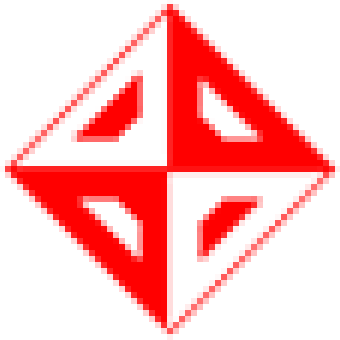
DETAILED DESIGN REPORT



Fall 2005

Ahmet Tolga Kılınç **1297944**
Berkan Kurtoğlu **1297993**
Hüseyin Özgür Batur **1297530**
Kerim Korkmaz **1347681**

TABLE OF CONTENTS

1. INTRODUCTION

1.1. PURPOSE OF THE DOCUMENT

This document is about the detailed design features and strategies of the Application Level Gateway for Web Access Control and Accounting Project. Throughout this document, the following items exist: Description and Goals of the Project, Technical Description of System Units, Architectural Design and Interface Design of the Project.

1.2. SCOPE OF THE PROJECT

1.2.1 Description of the Project

MindGate is designed to be a gateway application which controls internet access of an organization that has a LAN (Local Area Network). The software runs in the application layer of the gateway computer of the Local Network and it does not slow down the web traffic of organization. MindGate will provide several facilities to administrators for maintaining the web traffic of the local area network.

1.2.2 Features of the Project

MindGate contains numerous features to control the web traffic of an organization. The main features to be provided are listed below:

- MindGate inspects all requests and responses on the traffic and allow the ones which satisfy the pre-conditions.
- MindGate enables the administrators to define new user groups, web site categories and filter thresholds for the filtering process.
- MindGate requires logging into the system before using the network resources of the organization, so the system has the total control over users.
- MindGate has smart filtering techniques for preventing users entering sites which contain malicious content.
- MindGate uses the system resources effectively and contains several methods not to slow down the network traffic.
- MindGate keeps the track of the users' all actions and provides the statistics of uses to administrators.
- MindGate has user friendly interface for administrators who can handle all the web traffic easily.

1.3. SUCCESS CRITERIA OF THE PROJECT

In the Requirement Analysis Report, the requirements of the MindGate project are stated. The success criteria of this project is fulfilling the normal and expected requirements of the project. However, we have omitted some of the requirements after further research. Network traffic controlling is no longer in the success criteria of the project since MindGate aims to control web traffic of the LAN. The success criteria of the project are listed below:

- Reliable filtering
- Sophisticated Categorization
- Blocking harmfully categorized content in a flexible way
- Feedback from users about filtering
- Not slowing down the network traffic
- Constructing clearly defined categories
- Priorities for some users to access Web Content
- User Account-IP Binding
- Keeping user information and history
- Storing user internet access statistics
- User- friendly Interface
- System Rescue

1.4. DESIGN CONSTRAINTS AND LIMITATIONS

1.4.1 Performance

MindGate works on web control, so there will be lots of requests and responses coming through every instance. Because of this MindGate should perform efficiently and meet the demands of the users while satisfying the security issues. To be fast enough, we have omitted some features such as dynamic filtering.

1.4.2 Reliability of Libraries and Other Software

MindGate has several features such as RAM indexing which uses libraries and other software. These libraries and software decrease our amount of work, but our design of classes and functions should be made accordingly to manage compatibility. Lucene and JigSaw are examples of libraries and software that will be used in MindGate project.

1.4.3 Time

The MindGate project has the due date of May 2006, so the project must be completed accordingly.

1.5. GENERAL DESIGN GOALS

1.5.1. Reliability

MindGate will be responsible for the security of web traffic of an organization, so the reliability is the leading goal of the project. The program will keep all the related information in the logs and relational database to handle the tasks. It will also check all the requests and responses in web traffic.

1.5.2. Dynamic Knowledge Base

MindGate enables the administrators to mention categories of websites and give thresholds and values them to secure the organization. Also MindGate has the main feature of increasing its knowledge base which will increase the security reliability and decrease the user's waiting time for the program to function the request. MindGate keeps the record of pre-visited sites in the RAM index in a categorized fashion, so as the time passes the knowledge base of the program will enhance and the program will function better.

1.5.3. Usability

Although MindGate program will be used by administrators, our goal is to design the program with user-friendly interfaces. This will contribute to administrators' efficiency and ease the tasks to be done.

2. MINDGATE APPLICATION LEVEL GATEWAY DESIGN

System will be described as sequences of data flow diagrams, data dictionary and state transition diagrams. These diagrams reflect the structure of the system to be used in further design and implementation process. For description of the flowing data on the DFD diagrams see Data Dictionary Part.

2.1. DATA FLOW DIAGRAMS:

2.1.1. Mind Gate System:

MindGate will run on a *Linux server*, with an *SQL server* and a *web server (Jigsaw)* and *a Proxy (Jigsaw Proxy mod)*. With a*rchive file system*, these are the four external interfaces to the system. MindGate will have four main processes inside, **Connection Control** which will handle all requests and responses, **Database Control** which handles static index checking, user authentication, and log archiving. **Interface Control** generates form pages for user interaction and passes the filled forms to the system. **Categorizer** which is the most sophisticated part of the system which will handle dynamic categorization of the web pages to improve the knowledge base of the system. General data flow of the system is below:

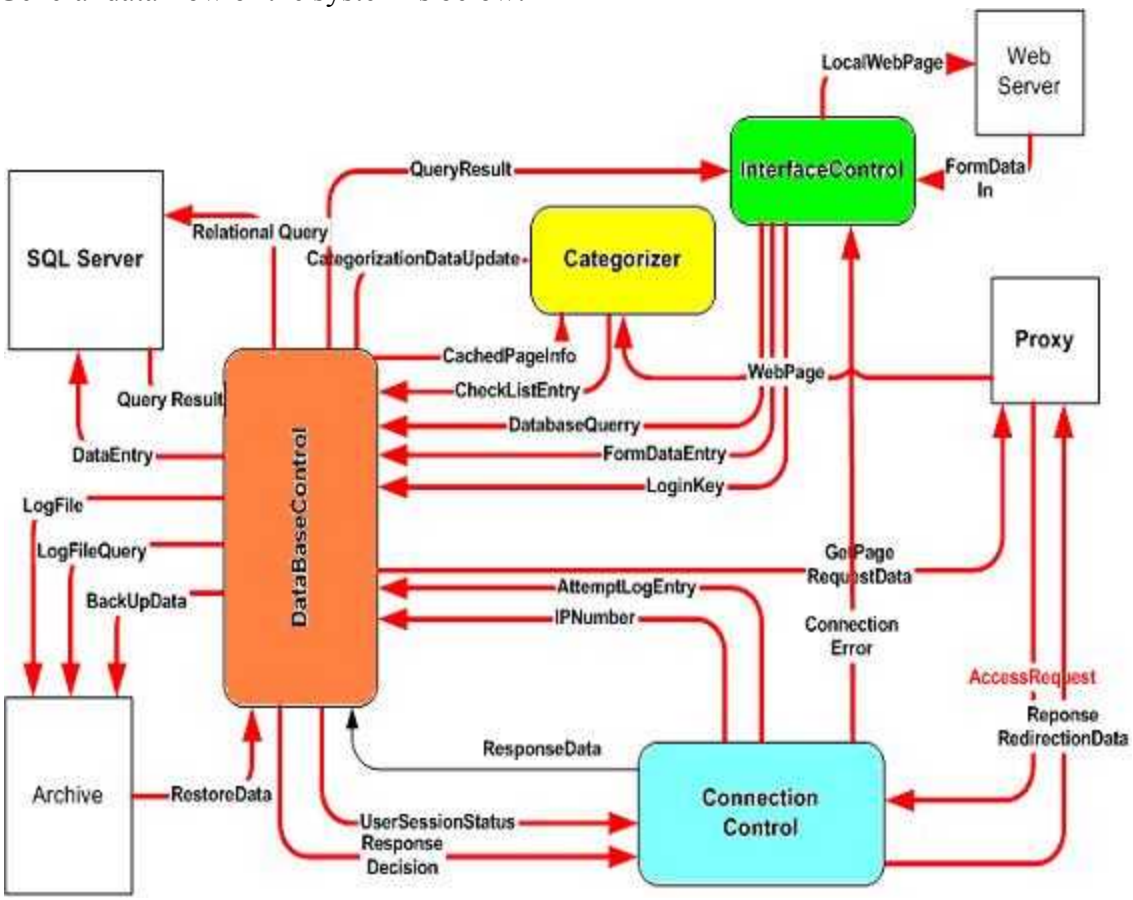


Figure1. Level 1 Data Flow Diagram

2.1.2. Connection Control

Connection Control handles all interaction with the clients inside the subnet. Each access request that comes to the *Proxy* is invokes a new thread in the ***Connection Control***. ***Connection Control*** takes the request info using *Request Handler* and asks *Database Control* if the owner of the access request has a session opened in the system. *Response Handler* gives the appropriate response to the client by redirecting user to login page, a cached page, or a page outside the network or a generated page. In case of redirection to page generation *Interface Control* invoked to generate the required page on internal web server. In all cases, fate of the request is logged as an attempt log entry and passed to the *Data Base Control* for achieving purposes.

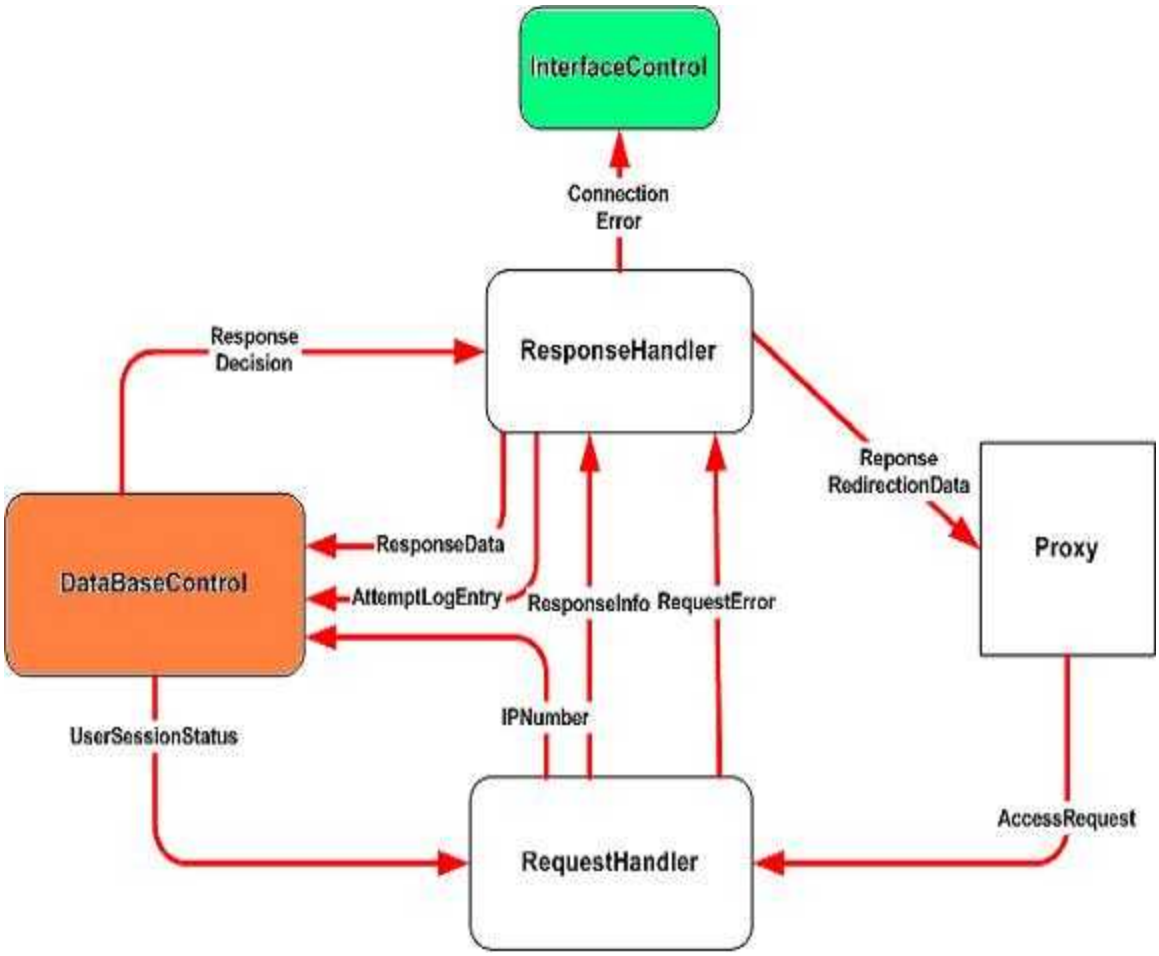


Figure2. Level 2 Data Flow Diagram for Connection Control

2.1.3. Database Control

Database Control is the largest and most complex part of the application. To deal with the problems of concurrency and synchronization, most of the data storages are connected this part.

Main responsibility of the ***Database Control*** is to hold the static check list. When connection control handles a requests it asks to block or allow the request to the database control. If the requested website is not in the checklist, database controller invokes the *Categorize*r to learn the category of the webpage. Result is added to the C*heck List* as a new entry. Then using the username info coming from Connection Control, the Group info table returns a category set for that user and if the requested web page's category found in this set, decision is signaled to the *Connection Control* if not, allow decision is signaled. In case of block *Connection Control* will redirect client to an error page and error page will be generated by the *Interface Control.*

Logging and Backup is also responsibility of the ***Database Control***. Each action, error in the other parts of the program is delivered to the ***Database Control***. These are collected and entered into to *Log File Image* buffers and flushed into the *Archive* file system periodicaly. The system tables: group info table, check list and category info table (in the *Categorizer*) are also flushed into archive file system on request and shutdown of the gateway.

Another responsibility of the ***Database Control*** is to handle the user interaction (mostly administrator in our system) coming form the ***Interface control*** unit. From Interface Control Unit, a data entry or a data query will come, and these requests might address three kinds of data sources: Relational Database, Log files and System tables hold in the RAM.
Administrator may see or change user information from the SQL server or Group Info Table.

Contrary to its name ***Database Control*** controls the whole system since administrator makes a change in the data and behavior of the program changes according to these data. This is because whole software is not a conventional application but a network service.

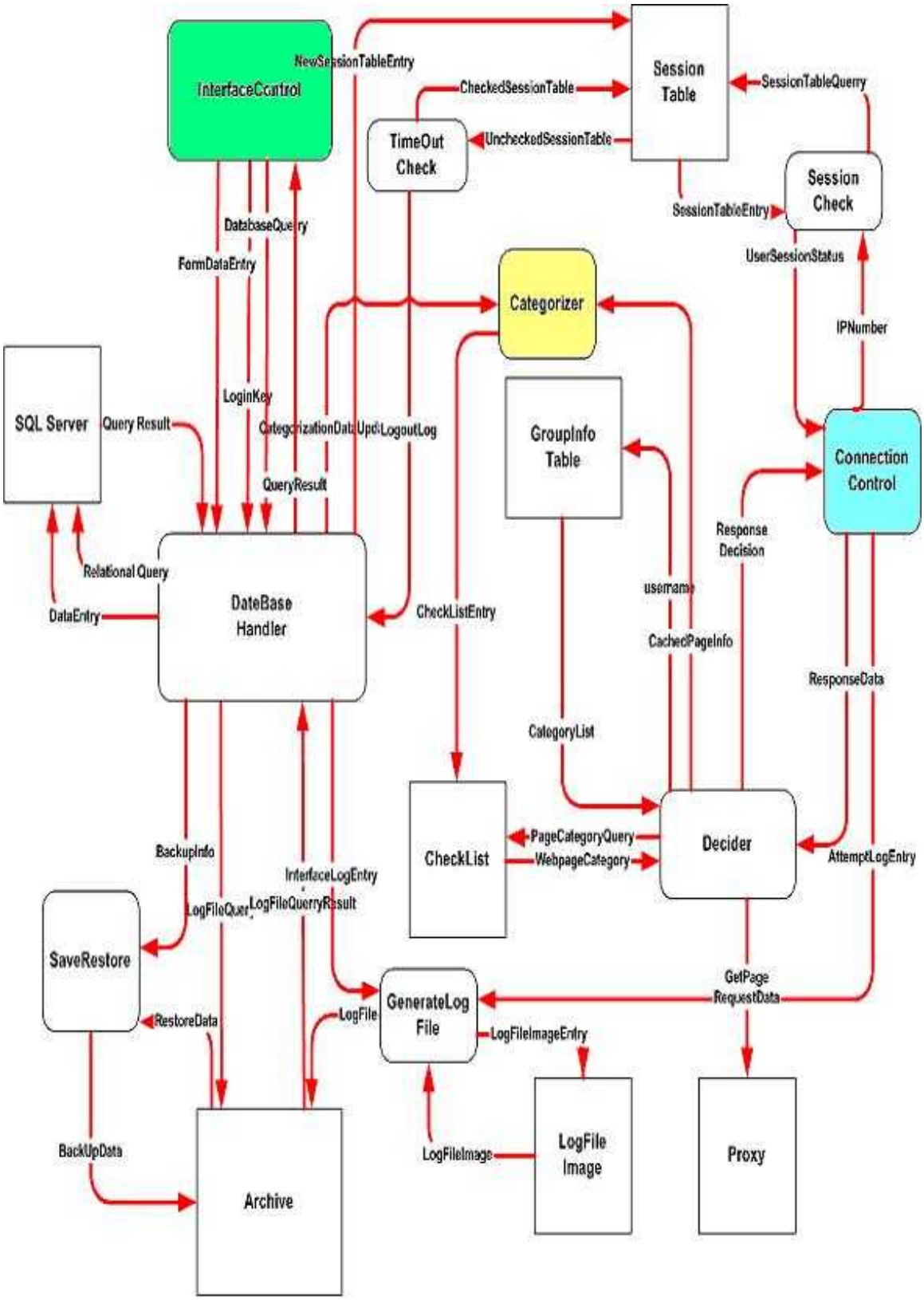


Figure3. Level 2 Data Flow Diagram for Database Control

2.1.4. Categorizer

Categorizer is the part which dynamically categorizes new web pages which were not hold in the knowledge base of the system. When a requested page is not found by the *Database Control*, ***Categorizer*** takes action and Html Parser fetches required page from the Proxy and Parses it to get extracted page data. The Split function takes knowledge base data from the Categorization Data table and divides it into structural and content data parts pass data to the analyzers. Evaluator takes the numerical results from the analyzers and gives a category name. Shortly, categorizer categorizes web page using certain heuristics and adds a new entry to the *Checklist Table* of the *Database Control* part.

