actions of menu items as you use the arrow keys to navigate through menus. Similarly, this area shows messages that describe the actions of toolbar buttons as you depress them, before releasing them. If after viewing the description of the toolbar button command you do not wish to execute the command, then release the mouse button while the pointer is off the toolbar button.

6.5.6.7 The „Save Settings on Exit“ and „Save settings now“ commands

Here you can specify when Amadeus should store its user configuration settings (link fonts, colors, search type, windows size, ...). Either when exiting Amadeus, or on demand (save now) or never (none of these two options selected or checked).

6.5.7 The „Window“ menu

6.5.7.1 Overview

These are the commands that are available in the Window menu:

Commands	**Purpose**
Cascade	cascades displayed windows on the desktop
Tile Horizontal	tiles displayed windows horizontally on the desktop
Tile Vertical	tiles displayed windows vertically on the desktop
Arrange Icons	arranges icons on the desktop
Tile Always Horizontal	tiles displayed windows horizontal on the desktop automatically whenever the program window is resized
Tile Always Vertical	tiles displayed windows vertical on the desktop automatically whenever the program window is resized

6.5.7.2 The „Cascade“ command

Arranges windows in an overlapped fashion.

6.5.7.3 The „Tile Horizontal“ and „Tile Vertical“ commands

Arranges windows in non-overlapping horizontal or vertical tiles.

6.5.7.4 The „Tile Always Horizontal“ and „Tile Always Vertical“ commands

Arranges windows in non-overlapping horizontal or vertical tiles every time the main windows is sized or a new viewer is created (= not the same as opening an existing window!)

6.5.7.5 The „Arrange Icons“ command

Use this command to arrange the icons for minimized windows at the bottom of the main window. If there is an open document window at the bottom of the main window, then some or all of the icons may not be visible because they will be underneath this document window.

6.5.8 The „Help“ menu

6.5.8.1 Overview

These are the commands that are available in the Help menu:

Command	Shortcut Key	Purpose
Index		displays the index of Amadeus Help topics
Using help		displays instructions on how to use help
About		displays Amadeus copyright and version information

6.5.8.2 The „Index“ command

Use this command to display the opening screen of Help. From the opening screen, you can jump to step-by-step instructions on how to use Amadeus and various types of reference information.

Once you open Help, you can click the Contents button whenever you want to return to the opening screen.

6.5.8.3 The „Using help“ command

Use this command for instructions about using Help.

6.5.8.4 The „About“ command

Use this command to display the copyright notice and version number of your copy of Amadeus.

6.6 Appendix to Amadeus

6.6.1 Command line options

These override default settings specified in the „Amadeus.ini“-file

Switch	**Old Switch**	**Default**	**Note**
-hghost <hghost>	-r <hghost>	-r hgiicm.tu-graz.ac.at	specifies the Hyper-G database to which to connect. E.g. hmu1.cs.aukuni.ac.nz (New Zealand), tracy.esrin.esa.it (Italy) email.tuwien.ac.at (Vienna, Austria), hyperg.edvz.uni-linz.ac.at (Linz, Austria)
-hgport <hgport>	-p <hgport>	-p 418	specifies a different port number
-lang <language>	-l <language>	-l english	specifies a different user language
-coll <startcoll>	-c <startcoll>	-c ~anonymous	jumps directly to a specified collection at startup
-sort	-s	#T	sets default sort order for collection browser

6.6.1.1 The ‘hghost’ command line option

Typically users would specify the Hyper-G server here, but they may also define an URL, if they don’t have an Hyper-G server in place or want to use Amadeus as an WWW or Gopher client.

Example for specifying the server without an URL:

```
-hghost hgiicm.tu-graz.ac.at
```

Currently the following URLs are supported:

Hyper-G

```
hyperg://{<user>{:<password>}@}<host>{:<port>{;<binary>{;<distri
buted>{;<compressed>}}}}/<collection>
```

or

```
hyperg:<global object id>
```

Examples:

hyperg://hyperg.tu-graz.ac.at:418;0;1/~anonymous

hyperg:0x811b9908_0x00077da0

Hint: It is a good idea to copy the Amadeus icon into the Program-manager several times, every time with different command line options (e.g. different Hyper-G servers).

Http (WWW) Protocol

```
http://<host>{:<port>}/<path>
```

Example:

http://www.tu-graz.ac.at/C~anonymous

Gopher Protocol

`gopher://<host>{:<port>}/<path>`

Example:

gopher://gopher.tu-graz.ac.at/1~anonymous

6.6.1.2 The ‘lang’ command line option

This sets the preferred language when connected to a Hyper-G server.