For details of categorization algorithms *see About Web Page Categorization* part.

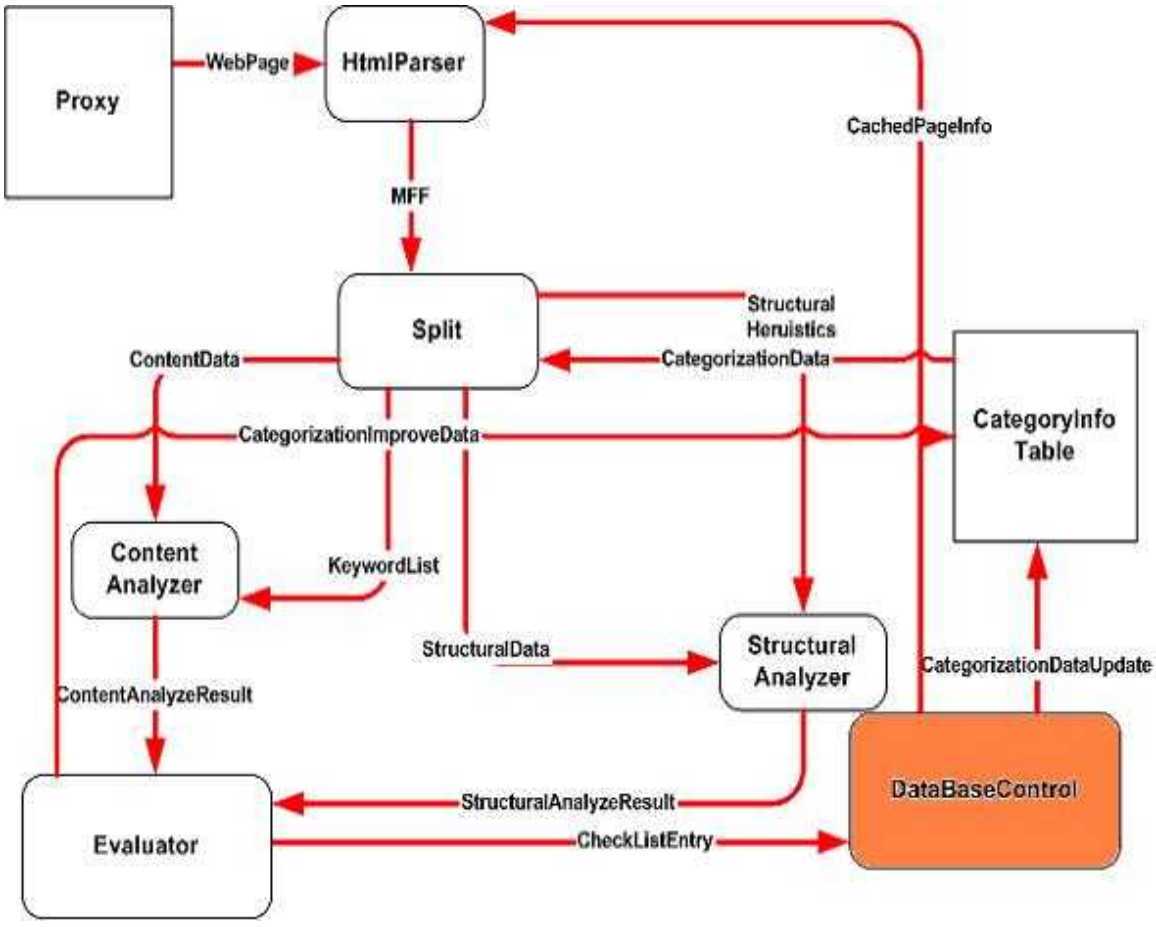


Figure4. Level 2 Data Flow Diagram for Categorizer

2.1.5. Interface Control

Interface Control is the part between the web server and the database control. It holds the forms filled by user and checks the form if it is valid, if not it passes for to the web server and asks user to fill the form correctly. *Admin Console* handles forms related to admin and transforms actions into data query or data entry types which will be handled in the *Database Control*. Form generator passes result of the queries into forms and the page generator delivers the resultant web page to the server. The connection error messages passed to the user are also delivered using *Page Generator*. In either cases redirection to these pages are handled by *Connection Control.*

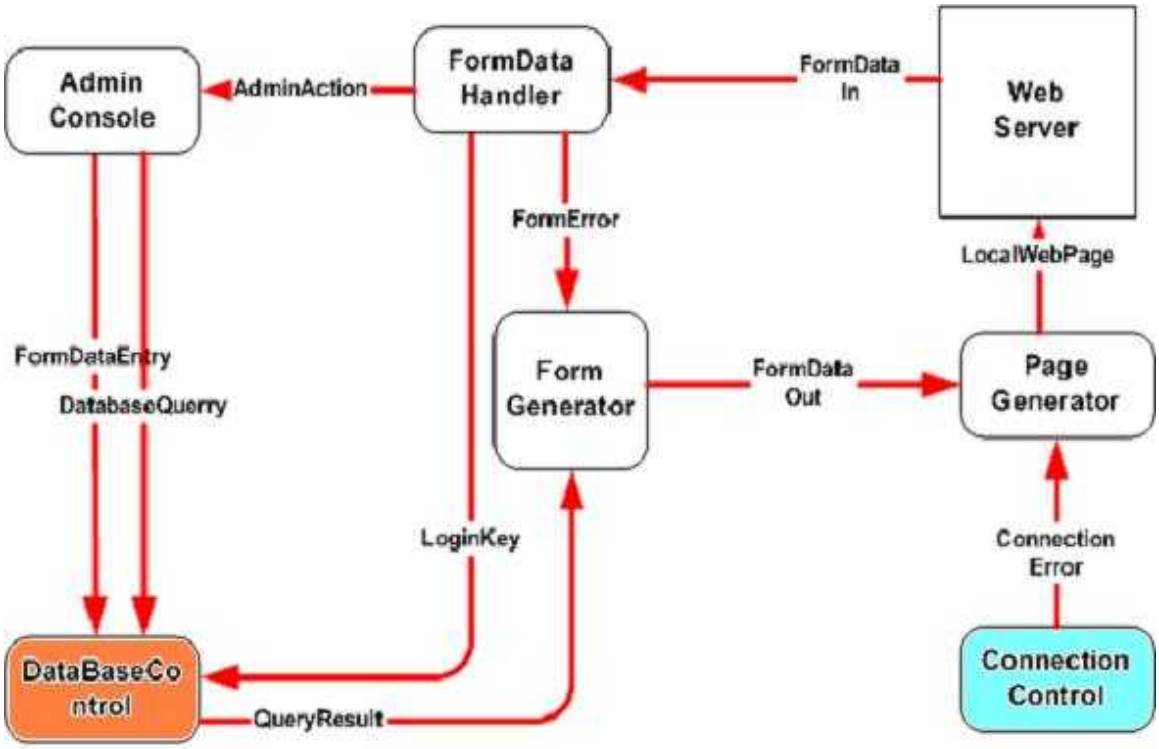


Figure5. Level 2 Data Flow Diagram of Interface Control

2.2. STATE TRANSITION DIAGRAMS

In this section state transition diagrams for the frequently occurring main events will be described. Since the system is complex whole states and events will not be covered yet.

The system is a gateway so it is obvious that it must concurrently handle all clients' request. According to this scheme, MindUs designed the system as composition of threaded components invoked on user requests one for each user just as in the web servers. All interfaces except the file system handles concurrency so consistency inside the system will be sufficient to avoid race conditions and other problems with timing.

The three diagrams below are about the sequential processing of three main scenarios, and each of them reflects not the whole system but just the invoked thread.

2.2.1. Processing of an Authorized User Request

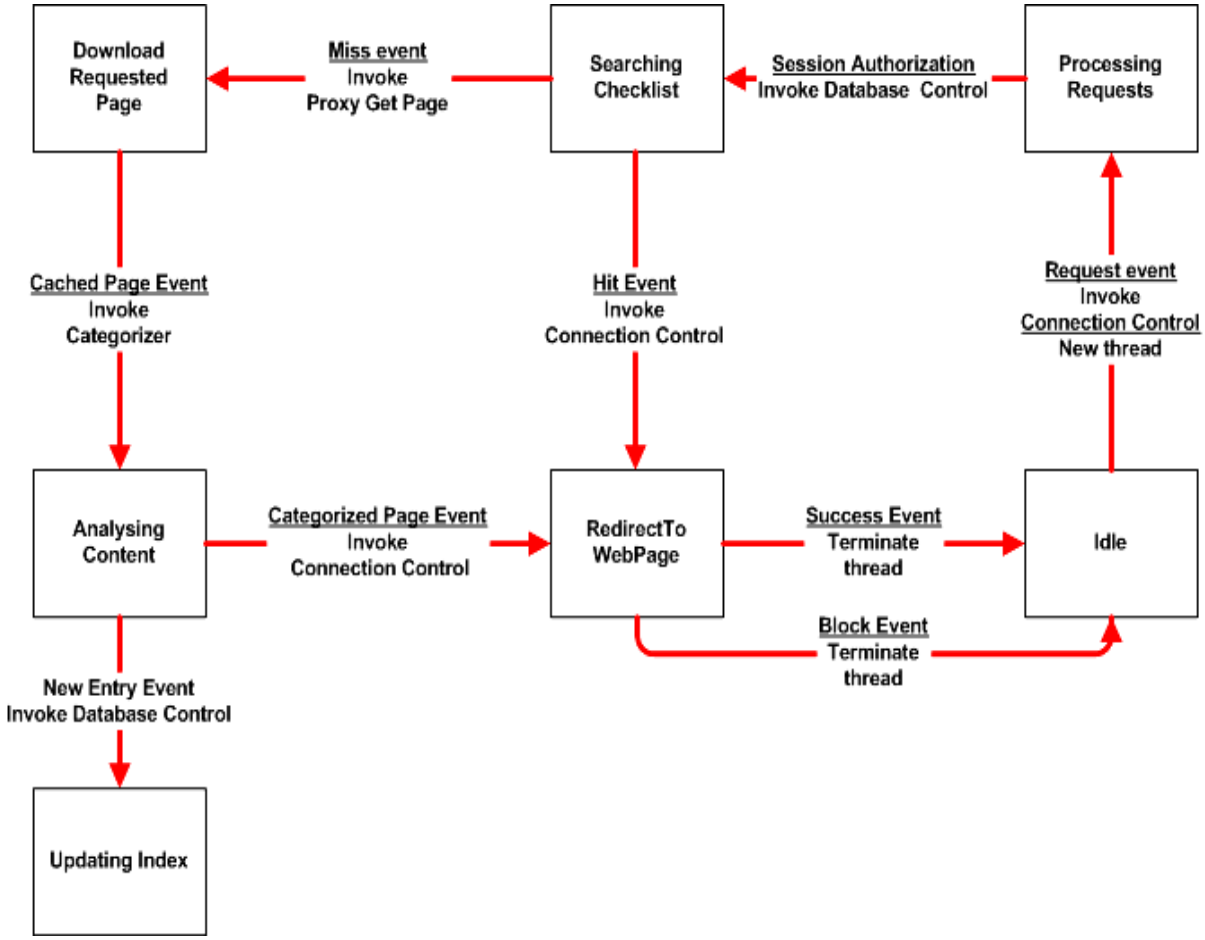


Figure6. State Transition Diagram for Authorized User Request

2.2.2. Processing of an Unauthorized User Request

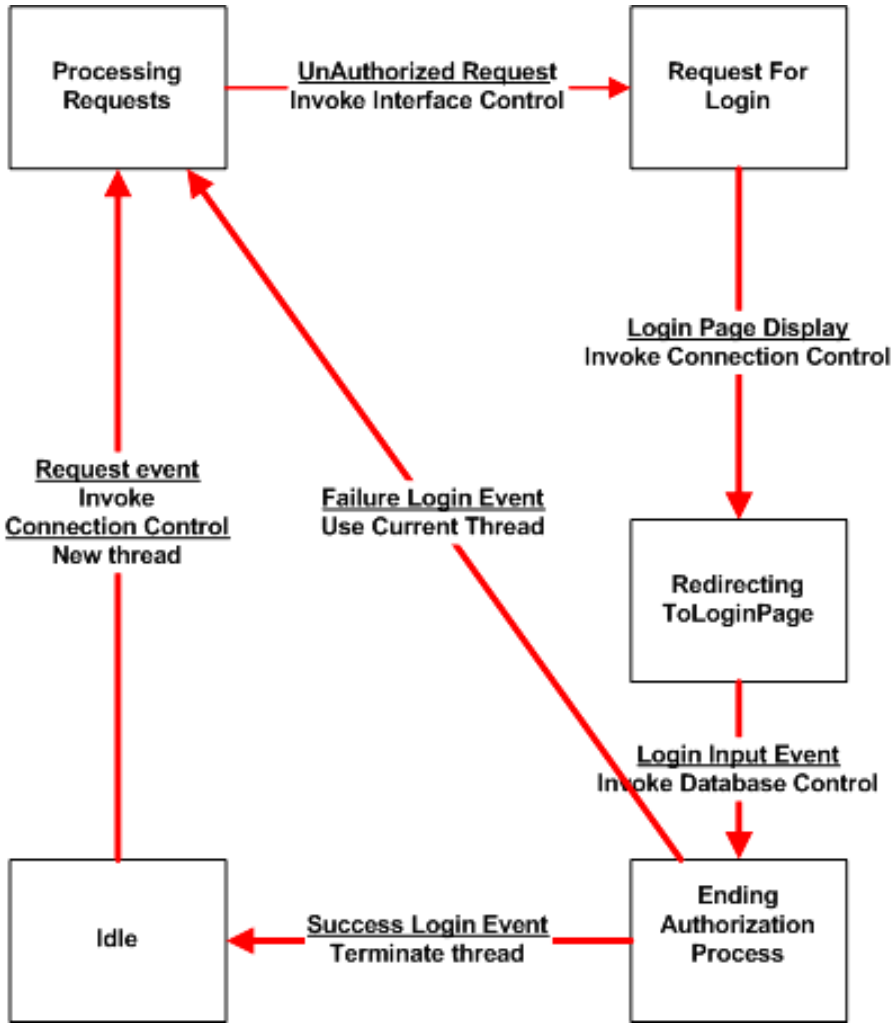


Figure7. State Transition Diagram for Unauthorized User Request

2.2.3. Processing of an Interaction with Administrator

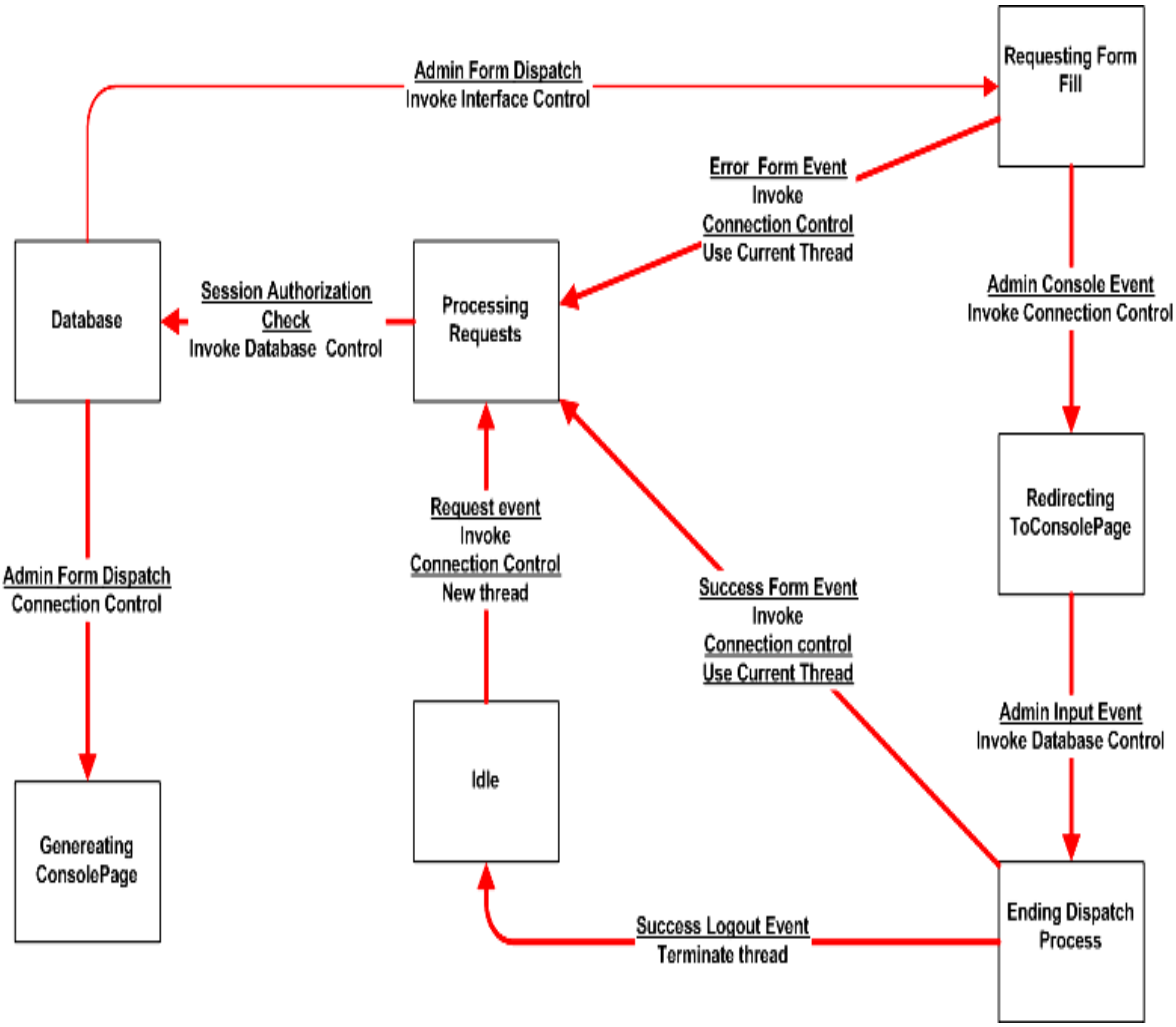


Figure8. State Transition Diagram for Administrator Interaction

2.3. DATA DICTIONARY:

These are the data shown in the Data Flow Diagrams with detailed explanation:

2.3.1. Categorizer DD

Type	CachedPageInfo
From	DataBaseControl
To	HtmlParser
Explanation	The URL of the website is passed to HTML Parser to analyze the content of the website by Database Control.

Type	CategorizationData
From	CategoryInfoTable
To	Split
Explanation	Categorization Data consists of name of the category, keywords of the category, image weight of the page, link weight of the page and script weight of the page.

Type	CategorizationDataUpdate
From	DataBaseControl
To	CategoryInfoTable
Explanation	Categorizer Unit uses Category Info Table to analyze and categorize the website. CategorizationDataUpdate updates this table in the Categorizer Unit. Categorization Data consists of name of the category, keywords of the category, image weight of the page, link weight of the page and script weight of the page.

Type	CategorizationImproveData
From	Evaluator
To	CategoryInfoTable
Explanation	Evaluator updates the Category Info Table in order to improve categorization. Category Info Table holds the name of the category, keywords of the category, image weight of the page, link weight of the page and script weight of the page.

Type	CheckListEntry
From	Evaluator
To	DataBaseControl
Explanation	Categorizer returns the Domain, Subdomain and Category of the given URL after analyzing the website. This information is stored in the Check List.

Type	ContentAnalyzeResult
From	ContentAnalyzer

To	Evaluator
Explanation	Content Analyzer analyzes the content according to given keywords and returns the vector of constants to evaluator. The constants correspond to meta information, title and other evaluation according to mechanisms defined in the Content Analyzer.

Type	ContentData
From	Split
To	ContentAnalyzer
Explanation	Content Data contains text related content of the web page. The text related content is separated in the Split process.

Type	KeywordList
From	Split
To	ContentAnalyzer
Explanation	Keyword List is a string vector containing keyword lists for the web page categories.

Type	MFF
From	HtmlParser
To	Split
Explanation	MFF Data consists of Content Data and Structural Data. It is created in the HTML Parser and separated in the Split.

Type	StructuralAnalyzeResult
From	StructuralAnalyzer
To	Evaluator
Explanation	Structural Analyzer Result consists of constants which correspond to image weight of the image, link weight of the image and script weight of the image. These constants are evaluated in the Structural Analyzer and returned to Evaluator.

Type	StructuralData
From	Split
To	StructuralAnalyzer
Explanation	Split separates the Content Data and Structural Data from MFF and passes Structural Data to Structural Analyzer. Structural Data consists of images, links and scripts.

Type	StructuralHeruistics
From	Split
To	StructuralAnalyzer
Explanation	Structural Heuristics are constants stored in the Category Info Table. They show the values of structures such as images, links, scripts in the webpage category.

Type	WebPage
From	Proxy
To	HtmlParser
Explanation	Proxy redirects the HTML Stream to HTML Parser.

2.3.2. Connection DD

Type	AccessRequest
From	Proxy
To	RequestHandler
Explanation	Access Request contains the IP Number of the client machine where user requests a site and the Http Header of the request. The proxy redirects this info to Request Handler process to be inspected and evaluated.

Type	AttemptLogEntry
From	ResponseHandler
To	DataBaseControl
Explanation	Attempt Log Entry consists of IP Number of the client machine, Username, URL of the requested webpage, Attempt Status and Attempt Time. *See Attempt Logs in 4.1.3 Connection Logs Section*

Type	ConnectionError
From	ResponseHandler
To	InterfaceControl
Explanation	In case of error, the Response Handler returns the description and the code of the error to the Interface Control

Type	IpNumber
From	RequestHandler
To	DataBaseControl
Explanation	IP Number of the client machine where the request comes from.

Type	RequestError
From	RequestHandler
To	ResponseHandler
Explanation	Request Handler returns either the IP Number of the requesting client machine or the Error Code.

Type	ResponceData
From	ResponseHandler
To	DataBaseControl
Explanation	Response Handler passes the URL and the Username to the Database Control in order to get the Response Decision

Type	ResponceDecision
From	DataBaseControl
To	ResponseHandler
Explanation	Database Control returns a decision about the Response with or without a Description

Type	ResponseInfo
From	RequestHandler
To	ResponseHandler
Explanation	Request Handler gives the requested URL, Username and the IP Number of the client machine to the Response Handler process.

Type	ResponseRedirectionData
From	ResponseHandler
To	Proxy
Explanation	The URL to be displayed and the IP number of the requesting client machine is passed to the Proxy in order to display the website.

Type	UserSessionStatus
From	DataBaseControl
To	RequestHandler
Explanation	The Username of the user who is using the client machine of a given IP Number is returned from the Database Control to the Request Handler.

2.3.3. Database DD

Type	AttemptLogEntry
From	ConnectionControl
To	GenerateLogFile
Explanation	Attempt Log Entry consists of IP Number of the client machine, Username, URL of the requested webpage, Attempt Status and Attempt Time. *See Attempt Logs in 4.1.3 Connection Logs Section*

Type	BackupData
From	SaveRestore
To	Archieve
Explanation	All types of log files are stored in the Archieve for backup purposes. *See 4.4 Archive for details.*

Type	BackupInfo
From	DataBaseHandler
To	SaveRestore
Explanation	Files to be stored in the Archive is passed to the SaveRestore process. *See 4.4 Archive for details.*

Type	CachedPageInfo
From	Decider
To	Categorizer
Explanation	The URL of the website is passed to Categorizer Unit to analyze the content of the website.

Type	CategorizationDataUpdate
From	DataBaseHandler
To	Categorizer
Explanation	Categorizer Unit uses Category Info Table to analyze and categorize the website. CategorizationDataUpdate updates this table in the Categorizer Unit. Categorization Data consists of name of the category, keywords of the category, image weight of the page, link weight of the page and script weight of the page.

Type	CategoryList
From	GroupInfoTable
To	Decider
Explanation	A vector containing information about the User Groups is returned from Group Info Table to Decider. Group Info Table consists of Category Lists and Username Lists. *See 4.2.2. GroupInfo Table for details.*

Type	CheckedSessionTable
From	TimeOutCheck
To	SessionTable
Explanation	The Time Out Check process finds out if the user is online or logged out (the system may log out the user automatically if he/she is idle-sends no requests- in a given time period) by looking in the Session Table. Session Table holds the IP Number of the client machine, Username of the user using the given IP, Session Action (Login, Logout) and Action Time.

Type	CheckListEntry
From	Categorizer
To	CheckList
Explanation	Categorizer returns the Domain, Subdomain and Category of the given URL after analyzing the website. This information is stored in the Check List.

Type	DatabaseQuery
From	InterfaceControl
To	DataBaseHandler
Explanation	The queries coming from the Interface Unit. They could be Relational Query, Archive Query or System table Query. *See Sections 4.3 and 4.4 for details.*

Type	DataEntry
From	DataBaseHandler
To	SQLServer
Explanation	Database Handler inserts new entries in the SQL Server.

Type	FormDataEntry
From	InterfaceControl
To	DataBaseHandler
Explanation	User can enter new information to be stored in the SQL Server and this information is handled by Database Control.

Type	GetPageRequestData
From	Decider
To	Proxy
Explanation	The Decider gives the URL of the webpage to the Proxy. Proxy is supposed to get page of the given URL.

Type	InterfaceLogEntry
From	DataBaseHandler
To	GenerateLogFile
Explanation	The information generated from interface is logged into files. Database Handler passes the entries to Generate Log File process to handle logs.

Type	IPNumber
From	ConnectionControl
To	SessionCheck
Explanation	Connection Control gives IP Number to Session check process to find out the User Session Status.

Type	LogFile
From	GenerateLogFile
To	Archieve
Explanation	The log files of the system are kept in the Archive. *See 4.1. Log Files and 4.4. Archive for details.*

Type	LogFileImage
From	LogFileImage
To	GenerateLogFile
Explanation	Log File Image is returned to Generate Log File process to handle log files. *See 4.1. Log Files for details.*

Type	LogFileImageEntry
From	GenerateLogFile
To	LogFileImage
Explanation	An entry of Log File Image is passed to Log File Image process in order to form the new Log File Image. *See 4.1. Log Files for details.*

Type	LogFileQuery
From	DataBaseHandler
To	Archive
Explanation	Database Handler may need log files and information from the Archive. These are retrieved with the Log File Query. *See 4.1. Log Files and 4.4. Archive for details.*

Type	LogFileQueryResult
From	Archieve
To	DataBaseHandler
Explanation	Log files from the Archive are returned to the Database Handler in case of queries. *See 4.1. Log Files and 4.4. Archive for details.*

Type	LoginKey
From	InterfaceControl
To	DataBaseHandler
Explanation	Username and Password entered by the user in the Interface Unit is passed to the Database Unit in order to be verified.

Type	LogoutLog
From	TimeOutCheck
To	DataBaseHandler
Explanation	If the user has logged out, the Time Out Check process sends this information to Database Handler. The log will be entered in the system by the Database Handler. The log file contains IP Number, Username, Action Type (Log-in/Log-out) and Action Time. *See 4.1.3. Connection Logs for details.*

Type	NewSessionTableEntry
From	DataBaseHandler
To	SessionTable
Explanation	If the user has logged in the system or requested a page, Database Handler enters the new information into the Session Table. The entry consists of IP Number, Username and Last Request Time.

Type	PageCategoryQuery
From	Decider
To	CheckList
Explanation	The Decider asks the category of the website by supplying the URL of the site to the Check List.

Type	QueryResult
From	DataBaseHandler
To	InterfaceControl
Explanation	The results of the queries coming from the Interface Unit are returned from Database Control.

Type	RelationalQuery
From	DataBaseHandler
To	SQLServer
Explanation	The statistical information about the users are kept in the SQL Server. Database Handler sends queries to SQL Server if necessary information is needed.

Type	ResponceData
From	ConnectionControl
To	Decider
Explanation	Connection Control passes the URL and the Username to the Decider in order to get the Response Decision.

Type	ResponceDecision
From	Decider
To	ConnectionControl
Explanation	Decider returns a decision about the Response with or without a Description to Connection Control.

Type	RestoreData
From	Archieve
To	SaveRestore
Explanation	All types of log files are stored in the Archive for backup purposes. If the restored logs are needed by the system they can be restored to SaveRestore process. *See 4.4 Archive for details.*

Type	SessionTableEntry
From	SessionTable
To	SessionCheck
Explanation	IP Number of the client machine, Username of the user and the Last Request Time is passed to Session Check from Session Table.

Type	SessionTableQuery
From	SessionCheck
To	SessionTable
Explanation	Session Check process checks the IP Number of the user who enters the request by looking up in the Session Table.

Type	UncheckedSessionTable
From	SessionTable
To	TimeOutCheck
Explanation	The Session Table returns if the user is online or not to the Time Out Check process.

Type	Username
From	Decider
To	GroupInfoTable
Explanation	All users belong to a group in the MindGate system. The Group Info Table holds this information. The Decider process enters the username of the user to get the group of the user.

Type	UserSessionStatus
From	SessionCheck
To	ConnectionControl
Explanation	Session Check returns the Username that is currently linked with the entered IP Number by the Connection Control.

Type	WebPageCategory
From	CheckList
To	Decider
Explanation	The category name of the Web Page is returned to the Decider by the Check List.

2.3.4. Interface DD

Type	AdminAction
From	FormDataHandler
To	AdminConsole
Explanation	In the Web Forms of Admin Console, either Form Data Entries or Database Queries can be displayed.

Type	ConnectionError
From	ConnectionControl
To	PageGenerator
Explanation	In case of error, the Connection Control returns the description and the code of the error to the Page Generator.

Type	DatabaseQuery
From	AdminConsole
To	DataBaseControl
Explanation	The queries coming from the Admin Console. They could be Relational Query, Archive Query or System table Query. *See Sections 4.3 and 4.4 for details.*

Type	FormDataEntry
From	AdminConsole
To	DataBaseControl
Explanation	The Administrator can enter new information to be stored in the SQL Server and this information is handled by Database Control.

Type	FormDataIn
From	WebServer
To	FormDataHandler
Explanation	Possible form content is transferred from Web Server to Form Data Handler.

Type	FormDataOut
From	FormGenerator
To	PageGenerator
Explanation	Form Generator produces the form and pass this generated form content to Page Generator.

Type	FormError
From	FormDataHandler
To	FormGenerator
Explanation	Form Data Handler returns either Form Content or the Error occurred to the Form Generator.

Type	LocalWebPage
From	PageGenerator
To	WebServer
Explanation	Page Generator forms the Web Page to be displayed and passes to the Web Server.

Type	LoginKey
From	FormDataHandler
To	DataBaseControl
Explanation	Username and Password entered by the user in the Interface Unit is passed to the Database Unit in order to be verified.

Type	QueryResult
From	DataBaseControl
To	FormGenerator
Explanation	The results of the queries coming from the Admin Console are returned from Database Control. Form Generator creates necessary forms to be displayed.

3. DETAILED DESCRIPTION OF EXTERNAL COMPONENTS IN MINDGATE ARCHITECTURE

This section will be about external software components which will be included in MindGate Architecture to achieve maximum performance and many important goals.

MindGate will be a Content Filtering Gateway which should contain other important components which are a must in the web based technologies and also many other software components which make MindGate more powerful.

In the Architecture Description Part and all other describing graphs, figures and diagrams we describe the core MindGate Software, but according to our design methodology (Software Methodology), we see external parts as blackboxes and we designed software with interfaces to blackboxes which will be clearly described in the detailed design of MindGate Project.

While designing these interfaces in to the external parts in the architecture we consider some crucial assumptions which gathered during analysis report. These assumptions are all subject to change because they are all related to other software which we have not enough knowledge about. But we tried to minimize affects of these unknown parts by carefully encapsulating them.

Our assumptions about external components in the architecture:

3.1. FILTERING UNIT

3.1.1. Proxy Server

In Filtering Unit there will be a proxy server which will run in the client side of the local area network, handling all http requests coming from clients and all responses to these requests coming from internet.

Assumptions

1. Proxy Server should run thread based to handle many connections at the same time.

2. It should be secure.

3. It should have mechanisms to handle requests from clients, and interfaces to access, modify and control all these requests.

4. It should have mechanisms to make response to clients according to request, and interfaces to access, modify and control all these responses.

5. It should be fast enough.

6. It should have web page caching mechanisms and interfaces to access these cached pages.

7. Easy to integrate to Overall Architecture which will be developed in Java Programming Language.

According to these assumptions we make a technical search about possible proxy servers and found out some possible proxy servers which have these capabilities. Among these choices we decided to use **World Wide Web Consortium's Web Server Jigsaw** in proxy mod. See http://www.w3.org/Jigsaw/ for further information.

3.2. ARCHITECTURE MANAGEMENT UNIT

3.2.1. Web Server

In Architecture Management Unit there will be Web Server which will contain Admin Interface, Error Pages, Login Page, and User Feedback forms. This server will run as a part of the MindGate and will serve its contents as needed; it can be accessed via internet anywhere around the world.

All of the members in MindUs Team have enough knowledge about web servers and their technologies so we did not have a lot of assumptions about this component while designing the overall architecture.

Assumptions

1. It should be secure enough.
2. It should handle multiple connections, so it should run thread based.
3. It should be fast.
4. Easy to integrate to Overall MindGate Architecture.

As a web server we decided to use **World Wide Web Consortium's Web Server Jigsaw** which is implemented in Java Programming Language and has a powerful API.

3.3. DATABASE MANAGEMET UNIT

3.3.1. SQL Server

In Database Management Unit there will be an SQL Server which will used to store User Related Data of the MindGate. These data will be used for login and administrative purposes.

As in the Web Server Component nearly all of the members in MindUs Team have enough knowledge about SQL Servers so we did not have a lot of assumptions about this component while designing the Overall Architecture.

Assumptions

1. SQL server can connect to internet.
2. It should be thread based, to achieve concurrency in the System.
3. It should evaluate SQL queries, hold relational tables.
4. Effective Insertion, deletion and update mechanisms for relational tables it holds.
5. Easy to integrate to Overall MindGate Architecture.

As an SQL server we did not clearly decided to use one at this time but we think this will not be a problem for the design, because Database technology has a lot of standards all around the world so nearly all tools about Databases will apply them with slightly changes.

But possible ones which we can use in the implementation,

MySQL
Oracle
PostgreSQL

3.3.2. Indexer-Searcher

In Database Management Unit there will be an Indexer-Searcher Module these module is not really an external module. Detailed description of these module and parts it contains described in design part of this document.

Simply to make search faster as a team we choose to implement our text based indexer. This indexer will also have a searcher which will do fast searches on the index as needed.