Currently the following languages are supported:

- english
- german
- french
- styrian

6.6.1.3 The ‘coll’ command line option

This sets the starting collection when connected to a Hyper-G server and if the user has not specified the Hyper-G server using an URL. Mainly used for compatibility with other Hyper-G tools.

Example:

-hghost hgiicm.tu-graz.ac.at -coll hyperg

6.6.1.4 The ‘sort’ command line option

This sets the default sort order for the collection browser, look at The „Sort Order“ command, page 88) for a detailed description of the options.

6.6.2 The „AMADEUS.INI“ file

Most of the Amadeus.ini-file settings are made by Amadeus itself or through menu commands and options. However, there are some settings that might be interesting for the advanced user.

6.6.2.1 The section [Amadeus]

Field	Default Value	Meaning
Port	418	specifies default port number for communicating with the Hyper-G database (usually port 418)
Server	hgiicm.tu-graz.ac.at	specifies the default Hyper-G database
ShowLogo	1	specifies whether the welcome logo is shown or not
CollectionHead	1	specifies whether the opening text pages are shown automatically
LocalDataPath=		specifies the path to the control file for the local database. If this field is omitted LocalAmadeus will look in the Amadeus working directory (default).

6.6.2.2 The section [Text]

Field	Meaning
Import#<nr>	used to specify several text import filter

`Export#<nr>`	`used to specify several text export filter`
`Editor#<nr>`	`used to specify (external) editors for modifying and creating new text documents (can be combined with import and export filters)`

The “Importxxx” Field

Syntax:

`Import#<nr>=<filter>,<description>,<filename extension>`

where **<nr>** is a unique successive number identifying this import filter (range from 000 to 999)

<filter> is an external filter application, or one of the following build-in internal filters:

- internalRTF, for converting RTF text to HTF
- internalHTF, no convertion, just syntax checking
- internalTXT, for converting ISO-text to HTF

Note: when using an external filter application, users may also use ‘$in’ and ‘$out’ as paramerters for the in (foreign format) and out (HTF format) file!

<description> is a description for the import filter format, that will be displayed in the list of import filters.

<filename extension> is an extension that is added to the converted (temporary created) filename to ensure correct recognization of the file format by the editor application (if used in conjunction with the “Editor”-field, see below).

The “Exportxxx” Field

Syntax:

```
Export#<nr>=<filter>,<description>,<filename extension>
```

where <**nr**> is a unique successive number identifying this export filter (range from 000 to 999)

<**filter**> is an external filter application, or one of the following build-in internal filters:

- internalTXT, for converting HTF to formatted ISO-text
- internalHTF, no convertion, pass through

Note: when using an external filter application, users may also use ‘$in’ and ‘$out’ as paramerters for the in (HTF format) and out (foreign format) file!

<**description**> is a description for the export filter format, that will be displayed in the list of export filters (“save as” dialog).

<**filename extension**> is an extension that is added to the converted (temporary created) filename

The “Editorxxx” Field

Syntax:

```
Editor#<nr>= <editor commandline>,<export filter nr>,<import
filter nr>,<description>
```

where <**nr**> is a unique successive number identifying this import filter (range from 000 to 999)

<**editor commandline**> is an external text editor application, or the build-in internal editor:

- internalHTF, very simple editor for HTF or ISO-text

<export filter nr> specifies the unique number of the export filter, or '-1' if none required (convertion is being processed before the launch of the editor)

<import filter nr> specifies the unique number of the import filter, or '-1' if none required (convertion is being processed after the saving the text in the editor)

<description> is a description for the import filter format, that will be displayed in the list of import filters.

Example:

Editor#000=internalHTF,-1,-1,Build-in HTF Editor

6.6.2.3 The section [TextViewer]

Field	Default Value	Meaning
`DefTabWidth`	`50`	`defines the multiplication factor for TAB stops in pixels`

The "DefTabWidth" Field

This field sets the multiplication factor of the TAB-stops. If users specify e.g. 100, that means that TAB stops are set at 100, 200, 300, 400, ...

6.6.2.4 The section [Generic]

Syntax

```
<subtype>=<viewer> {$arg0} {$arg1} {$arg2} {$arg3} {$arg4}
{$arg5 {$arg6} {$arg7} {$arg8} {$arg9}
```

Note the following rules for using user defined arguments:

- $arg? (?=0-9) are the arguments specified at 'Arguments=' in the document
- '$arg0' specifies the document file name, thus it should always be specified (usually)

- you may change the order of the arguments (e.g.: $arg2 $arg3 $arg4 $arg1)
- you may mix the argument-place holders with other chars, but you have to seperate the arguments with spaces or tabs ! E.g.: '-a -b -c $arg1 $arg0', but NOT '$arg2ab$arg3c $arg0'

Look for a detailed description on using generic objects and their ini file settings in “Inserting an Image, a Movie, a Sound, a Scene, a Postscript or a Generic document:”, page 61.