Storing all the data on a relational database without making any distinction between them which many programs will used and all have limited time to finish it's task is not seem to us reasonable. So we investigate new ways to solve this problem, after searching we found out that there exists libraries which can be used to implement such indexers and searchers, so we decided to implement our own indexer and searcher these are will be used for fast retrieval purposes in the system.

One of our members has some experience about these libraries and technologies which guide us while designing this part of the architecture. But we have also do not have enough experience about this technology at this time so we have many crucial assumptions which are:

Assumptions

1. Fast indexing for text documents.
2. Fast search on these indexes.
3. Indexing documents according to key value pairs.
4. Should have access mechanisms to indexed key-value pairs
5. Storing index on RAM to make search faster.
6. It should have clear well documented API.
7. Searcher should support query search.
8. Implemented thread based to support multiple accesses for concurrency reasons.

According to our assumptions we design Overall Architecture of MindGate and we decided to use **Apache Lucene Indexing and Searching API** to implement indexing and searching mechanisms. The next section describes the indexing and searching operations with Lucene.

3.3.3. Indexing and Searching with Lucene

MindGate Database Controller Unit basically uses Lucene to indexing and searching XML documents in memory. All indexed files are held in the fast-access memory and not on a slower hard disk. Memory indexing is suitable for situations where very quick access to the index is needed, whether during indexing or searching.

MindGate uses XML format which has increasing usage in software application to exchange data. For instance, when a web page's category is determined by Categorizer Unit, it sends a XML document (exchanging data between units) to Database Unit. All methods of MindGate can easily identify XML documents and this would give great efficiency to software. All XML file formats can be seen in *4.1 Log Files section*.

Indexing a transaction of a XML file is first step for locating a file in memory. After indexing a file, administrator or units of software can easily query an entry in memory document. It takes constant time to response for any query. We have mentioned that after Categorizer Unit determined category of a web page, it sends a XML document to Database Controller Unit. But this file can not be indexed directly with its XML format. Lucene can not deal with rich media documents. Therefore, we are going to use Jakarta Commons Digester for XML parsing. Jakarta Commons Digester has not got high-speed but is most frequently using XML parsing. Once data is extracted from rich media text, Lucene can easily indexes document in RAM. This process is given below in detailed.

DocumentHandler interface for all documents parser:

We use DocumentHandler interface to create a plain text document from a rich document. This interface consists of a single method, `getDocument (InputStream)` which returns a Lucene `Document` for indexing.

```
public  interface DocumentHandler {

Document  getDocument  (InputStream is)
throws  DocumentHandlerException;

}
```

`getDocument` gets `InputStream` as input. This method reads and parses input and extract text from input. `getDocument` method returns an instance of the `Document` class with one or more Fields. This `Lucene Document` is ready to indexed by caller. If there is an error, `DocumentHandler` interface throws a `DocumentHandlerException` which is a simple subclass of Java's `exception` class.

Parsing and Indexing a XML Document

```
public class DigesterXMLHandler implements DocumentHandler{
```

For parsing and indexing XML document, MindGate uses `DigesterXMLHandler class` which uses `DocumentHandler` interface.

This class takes XML document, and also parses and indexes it.

```
<checklist>
   -<url>
        <name>www.ceng.metu.edu.tr</name>
        <category>education</category>
    </url>
</checklist>
```

This XML document is a checklist file which is constructed by Categorizer Unit and contains URL and its category. This file is sent by Categorizer Unit to Database Unit.

```
// instance of DigesterXMLHandler is created
dig.addObjectCreate("checklist", DigesterXMLHandler.class);
```

```
// instance of url is created
dig.addObjectCreate("checklist/url", url.class);

// set different properties of url instance using
// specified methods
dig.addCallMethod("checklist/url/name", "setName", 0);

// call 'populateDocument' method when the next
// 'checklist/url' pattern is seen
dig.addSetNext("checklist/url", "populateDocument");

//implement DocumentHandler interface
public synchronized Document getDocument(InputStream is)
throws DocumentHandlerException {
try {
dig.parse(is);  //Start parsing XML InputStream
}

//Populate Lucene Document with Fields
public void populateDocument(URL url) {
// create a blank Lucene Document
doc = new Document();
doc.add(Field.Keyword("name", url.getName()+url.getCategory()));
}
```

When administrator or system units want to query whether url is in check list or not , they query according to url name.

```
// JavaBean class that holds properties of each Url entry.
public static class Url {
private String name;
public void setType(String newName) {
name = newName;
}
public String getName() {
return name;
}
}

public static void main(String[] args) throws Exception {
DigesterXMLHandler handler = new DigesterXMLHandler();
Document doc =
handler.getDocument(new FileInputStream(new File(args[0])));
System.out.println(doc);
}
```

It can be seen that Digester provides a high-level interface for parsing XML documents.

Searching a Lucene Index

Searching in Lucene is as fast and simple as indexing. Searcher is a command-line program that we'll use to search the index which created by Indexer.

```
//Search class
public class Searcher {
//main takes two arguments
public static void main(String[] args) throws Exception {


File indexDir = new File(args[0]); // first Index directory
String q = args[1]; //second Query string

search(indexDir, q);  // search method is called


// definition of Search method
public static void search(File indexDir, String q)
throws Exception {
Directory rmDir = RMDirectory.getDirectory(indexDir, false);
IndexSearcher is = new IndexSearcher(rmDir);  //open index


//Parse query
Query query= QueryParser.parse(q,"contents",new StandardAnalyzer());


//Search index
Hits hits = is.search(query);


Document doc = hits.doc(i); //Retrive matching documents

System.out.println(doc.get("filename")); //Display filename
```

Administrator and Connection Controller Unit can do a lot of queries for permanency of the system. For instance, when an user attempt to request an URL, Connection Controller Unit queries whether or not URL is categorized. Therefore, Searcher takes URL and send a query to indexed file. If URL indexed before, Searcher finds URL in index file in a constant time and returns its' URL and category name. After Searcher find them, send them to Connection Control Unit. Eventually, Connection Control Unit decides to respond of request according to user group and other policies.

3.4. CATEGORIZER UNIT

3.4.1. Text Similarity Calculator

In Categorizer Unit there will be a Document Similarity Calculator Module,
It will be implemented using Indexer and Searcher Modules described above.
These modules will be used to find out similarity between two text documents.

We also have important assumption about this part according to our investigations:

Assumptions

1. Interfaces to implement important similarity criteria.
2. Interfaces to access document term vectors in the indexes.

All these assumptions are bases for us during our design. About these concepts and technologies we do not have enough knowledge but after again **Apache Lucene API** seems to be helpful while implementing this module, it has expert packages which have methods to access term vectors.

3.5. CONCLUSION

This section is written to make all the parts of the MindGate Software Architecture clearer. External dependencies are important parts for a Software Project and MindGate Project contains many external components which easily lead project to failure. To avoid this we try to identify all the internal and external dependencies project has. All design related diagrams clearly describe internal dependencies, and all external dependencies described in this section to fill the gap.

4. FILE FORMATS OF MINDGATE SYSTEM

MindGate Web Filtering Software will need different kinds of data which are used for different purposes and by different units of the system while it is running. To make system concurrent, stable and efficient all this data should be stored and retrieved very fast. This means we need a very flexible and reliable data organization.
In this section of the initial design report describes the organization of all these data according to different files and their formats.

4.1. LOG FILES

This subsection describes the log files MindGate creates and stores while running.

4.1.1. Systems Logs

The logs in this category are about MindGate System. They will be used to find out the reason of an error or an unconditional situation.
An error can be occurred anywhere in the system while it is running, so each possible action should have an error code which will be unique in the system.

```
<?xml version="1.0" encoding="ISO-8859-1"?>

<!--
                        MindGate System Logs File
Description:

1.This file will contain information about system, these information
can
  be errors occurred while system is running or about status of the
system.

2.These file can be searched with queries such as:
  2.1. Query = (logtype AND timeperiod) -> (errorcode|statuscode|.. , )
-->
<systemlogs>

      <errorlog>   <!-- logtype -->

      <errorcode>reading_check_list_entry_unsuccessful</errorcode>
            <occurrancetime>02112005:1212</occurrancetime>
            <description>IO error</description>
      </errorlog>
      <statuslog>
            <statuscode>system_stop</statuscode>
            <occurrancetime>02112005:1243</occurrancetime>
            <description>system is stopped</description>
      </statuslog>

</systemlogs>
```

4.1.2. Admin Logs

These logs will contain detailed information about administrators' actions and effects of these actions to the MindGate System.

Each action which administrator can perform will have an action code which is unique in the MindGate System.
Each action will have a description which describes its intended behavior in context of action.

```
<?xml version="1.0" encoding="ISO-8859-1"?>

<!--

                        MindGate AdminLogs File
Description:
1.Logs can be listed according to time period and adminname.
period:range of two occurance times

2.Support query searches on the file such as:

 2.1 Query = (adminname  AND actiontype AND period)
 2.2 Ouery = (adminname  AND actiontype AND username)

3.Each log's action can be displayed when mouse click on.

-->

<adminlogs>

      <admlog adminname="berkank" >

            <action actiontype="add_new_user"
occurtime="02112005:1136">
                  <addnewuser username="tolgak"  groupinfo="it" >
                  </addnewuser>
            </action>

            <action actiontype="delete_user" occurtime="02112005:1146">
                  <deleteuser username="ozgurbtr">
                  </deleteuser>
            </action>

            <action actiontype="update_user" occurtime="02112005:1158">
                  <updateuser username="kerimk"  groupfrom="management"
groupto="it" >
                  </updateuser>
            </action>

      </admlog>

</adminlogs>
```

4.1.3. Connection Logs

These logs will be generated from the connections MindGate handles while running in gateway of LAN.

Two different kinds of logs can be generated which can be classified under two different names : Attempt Logs, Session Logs.

1. Attempt Logs

These logs will contain information about web access attempts. These logs will be used to extract statistical information about users' internet usage.

```
<?xml version="1.0" encoding="ISO-8859-1"?>

<!--

                        MindGate Attempt Logs File
Description:

1.All logs can be listed.

2.Support query searches on the file such as:

 period:range of two occurance times

 2.1 Query = (username  AND url)    ->   list (attemptstatus,
attempttime)
 2.2 Ouery = (username AND period)  ->   list (url, attemptstatus,
attempttime)
 2.3 Query = (url AND period)       ->   list
(username,attemptstatus,attempttime)
 2.4 Query = (machineip AND period) ->   list (username,attempttime)
for security
 2.5 Query = (username AND status AND period ) -> list(url,attempttime)

 and also many other complex query searches can be done on these file.

3.These query searches can be saved to another file or send to a
printer.

-->

<attemptlogs>

      <attempt>
            <machineip>144.122.112.201</machineip>
            <username>berkank</username>
            <url>www.google.com</url>
            <attemptstatus>successful</attemptstatus>
            <attempttime>02112005:1132</attempttime>
      </attempt>

      <attempt>
```

```
            <machineip>144.122.112.141</machineip>
            <username>ozgurbtr</username>
            <url>www.ogame.fr</url>
            <attemptstatus>unsuccessful</attemptstatus>
            <attempttime>02112005:1145</attempttime>
      </attempt>

</attemptlogs>
```

2. Session Logs

These logs will be generated when user of logged in to the system and logged out of the system.

```
<?xml version="1.0" encoding="ISO-8859-1"?>

<!--
                        MindGate Attempt Logs File
Description:

1.All logs can be listed.

2.Support query searches on the file such as:

 period:range of two occurance times

 2.1 Query = (username  AND period)     ->   list (sessionaction,
machineip,  actiontime)
 2.2 Ouery = (machineip AND period)     ->   list (username,
sessionaction,  actiontime)
 2.3 Query = (sessionaction AND period) ->   list
(username,machineip,actiontimetime)
 2.4 Query = (period)                   ->   list
(username,machineip,sessionaction,actiontimetime)



 and also many other complex query searches can be done on this file.

3.These query searches can be saved to another file or send to a
printer.

-->

<sessionlogs>

      <session>
            <machineip>144.122.112.201</machineip>
            <username>berkank</username>
            <sessionaction>login</sessionaction>
            <actiontime>02112005:1132</actiontime>
      </session>

      <session>
```

```
            <machineip>144.122.112.141</machineip>
            <username>ozgurbtr</username>
            <sessionaction>logout</sessionaction>
            <actiontime>02112005:1145</actiontime>
      </session>

</sessionlogs>
```

4.2. INDEXED FILES

Files in this category will be indexed to achieve fast access times. They will be used by system components to control and filter the LAN web traffic. Detailed description of the indexing methods given in the File Indexing Section of this document, but small descriptions will be given to make the context clearer.

4.2.1. CheckList

```
 <?xml version="1.0" encoding="ISO-8859-1"?>

<!--

                        MindGate Check List File


Description:

1.System will use this file, it will reside in the memory while system
is running.

2.On demand file can be listed.

3.
   given  (url)   ->   (category)
   given  (url)   ->   (hit)

4.Backup of the file can be stored on demand.

-->

<checklist>
      <url>
      <!-- only domain name -->
            <name>
            www.ceng.metu.edu.tr
            </name>
            <category>
            education
            </category>
            <hits>
            45
            </hits>
```

```
        </url>
        <url>
        <!-- sub directory of a domain implicite index.html -->
                <name>
                www.ceng.metu.edu.tr/courses  <!--/index.html-->
                </name>
                <category>
                education
                </category>
                <hits>
                45
                </hits>
        </url>
        <url>
        <!-- with index html-->
                <name>
                www.ogame.fr/index.html
                </name>
                <category>
                education
                </category>
                <hits>
                45
                </hits>
        </url>
</checklist>
```

4.2.2. GroupInfo Table

```
<?xml version="1.0" encoding="ISO-8859-1"?>

<!--

                        MindGate Group Info Table File

Description:

1.System will use this file, it will reside in the memory while system
is running.

2.On demand file can be listed.

3.
   given  (username)   ->   (category-list)
   given  (category)   ->   (user-list)

4.Backup of the file can be stored on demand.

5.Extensive search can be done on the backuped file.

-->
```

```
<groupinfotable>
      <group>
            <name>
            teacher
            </name>
            <category-list>
            pornography,gambling,violence
            </category-list>
            <username-list>
            tolgak,berkank,ozgurbtr
            </username-list>
      </group>
      <group>
            <name>
            student
            </name>
            <category-list>
            pornography,gambling,violence,game
            </category-list>
            <username-list>
            e1232424,e1314355,e1243455
            </username-list>
      </group>
</groupinfotable>
```

4.2.3. Session Table

```
<?xml version="1.0" encoding="ISO-8859-1"?>

<!--
                        MindGate Session Logs File
Description:

1.All logs can be listed.

2.Support query searches on the file such as:

 period:range of two occurance times

 2.1 Query = (username  AND period)     ->   list (sessionaction,
machineip, actiontime)
 2.2 Ouery = (machineip AND period)     ->   list (username,
sessionaction, actiontime)
 2.3 Query = (sessionaction AND period) ->   list
(username,machineip,actiontimetime)
 2.4 Query = (period)                   ->   list
(username,machineip,sessionaction,actiontimetime)

 and also many other complex query searches can be done on this file.

3.These query searches can be saved to another file or send to a
printer.

-->
```

```
<sessionlogs>

      <session>
            <machineip>144.122.112.201</machineip>
            <username>berkank</username>
            <sessionaction>login</sessionaction>
            <actiontime>02112005:1132</actiontime>
      </session>

      <session>
            <machineip>144.122.112.141</machineip>
            <username>ozgurbtr</username>
            <sessionaction>logout</sessionaction>
            <actiontime>02112005:1145</actiontime>
      </session>

</sessionlogs>
```

4.2.4 CategoryInfo Table

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!--
                  MindGate Category Info Table File

Description:

1. This table contains data which will be used by structure analyzer
and content analyzer to categorize the web pages
according to given heuristics.

2. Support query searches on the file such as:

      2.1 Query = (name AND keyword) -> if keyword is available under
this category or not
      2.2 Query = (name AND URL)     -> if URL is available under this
category or not
-->

<CategoryInfo>
      <Category>
            <name>
            gambling
            </name>
            <keywords>
            bet,gamble,cards,casino,toss,win
            </keywords>
            <imweight>    <!--image weight = (  imnum )   /  (txtlen) -
->
            4
            </imweight>
            <linweight>   <!--link weight  =(  linnum ) /  ( txtlen )--
>
```

```
            7
            </linweight>
            <scrweight>   <!--script weight =(  scrnum ) /  ( txtlen )-
->
            10
            </scrweight>
      </Category>
</CategoryInfo>
```

4.3. USER GROUP and CATEGORY RELATION in MINDGATE SYSTEM

In MindGate system there are user, group, and category entities. To avoid confusion between these entities these section was written.

4.3.1. USER

Each user refers to a real person in the organization who has access to the system. These users must admitted that their access to the internet using organization's internet access is not private and will be logged and filtered according to organization's web access policies. In MindGate System a user will be shown as an abstract entity with following attributes:

```
<?xml version="1.0" encoding="ISO-8859-1"?>

<!--

                        MindGate User Info Table File


Description:

1.System will use this file, it will reside in the memory while system
is running.

2.On demand file can be listed.

3.
   given  (username)   ->   (URL-list)
   given  (username)   ->   (groupname)

4.Backup of the file can be stored on demand.

5.Extensive search can be done on the backuped file.
```

```
-->

<UserList>
      <User>
            <username>
             e1347681
            </username>
            <name>
             Kerim
            </name>
            <surname>
             Korkmaz
            </surname>
            <department>
             computer engineering
            </department>
            <groupname>
             student
            </groupname>
      </User>
</UserList>
```

4.3.2. GROUP

Group is a group of users who share same access policies. **If a user in a group can not access a web page another user in the same group cannot access that web page also.** System administrator can adjust and change policies for certain groups without dealing with users one by one. As one can see below, group is not real entity, but just a reference to a category list. In addition, users not collected under group but each user has a reference to that group which is encapsulated in the system and hidden from the administrator.