Hint: Look in the „Amadeus.ini“-file, there are some comments on additional entries.

7. Bibliography

[Chi95] Keith Andrews, Frank Kappe, and Hermann Maurer: “Hyper-G and Harmony: Towards the Next Generation of Networked Information Technology”: “Formal Demonstration, CHI'95 Conference Companion”, p33-34; May 1995; also avaiblable at ftp://ftp.iicm.tu-graz.ac.at/pub/Hyper-G/papers/chi95.ps

[Fenn94] Barry Fenn, Hermann Maurer: “Harmony on an expanding net”; Interaction: p26-38; October 1994

[Gopher+] Farhad Anklesaria, Paul Lindner, Mark McCahill, Daniel Torrey, David Johnson, and Bob Alberti: “Gopher+ Upward Compatible Enhancements to the Internet Gopher Protocol”: ftp://boombox.micro.umn.edu/pub/gopher/gopher_protocol/Gopher+/Gopher+.txt, July 1993

[GopherVR] Mark P. McCahill and Thomas Erickson. “Design for a 3D Spatial User Interface for Internet Gopher”: Proc. of ED-MEDIA 95, pages 39-44, Graz, Austria, June 1995. AACE

[HGProt] “Hyper-G Client/Server Protocol (HG-CSP)”: ”: ftp://ftp.iicm.tu-graz.ac.at/pub/Hyper-G/papers/Protocol.ps

[HTF] “Hyper-G Text Format”: ftp://ftp.iicm.tu-graz.ac.at/pub/Hyper-G/papers/HTF.ps

[HTML] Tim Berners-Lee and Dan Conolly: “Hypertext Markup Language (HTML).”: http://www.w3.org/hypertext/WWW/MarkUp/MarkUp.html, June 1995

[Kappe91] Kappe F.:Aspects of a Modern Multi-Media Information System, PhD theses Graz University of Technology, Jun. 91, also available as IIG report 308 and per anonymous ftp: ftp://ftp.iicm.tu-graz.ac.at/pub/Hyper-G/papers/report308.ps

[Kauf91] F.-J.Kauffels: “Rechnernetzwerk-Systemarchitekturen und Datenkommunikation”; B.I. Wissenschaftsverlag; 1991

[Krol94] Ed Krol: “The Whole Internet”; O’Reilly of Associates, Inc; 1994

[FreyAdam94] Donnalyn Frey/Rick Adams: “A Directory of Electronic Mail”; O’Reilly of Associates, Inc; 1994

[Liu94] Liu, Peek, Jones, Buus & Nye: “Managing Internet Information Services”; O’Reilly of Associates, Inc; 1994

[RFC791] “INTERNET PROTOCOL-DARPA INTERNET PROGRAM-PROTOCOL SPECIFICATION”: http://ds1.internic.net:80/rfc/rfc791.txt; September 1981

[RFC793] “TRANSMISSION CONTROL PROTOCOL-DARPA INTERNET PROGRAM-PROTOCOL SPECIFICATION”: http://ds1.internic.net:80/rfc/rfc793.txt; September 1981

[RFC826] David C. Plummer: “An Ethernet Address Resolution Protocol”: http://ds1.internic.net:80/rfc/rfc826.txt; November 1982

[RFC882] P. Mockapetris: “DOMAIN NAMES - CONCEPTS and FACILITIES”: http://ds1.internic.net:80/rfc/rfc882.txt; November 1983

[RFC894] Charles Hornig: “A Standard for the Transmission of IP Datagrams over Ethernet Networks”: http://ds1.internic.net:80/rfc/rfc894.txt; April 1984

[RFC1055] J. Romkey: “A NONSTANDARD FOR TRANSMISSION OF IP DATAGRAMS OVER SERIAL LINES: SLIP”: http://ds1.internic.net:80/rfc/rfc1055.txt; June l988

[VRML] Gavin Bell, Anthony Parisi and mark Pesce. “The Virtual Reality Modeling Language - Version 1.0 Specification”: http://vrml.wired.com/vrml.tech/vrml10-3.html

[WashEvan93] K. Washburn/J.T.Evans: “TCP/IP Running a Successful Network”; Addison-Wesley; 1993

[WWW] Tim Berners-Lee, Robert Cailliau, Ari Luotonen, Henrik Frystyk Nielsen, and Arthur Secret: “The World-Wide-Web”: Communications of the ACM, 37(8):76-82, August 1994

[WWWHist] “History to date”: http://www.w3.org/hypertext/WWW/History.html

8. Table of Figures

9. Index