```
<?xml version="1.0" encoding="ISO-8859-1"?>

<!--

                        MindGate Group Info Table File

Description:

1.System will use this file, it will reside in the memory while system
is running.

2.On demand file can be listed.

3.
   given  (username)   ->   (category-list)
   given  (category)   ->   (user-list)

4.Backup of the file can be stored on demand.

5.Extensive search can be done on the backuped file.
```

```
-->

<groupinfotable>
      <group>
            <name>
            teacher
            </name>
            <category-list>
            pornography,gambling,violence
            </category-list>
            <username-list>
            tolgak,berkank,ozgurbtr
            </username-list>
      </group>
      <group>
            <name>
            student
            </name>
            <category-list>
            pornography,gambling,violence,game
            </category-list>
            <username-list>
            e1232424,e1314355,e1243455
            </username-list>
      </group>
</groupinfotable>
```

4.3.3. CATEGORY

Category is a collection of websites which shares the same access policies. **If a user can access a web page same user can not access a web page in the same category.**
In the MindGate System, a category will be represented as below.

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!--
                  MindGate Category Info Table File

Description:

1. This table contains data which will be used by structure analyzer
and content analyzer to categorize the web pages
according to given heuristics.

2. Support query searches on the file such as:

      2.1 Query = (name AND keyword) -> if keyword is available under
this category or not
      2.2 Query = (name AND URL)     -> if URL is available under this
category or not
-->
```

```
<CategoryInfo>
      <Category>
            <name>
            gambling
            </name>
            <keywords>
            bet,gamble,cards,casino,toss,win
            </keywords>
            <imweight>  <!--image weight = (  imnum )   /  (txtlen) -->
            4
            </imweight>
            <linweight> <!--link weight  =(  linnum ) /  ( txtlen )-->
            7
            </linweight>
            <scrweight> <!--script weight =(  scrnum ) /  ( txtlen )-->
            10
            </scrweight>
      </Category>
</CategoryInfo>
```

4.4. ARCHIVE

Archive is the place where all log files and system data image files will be stored. It contains two parts which are designed for different purposes, this can be seen in Figure 9 which is the base schema of archive area.

Descriptions of these parts are:

4.4.1 Logs

All the described log files in previous parts of this section will be stored to this location in the archive. Log Files will be stored as plain xml files and will be indexed according to needed key-value pairs to make querying easy for administrative purposes.

In general log files will not be needed so frequently then storing them as plain text in xml format seems to be reasonable. But periodically these files will be indexed so administrators of the system could access them easily and will have chance to make query search in them.

4.4.2. System Data

These are resources which MindGate System uses so frequently. They all indexed according to their relevant key-value pairs and all these indexes stored in the RAM to make their access time lesser and lesser.

But unconditional situations can occur while program is running, such as electricity failure or system crash which could lead important collected data to be lost, so there

should be a mechanism to flush these data to the external file system periodically, so location is where this process will occur all RAM images of the system related data will be flushed to this location.

Also these areas can be seen as a collective backup store of the system resources which are updated periodically, in addition to this administrator can trigger system to take a fresh backup of all the used data any time he wants.

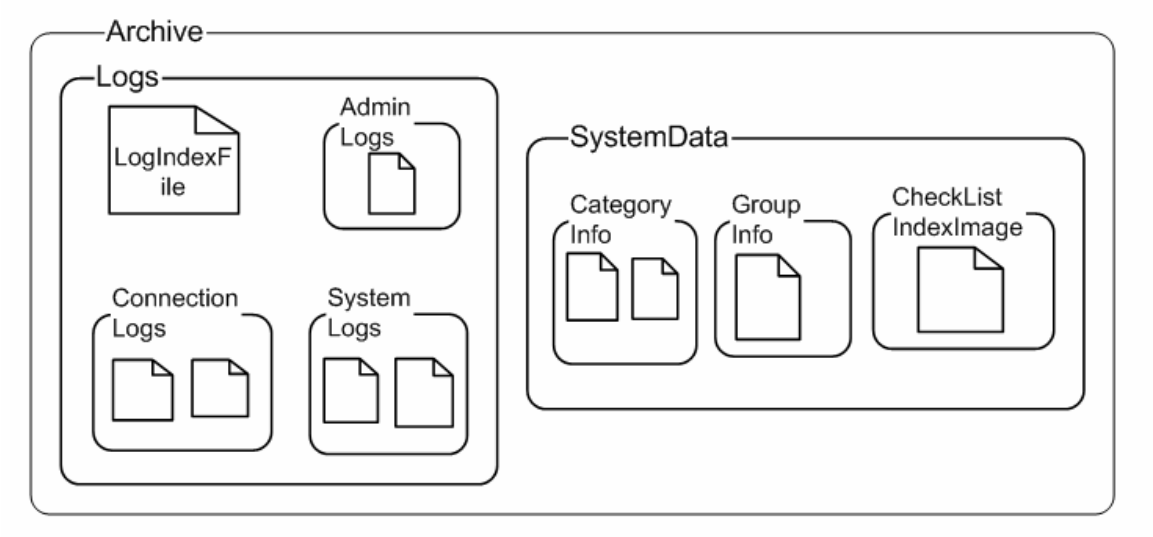


Figure9 – Diagram of the Archive File System

5. REVISION OF INITIAL DESIGN AND ABOUT FINAL DESIGN

We have mentioned about important constraint which should be achieved in the final design in the initial design report. In this section of the final design report we made an evaluation of our process during final design phase.

After initial design, according to feedbacks our instructors provided to us we add sections to initial design report and we check out our project progress again and again. This feedbacks lighten the way of us trough our project.

There were critical goals which should be achieved in the final design of MindGate Project. And after final design we are presenting our what we did in this section.

1. Data Flow diagrams, State Transition Diagrams and Graphical User Interface part of the system will be revised again and again in the final design of the project. Their consistency checked against newly added parts to the overall architecture.

   We revised our dataflow diagrams and state transition diagrams in the final design phase and we see that there exists no inconsistencies in dataflow diagrams and state transition diagrams according to our internal system architecture.

2. In the initial design all Data Flow Diagrams and State Transition Diagrams drawn carefully and they were revised to correct the errors in it. Also for each diagram all the process described in detail. In the final design all class diagrams will be drawn and by using these diagrams and their definitions.

   All the class diagrams of the MindGate Project are included in the final design report, which make the system's description more clear and more concrete.

3. External components of the MindGate System which are included in the architecture are clearly specified in initial design by making crucial assumptions which are based on our extensive search about those technologies. In the final design all these components and their interfaces to MindGate Software will be described in detail.

   External components in the project tested and some exemplifying software developed by the members of the team to become familiar about their technologies .These external components checked against our assumptions we make on them and we have not encountered an inconsistency, they seem suitable for our project.

4. File formats described in this report will be refined again and their xml formatted last versions will be prepared in final design phase of MindGate Project.

All the described file formats are included in the final design report to ease implementation phase and to make things more clear for us during implementation of MindGate.

5. Indexing and searching mechanisms will be designed in detail during final design phase and their performance estimations will also be considered precisely, to make them faster.

   MindGate does not depends on an SQL based server to it will contain its own hand written database which will based on XML technology to reduce the time consumed during insertion and querying processes. So it will contains special indexing and searching utilities for all the files it contains(internal & external). How these will be achieved also described in final design report.

6. ABOUT WEB PAGE CATEGORIZATION

This section of the initial design document describes the methods and techniques which will be used to categorize web pages in MindGate WebFiltering Software.

We made search about Data Mining Concepts in analysis and design phases to construct our own techniques about web page categorization. In these to phases we examined many techniques used to classify and categorize text documents and also we examined papers which are related to categorization of web based documents.

After constructing a base knowledge about text classification and web based document categorization we construct our web categorization mechanism which will run in MindGate System to categorize the web pages.

6.1. ABOUT TEXT CATEGORIZATION AND DATA MINING

In the analysis phase of the project we made extensive searches about text categorization techniques used by expert systems in the world.

In Classic Text Categorization there are many techniques which can be applicable for our project. Two of these techniques are Bayesian and Vector Classifiers which are based on statistical methods.

These methods used in practical applications and also there are many libraries which implements these techniques in the world. But all these methods need huge and carefully collected training data sets to make categorization correct.
This is one reason why we did not choose to apply these methods. But our main reason for not choosing these text classification methods is about categorization methods which a web content filter needs.

In reality categorizing a text is a hard job because generally text which is to be categorized in an unstructured form, then it needs preprocessing, such as feature selection and stopping word elimination to reduce the complexity in it before categorization process begin. But on the other side web pages has a semi structured format which contains many important clues for about their contents which is an important point for web page categorization and there is many different characteristics web pages have. All these lead us to approach web page categorization from different aspects.

6.2. MindGate Web Page Categorization Mechanism

At first we started to examine the web pages and their underlying structure, then we find out that while we are trying to categorize a page we find important points, titles, emphasized sections, pictures, symbols and so on. All these mainly grasp our attention.

At the end we scan the page with our eyes and make a decision about page's category, all these happens when a human tries to category a page without understanding the page content. This was an important base for our web page categorization method, because we do not have an intelligent tool which can understand a text.

Addition to this many web pages contain meta-information in these sources to increase their rank in the search engine indexes to get more hits than other pages in the world.

With this logic we arrive some criteria about web pages and their categories:

1. Web page titles are important keywords about categorization this is also same for text.

2. Words which are emphasized in the page with bigger fonts and different colors are also important clues for the overall document.

3. Images and their distribution in the web pages are also contain an information about a web page.

4. Number of links and their distribution in the page is a good clue.

5. Words in these links can be important keyword about the content in the page.

6. Meta-information exists in the page source contains keywords which are relevant to page content.

7. Number of scripts in the page reveals information about page category.

Of course all these criteria are so generalized and will fail in some situations, but from the side of the application we are trying to develop it can easily be used.

A product such as MindGate will generally used to block sites containing pornographic, violent or gambling content, and will allow sites such as library web pages or school home pages. So trying to categorize all the text in a page is nonsense in many situations, and also many times pages which should be categorized correctly will contain very little or no text in it.

6.2.1. Description of the Mechanism

To categorize the page we will extract some information from the page and create a structure from it which has these attributes:

title = Page Title
meta = Page Meta-Information
imnum = # of images in the page

txtlen = length of the text in the page
linnum = # of links in the page
scrnum = # of scripts in the page
empwords = list of the emphasized words in the text, which have bigger font or different colors.

This information gathered from the page will be used to assign a category to a page. Assigning a category to a page will be done in three steps which are:

1. Structural analyzing
2. Content analyzing
3. Page categorizing

1. Structural Analyzing

In this phase frequencies will be calculated

imfreq = (imnum) / (txtlen) txtlen != 0
linfreq = (linnum) / (txtlen)
scr = (scrnum) / (txtlen)

if txtlen == 0 then a frequency will be set to 1

These parameters will be inserted into heuristic equations which are stored in the system for each category such as

structural_result =

imageweight * imfreq + linkweight * linfreq + scriptweight * scrfreq

These weighted numbers will be given when a category added to the system in some known range by the administrator of the system.

Example :

Category name = gambling
Image weight = 6 (0-10)
Script weight = 8 (0-10)
Link weight = 4 (0-10)

After evaluating all the equations maximum of the structural_result of them will be chosen which means page structure most resembles to category of that equation.

2. Content Analyzing

In the content analyzing phase

meta
empwords
title

information will be used which are extracted from the page. We weighted this information according to some criteria such as;

meta information is the most important one, second is page title and the last one is empwords and then categorization info will be gathered from categorization info table which contains heuristics constants according to predefined categories and word sets for each category.

These word sets will be compared to the ones gathered from page which are empwords set, meta set and title member. This comparison is done with the help of string matching algoritms and result will be normalized between 0-1 to achieve a similarity measure.

3. Page Categorization

Finally results of content analyze and structural analyze are evaluated according to their importance against existing category information in the category criteria table, these information can be supplied by administrator of the system.

7. GRAPHICAL ADMINISTRATOR INTERFACE OF MindGate

7.1. INTRODUCTION

To view general form of software, we designed detailed prototype interface for MindGate in initial design phase. Every possible screen is constructed and all details are included. Before constracting last version of interface, we drew two basic drafts and gave them to our instructor as a progress report. According to feedback from our instructor we have made some logical corrections on our interface.

Our interface screenshots can be seen forthcoming pages. Basically our interface page has two parts. On the left side of the screen, there is navigation tabs which help the administrator to use software easily and affectively.

We choose active navigation tabs as a tool box. Active feature of tool box gives a powerful usage to the interface. Administrator becomes familiar to this tool box easily and can navigate one metu item to other quickly. This method is also advised by papers which are about user friendly interface and human computer interaction.

There are six main tabs and some of them contain subtabs in the toolbox. Tabs and subtabs configured and grouped according to their relevancy, to make learn and use the interface easy.

Subtabs are also composed of functions which display some result in the screen. For example in Figure15 there are three functions; add group, remove group and update group under manage groups profile tab. All these functions could be displayed in one or two screen but we choose three separate screens, because of easily usage of software and not to confuse administrator's mind while performing some job in a long period. All other parenttabs represent a different logical activity. When administrator enters a subtab, he/she can see all the menu from toolbox and easily return back to parent tab.

On the right side of screen there are functions button and input buttons. Right side is pure according to left side of screen, in spite of this contrast, display of screen is plain. Also we located logout button in the MindGate Status tag because of administrator does not want to see every time this button and at the same time this leads system administrator to see the general status of the system again before logging off.

But there are some missing points in graphical interface design which we will deal with later in the final design phase. First of all color of screen is too dense, after a long time sitting in front of this screen exhausts administrator. So we will choose more relaxing color in MindGate. Another one is that there is some screens which we need time to research and decide about them.

As a conclusion, this is only a prototype system which is designed to guide us for the final design of the project.

7.2. LOGIN SCREEN

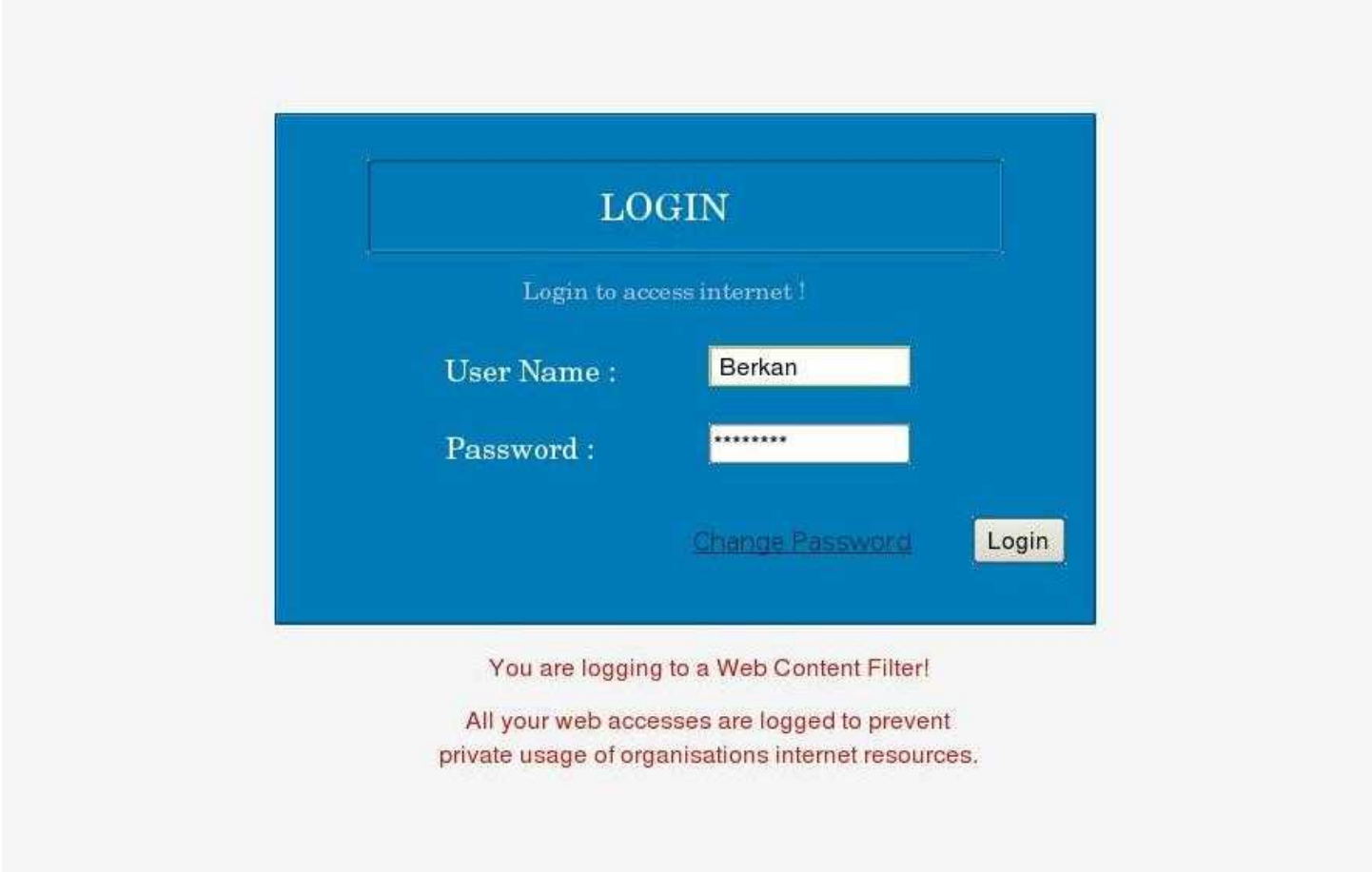


Figure10- MindGate Status Screen

When user attempts to request an URL from a web browser, MindGate examine user status according to machine's IP, if she/he has not logged to the system, software sends a login screen to user. All users and administrators of the system will login to system with this login page.

If username or password is not correct, program shows an error massage and asks username and password again. In order to protect MindGate from harmful software that send a lot of combinations of username and password, MindGate does not display which entity is not correct and wait 2 seconds to send another login screen. If user enters user name or password wrong again, second time MindGate waits 4 seconds to send login screen again. For every consecutive wrong entry from the same Internet Protocol Address, MindGate increments waiting time 2 second.

Also there is a link in this interface which redirect user to change password for changing his/her password.

At the bottom of the login page administrator of the system can include a disclaimer or other messages for the users of the system.

7.3. NAVIGATION TABS

7.3.1. MindGate Status

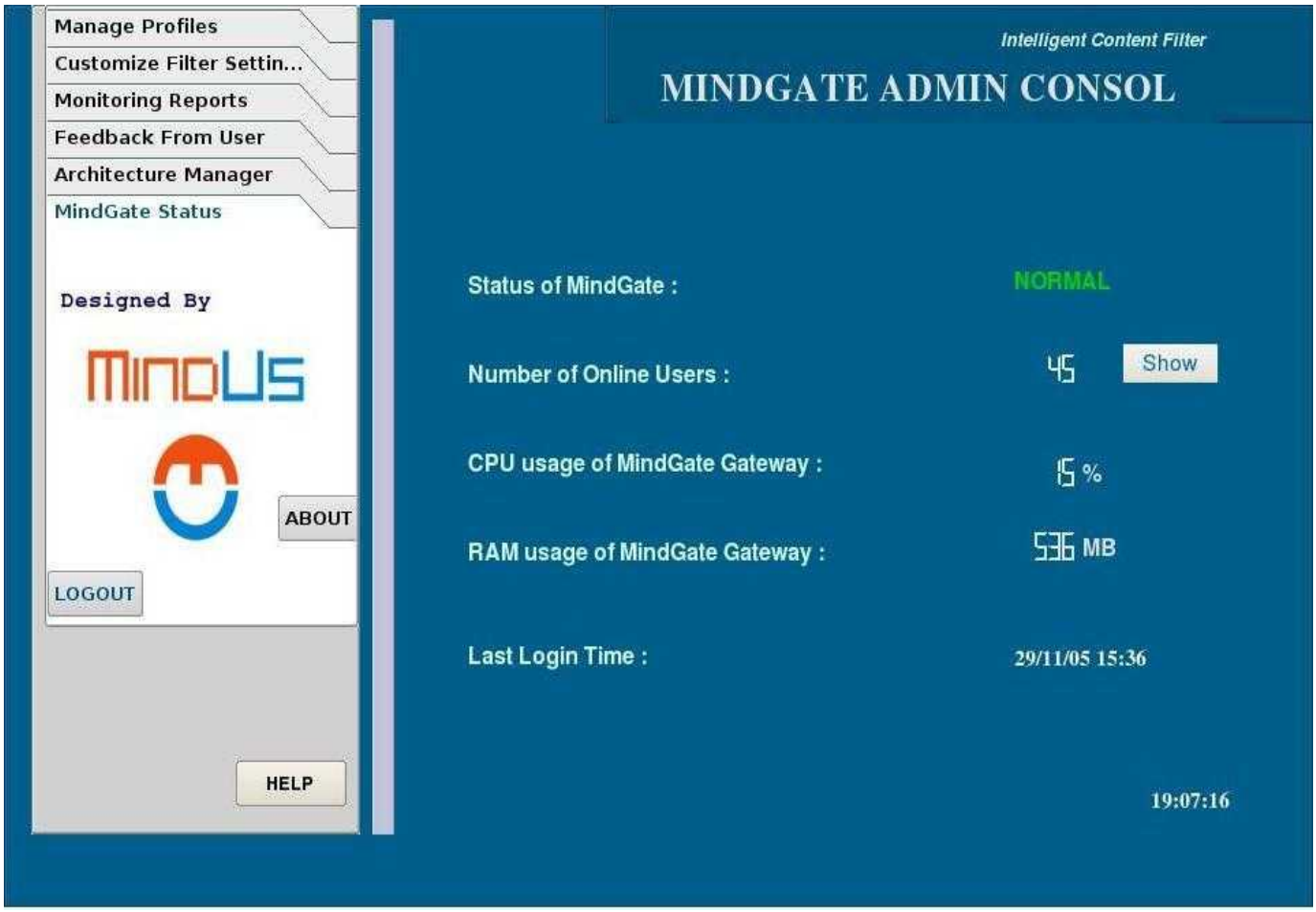


Figure11- MindGate Status Screen

After administrator login to Administrative Console of the system, he/she encounters with MindGate System Status Screen. On the left side of administrator consol there are navigation tabs which will provide access to other management configuration screens.

When administrator clicks to MindGate Status tab, main screen is displayed and administrator can see status of the overall system (Figure11). On the screen there are 5 messages. The first message is the 'Status of MindGate'. In this figure status of MindGate is NORMAL and its color is green. This means every thing is normal and content filter is running properly. Other situations are explained in Figure30. Main screen also shows number of users who are online and administrator can display them. Also administrator can see CPU and RAM usage of the MindGate Gateway Machine and if there is an unpredicted situation occurs in the machine MindGate running on, MindGate warns the administrator. If there is no critical situation, MindGate displays no warning massage. Last administrator login time is shown at the bottom of main screen.

7.3.2. Manage Profiles Screen

Manage Users Profile

Figure12- Add User Screen

Manage Profiles tab is on the top of software navigation tabs and it contains 2 subtabs which are called Manage User Profile and Manage Group Profile. Every user and group assignments are made in this section. Administrator can add users and groups to MindGate and also arrange their properties in Manage Profile tab.

Our first screen shown in the figure is Add User function. In this screen, administrator assigns a 'User Name' to user that can be an ID number or another unique string. And also administrator enters user's 'name', 'surname' and 'department' and selects user's group. MindGate binds user to a group and applies this group's filtering settings to this user. Administrator assigns a password to user and user gets his/her password from administrator.

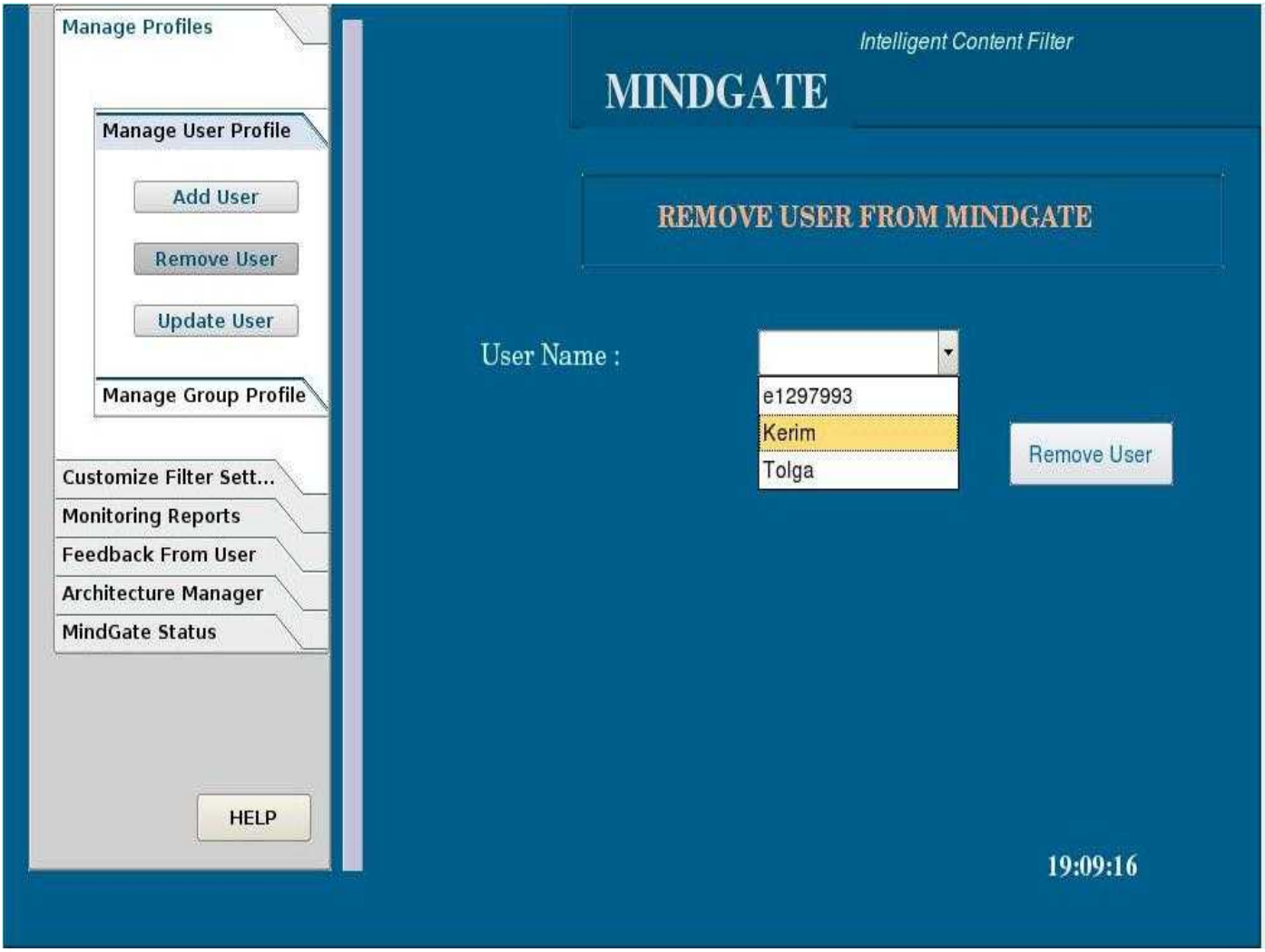


Figure13- Remove User Screen

Figure13 shows 'Remove User' function and in this function administrator choose user from a combobox to remove from MindGate user database. With this activity, every log and data of user are removed from MindGate.

Figure14- Update User Screen

In the 'Update User' screen, administrator can update features of user and assign new password. If there would be some cases such as forgetting password, user asks to administrator to change his/her password and administrator easily changes it. And then user can change password in the login window with clicking change password link which redirects user to password change window. With assigning new group to user, filtering setting of user is changed and new filtering is done according to new groups' categories.

Manage Groups Profile

Groups are user sets that are classified according to their status in the company. All users are under a different kind of group and changing a group's features affects filtering setting of users under that group.

Groups contain a group name and a blocked category list. When a category is assigned to a group, MindGate prevents users of this group to access websites which contains content categorized under this category.

Figure15- Add Group Screen

In the figure15, 'Add Group' function under manage group profile tab is shown. If new group is required in the company, this function gives a way out to solve this problem. Administrator creates a new group, gives a name to it and assigns categories to this group with clicking buttons near the categories. And then administrator assigns this group to

new added users or existing users. Their all filtering staff is done according to these categories.

Figure16- Remove Group Screen

In this screen removing an existing group is shown. When a group is removed from MindGate, users under this group are assigned to ungrouped for which all categories are forbidden. And then administrator assigns them to a new created or existing group or leaves them ungrouped.

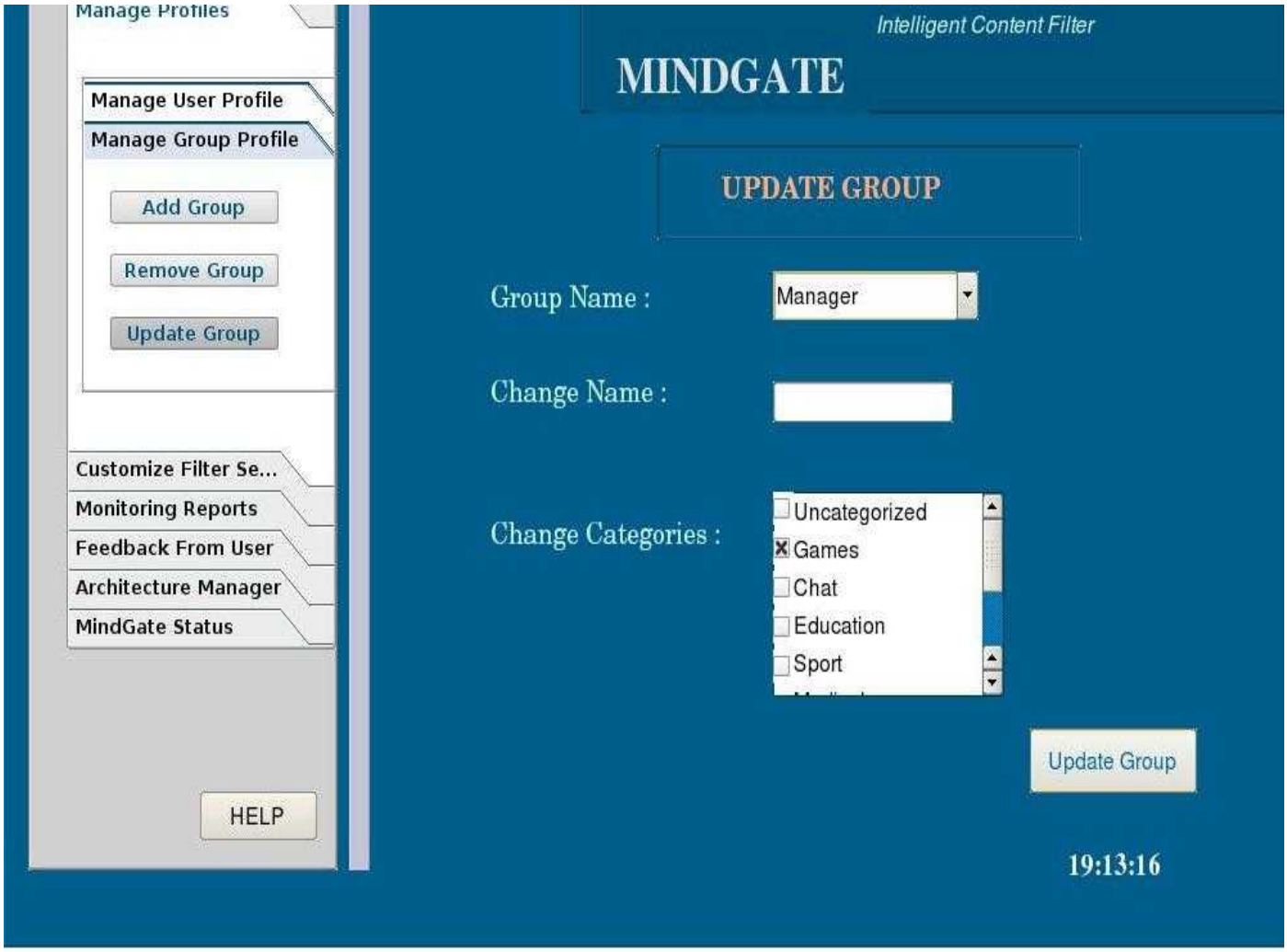


Figure17- Update User Screen

With 'Update Group' function, existing group can be renamed and all old entries in the database are updated with this new name. Also new categories can be assigned or existing categories can be removed with the help of this function. For example, when a group is blocked for all categories, administrator can change this configuration and give them unlimited access. So users under this group are not applied any filtering categories and they can access all websites.

7.3.3. Customize Filter Setting

In this navigation tab, administrator can change filtering settings of MindGate and check situation of lists. All filtering activities are done according to groups and categories which have been bind to them. So changing categories of group affects filtering activities of users. Using this tab accordingly, affects MindGate achievement directly. Administrator should be aware of importance of this section and know all his/her errors can be originated in this section. So we will give more attention to build help documentation of this section.

There are two subcategories under this tab; categories and check lists.

a) Categories

Administrator deals with categories and their features in this fragment. Categories, which are filtering methods' most important parts, are controlled from this tab with three basic functions: 'Add Category', 'Remove Category' and 'Update Category'.

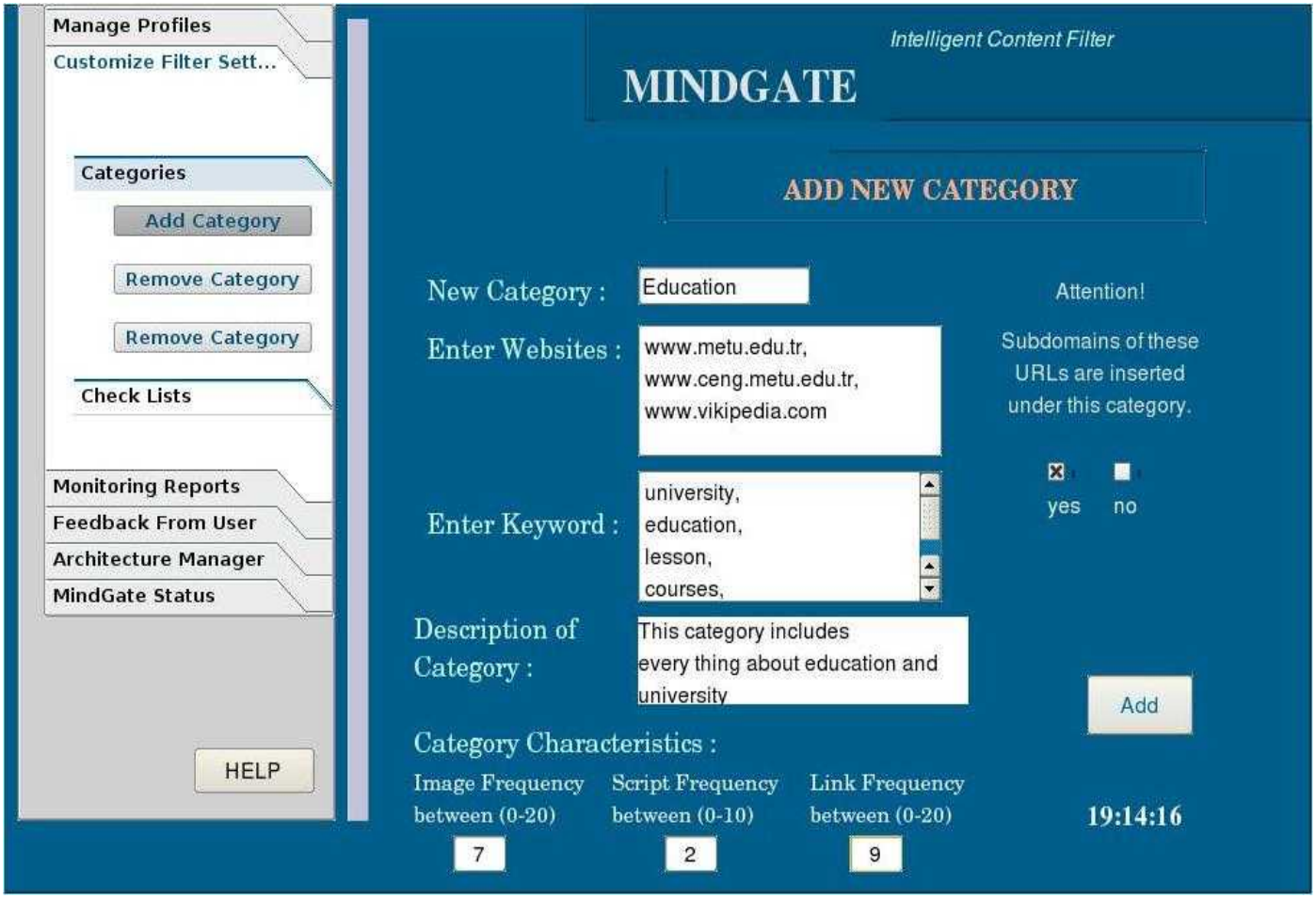


Figure18- Add New Category Screen

In Figure18, 'Add Category' function is shown with the example of adding a new category called Education. Administrator should enter a category name. If administrator wants to create a new category without a name, MindGate gives a warning to administrator. Administrator can add websites to this category while creating it. So software will be aware of these websites, when user request them even thought for first time. Also there is an attention message which asks to administrator that subdomains of websites are to be inserted to this category or not. If administrator clicks yes button, all subdomains are inserted into this category or if he/she clicks no all subdomains are analyzed when user enters to these subdomains and categorize according to their content. Administrator enters keywords for this category and MindGate applies filtering activity on websites according to these keywords. Categorizer in MindGate, categorizes webpage's content according to these keywords and category characteristics. There is also a description of category that describes properties of the category. When administrator leaves from company, new administrator could easily understand categories with these descriptions, and these descriptions can be used to inform users about a blocking situation.

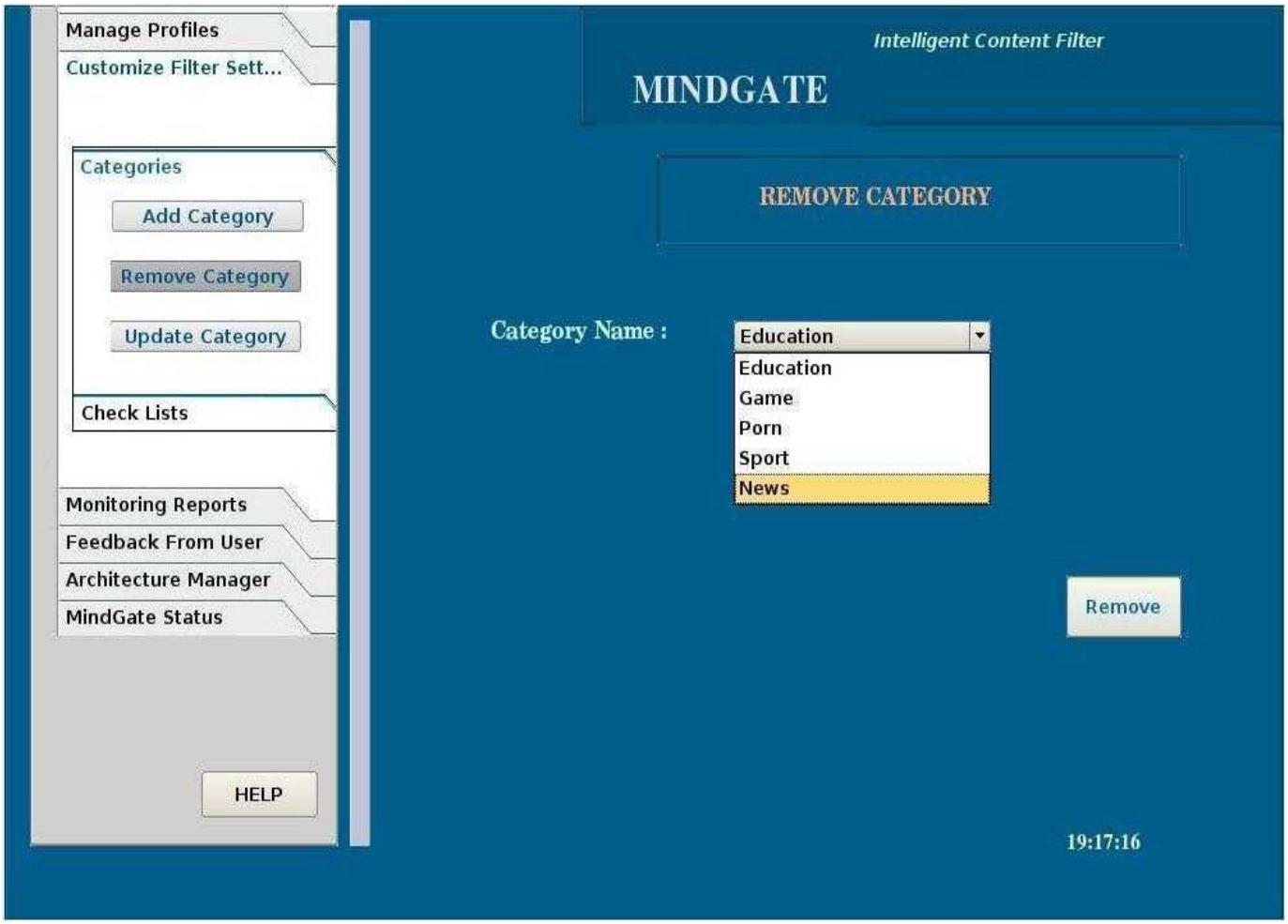


Figure19- Remove Category Screen

In Figure19, removing a category from the MindGate is shown. When a category is removed, URLs of this category, keywords, description, characteristic information of that category removed from database. After all, MindGate will not consider removed category while filtering web content.

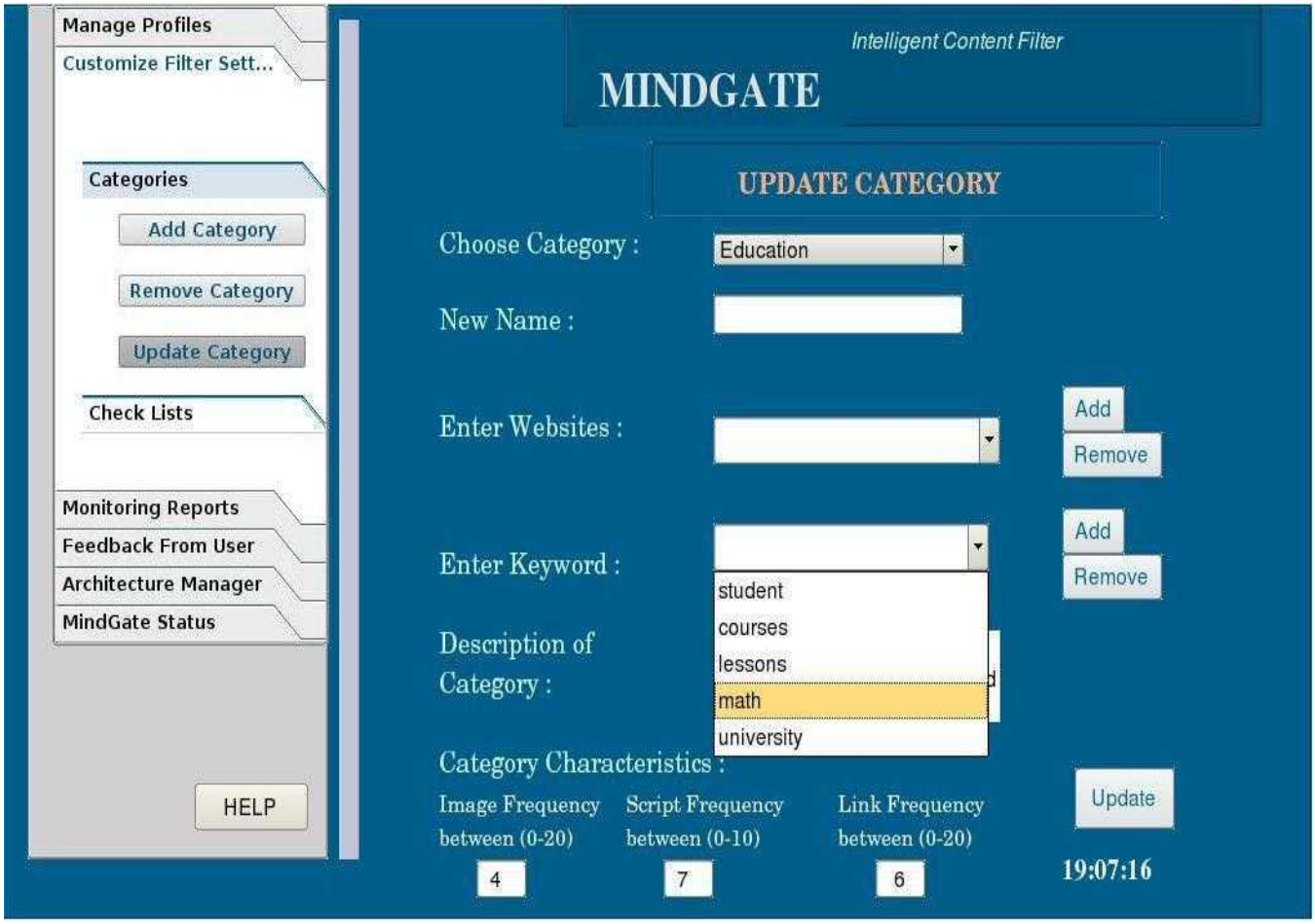


Figure20- Update Category Screen

In Figure20 updating a category is shown. With update category function, administrator can change name of the category. New websites can be added or removed from category URL list. Also here MindGate asks administrator while adding new URLs, if subdirectories are to be added to category or to be analyzed when it is requested, there are yes or no click buttons and according to administrators' choice subcategories are classified. MindGate also asks administrator if he/she wants to remove the subdomains of this URL. If administrator clicks yes, software remove all categories from URL list, otherwise subdirectories of URL are not removed. Most important part of update category function is 'Add' and 'Remove Keyword' parts. With combo input button, administrator can enter keyword, in the case of keyword is in the keyword list, administrator can remove it. In another case he/she add keyword to keyword list. If administrator modifies keywords, and characteristic then category gains a different property, and he/she can form a new description about category.

b) Check Lists

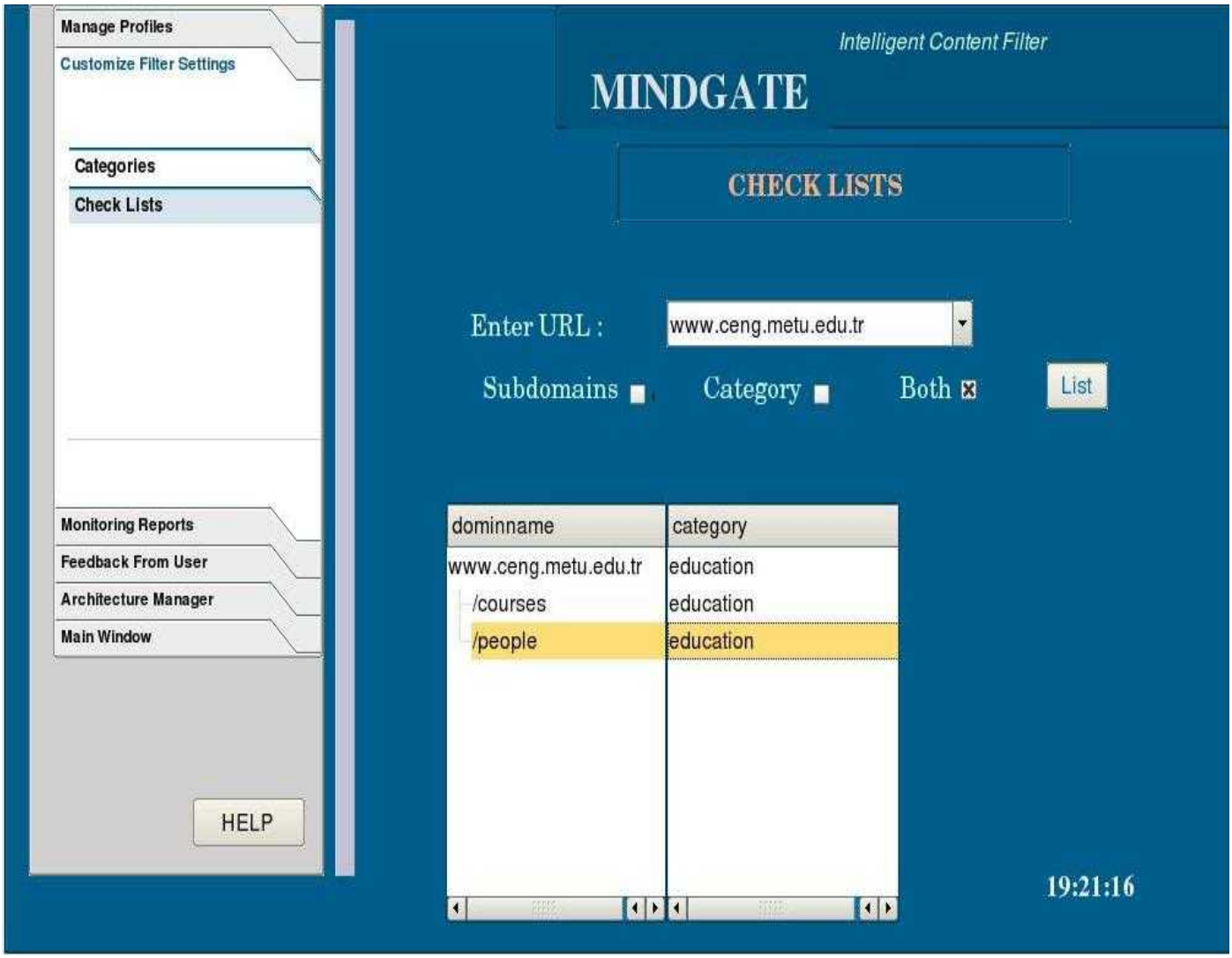


Figure21- Check Lists Screen

Check-lists is second sub tab of customize filter settings and is used for checking URL list. Administrator enters the website domain name or if he/she can not remember full name combo box helps administrator to complete domain name. Before checking domain name, administrator should choose subdomains, category or both of them to display. When administrator chooses only subdomains, software shows subdomains of URL, which have been requested before. In another case administrator chooses category, only category of URL is displayed in the screen. In the figure administrator choose both of them and MindGate displays subdomains of URL and their categories.

7.3.4. Monitoring Reports

One of the most important goals of MindGate is giving correct and detailed statistics of internet usage of users in the organization. These statistics are significantly important for companies that spend too much money for internet usage. Because of this reason, MindGate can be preferable than other software due to its powerful reporting facilities.

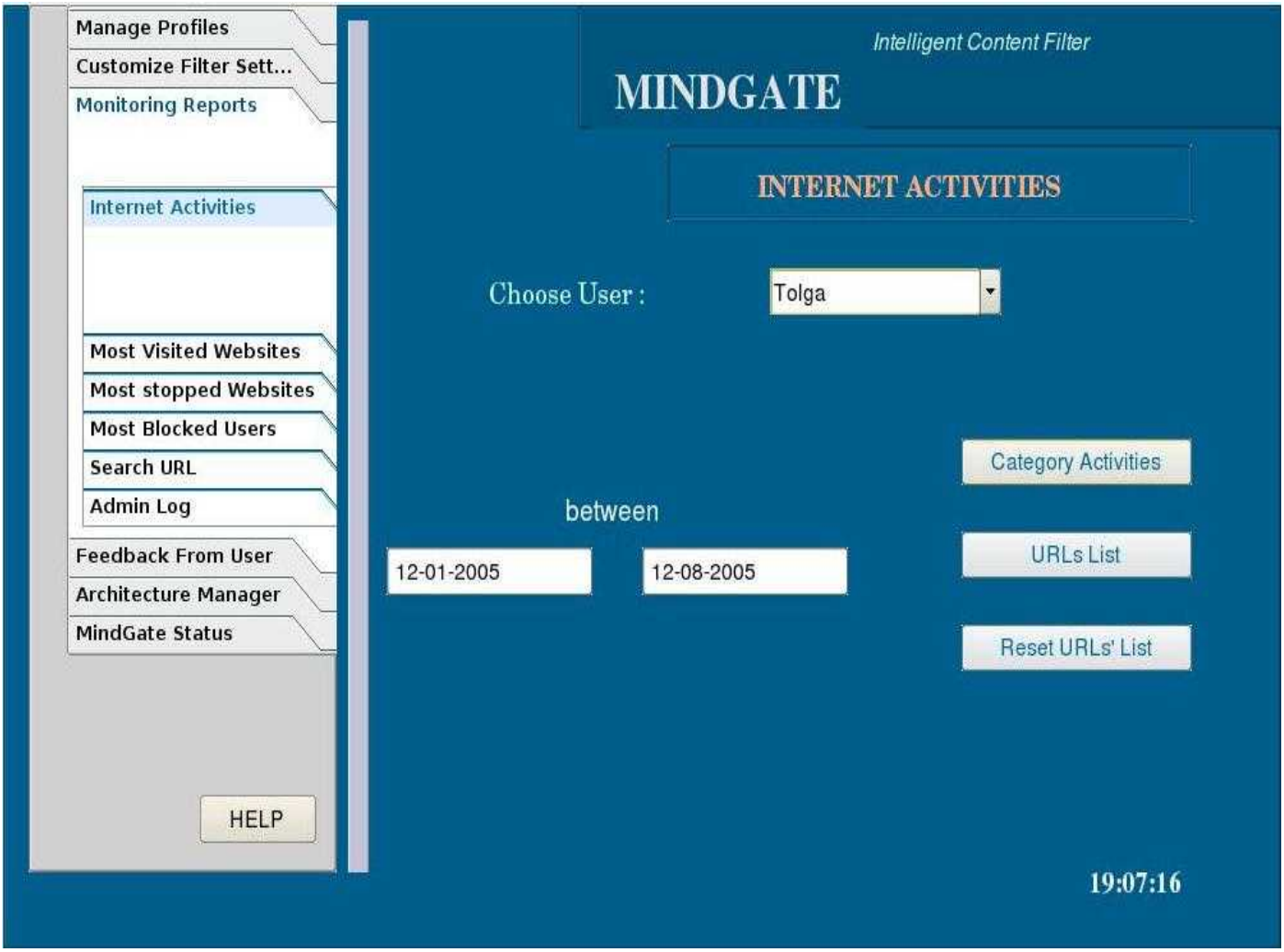


Figure22- Querying Internet Activities Screen

MindGate has capability of displaying all users' web usage statistics at one instance as a single screen or display every user's web usage separately. Administrator enters time interval and username for the query. There are two queries; category activities and URL list. Category activities display which category user has requested and display screen show results as a graph. Figure23 shows an example of category activities query. URL list query displays a URL list with requests made by user in the given period. After administrator examines results, he/she can reset statistics list.

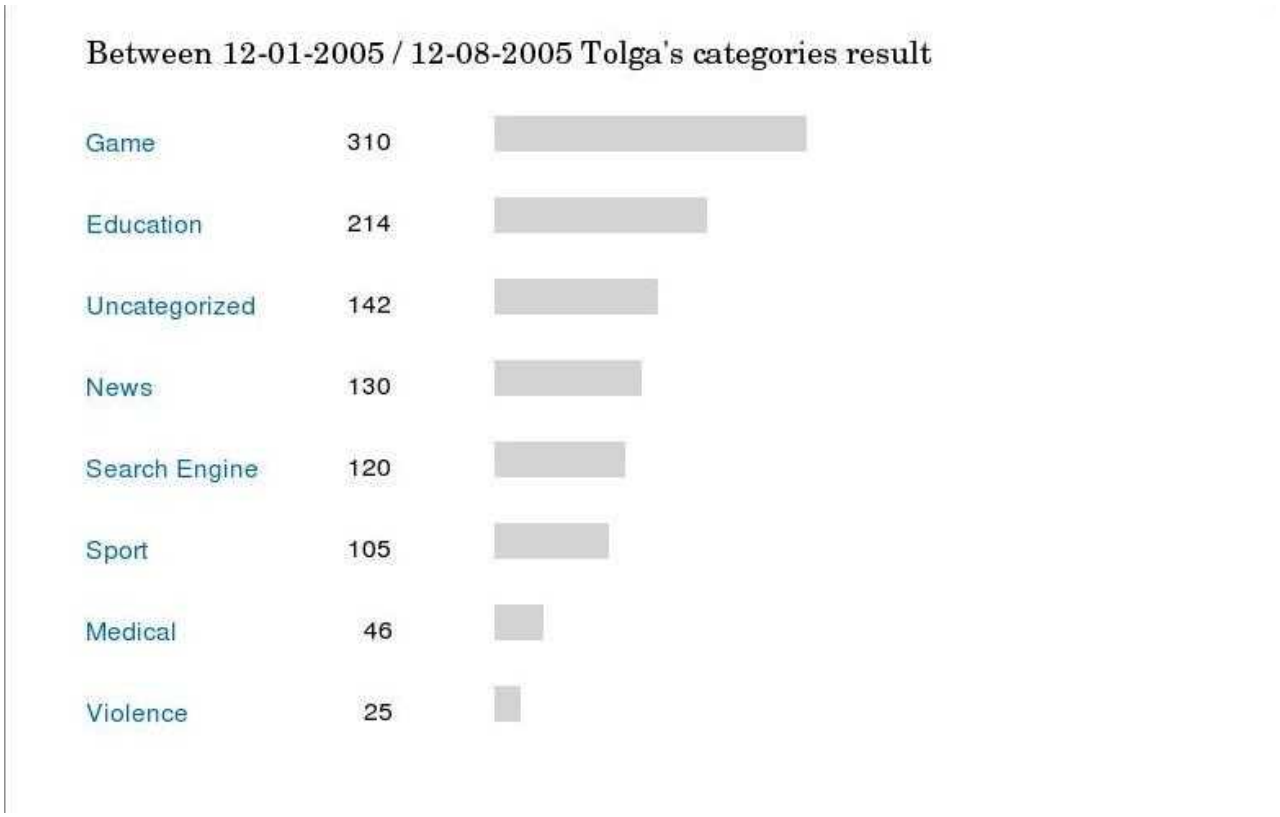


Figure23 - category result of user internet activities between a time period.

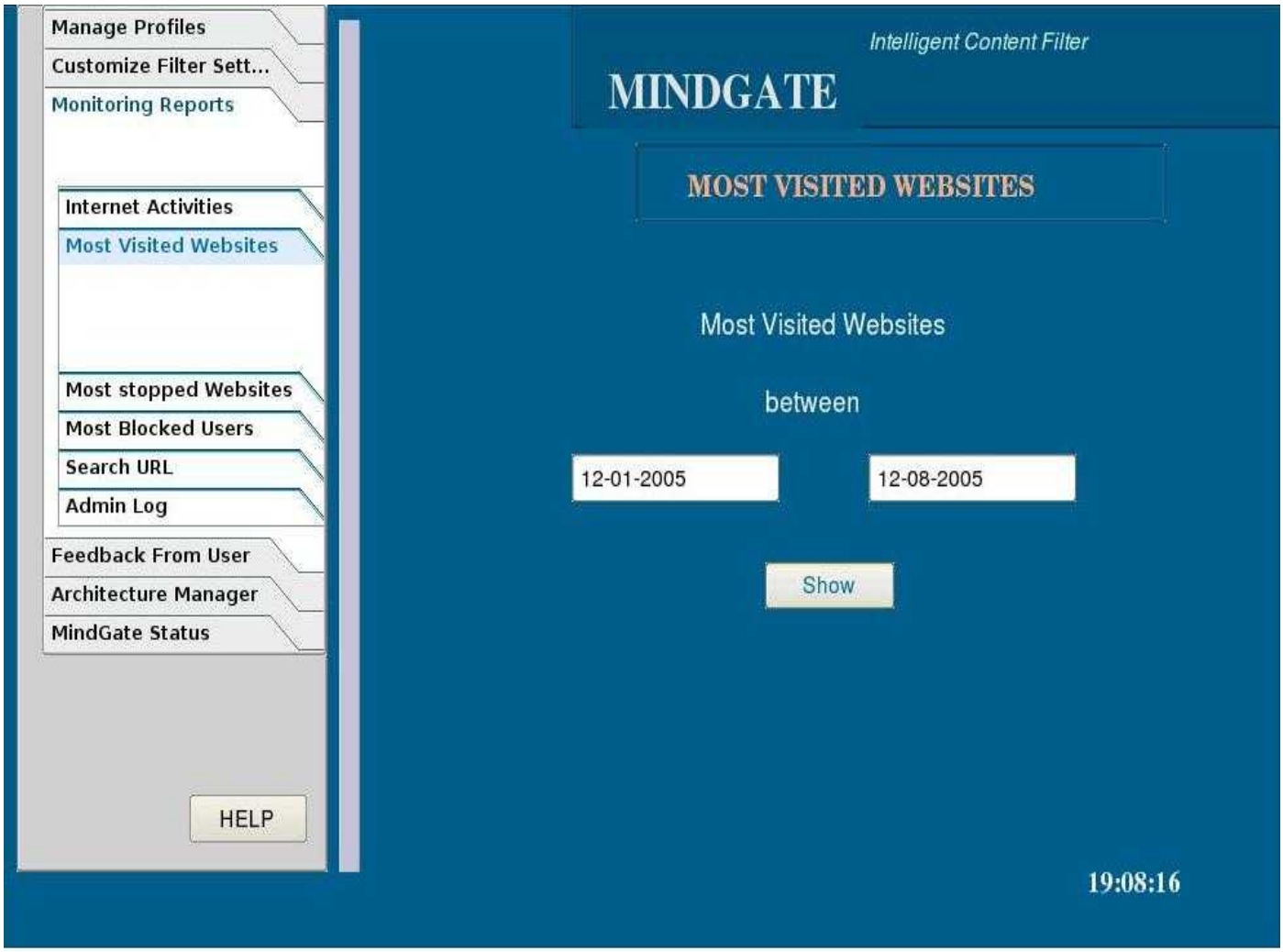


Figure24 - Querying most visited websites screen

Figure24 shows most visited websites of users between a time period. With this property, administrator can learn which websites are requested most and inclination of student or company staff. Especially in the school, this function can be appreciated by school directors.

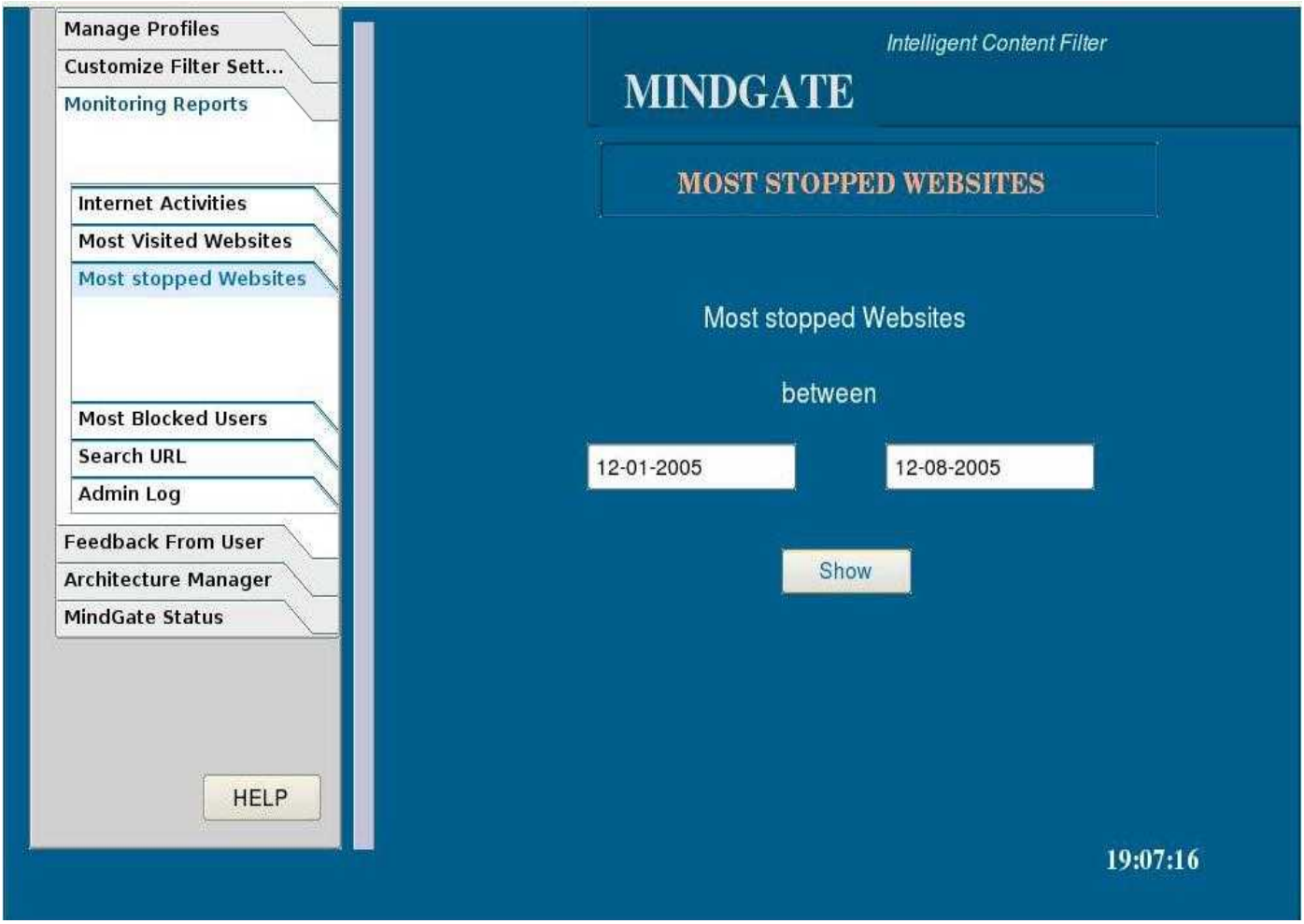


Figure 25- Querying most stopped websites screen

Most stopped website function is also important for companies and schools. This function's result shows achievement of MindGate about blocking of undesirable content.

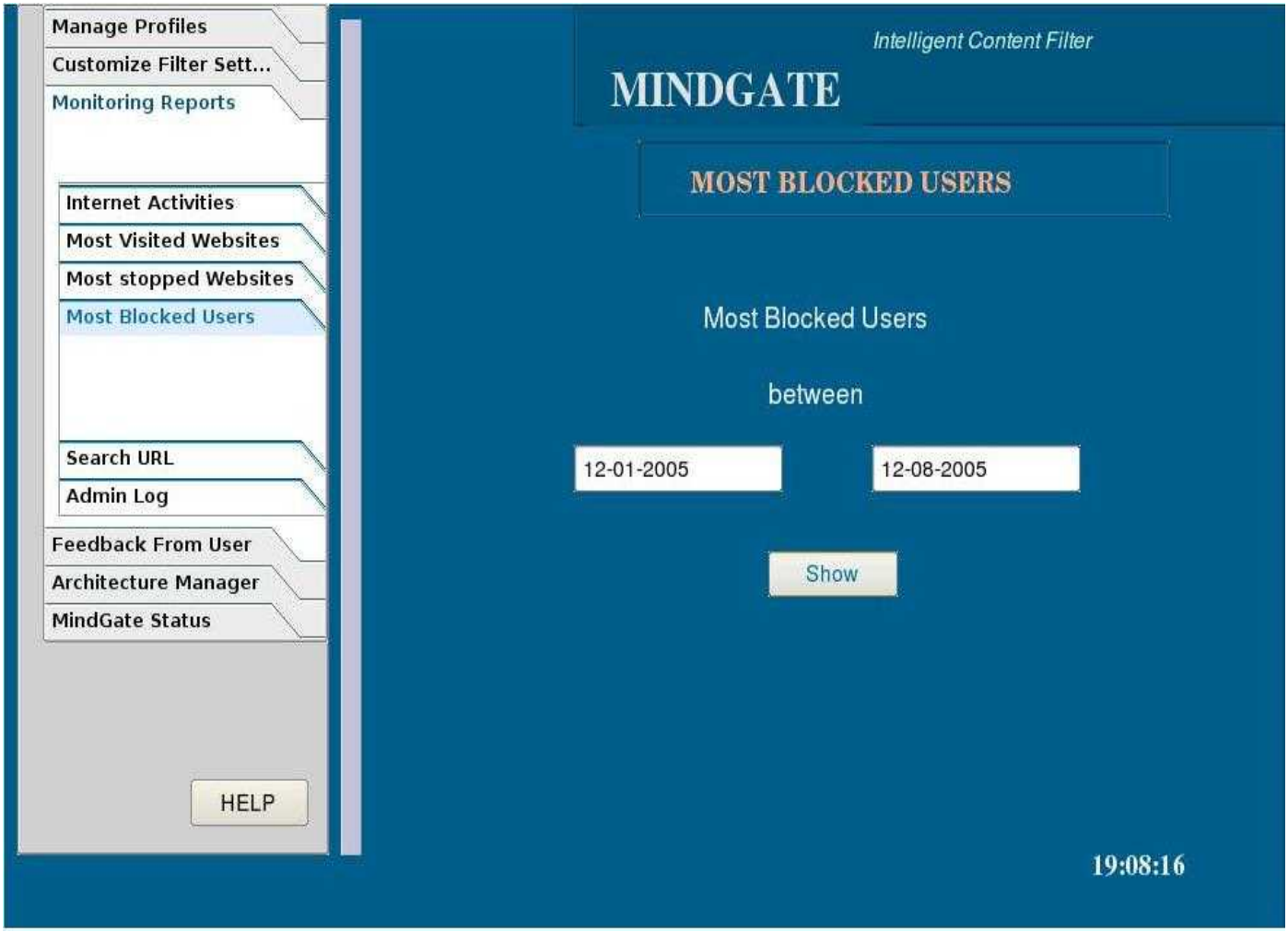


Figure26- Querying most blocked user screen

This function display users who disregard private usage of organization's internet resources. Control authority of administrator would increase in the companies and because of this property demand of MindGate would increase in the market.

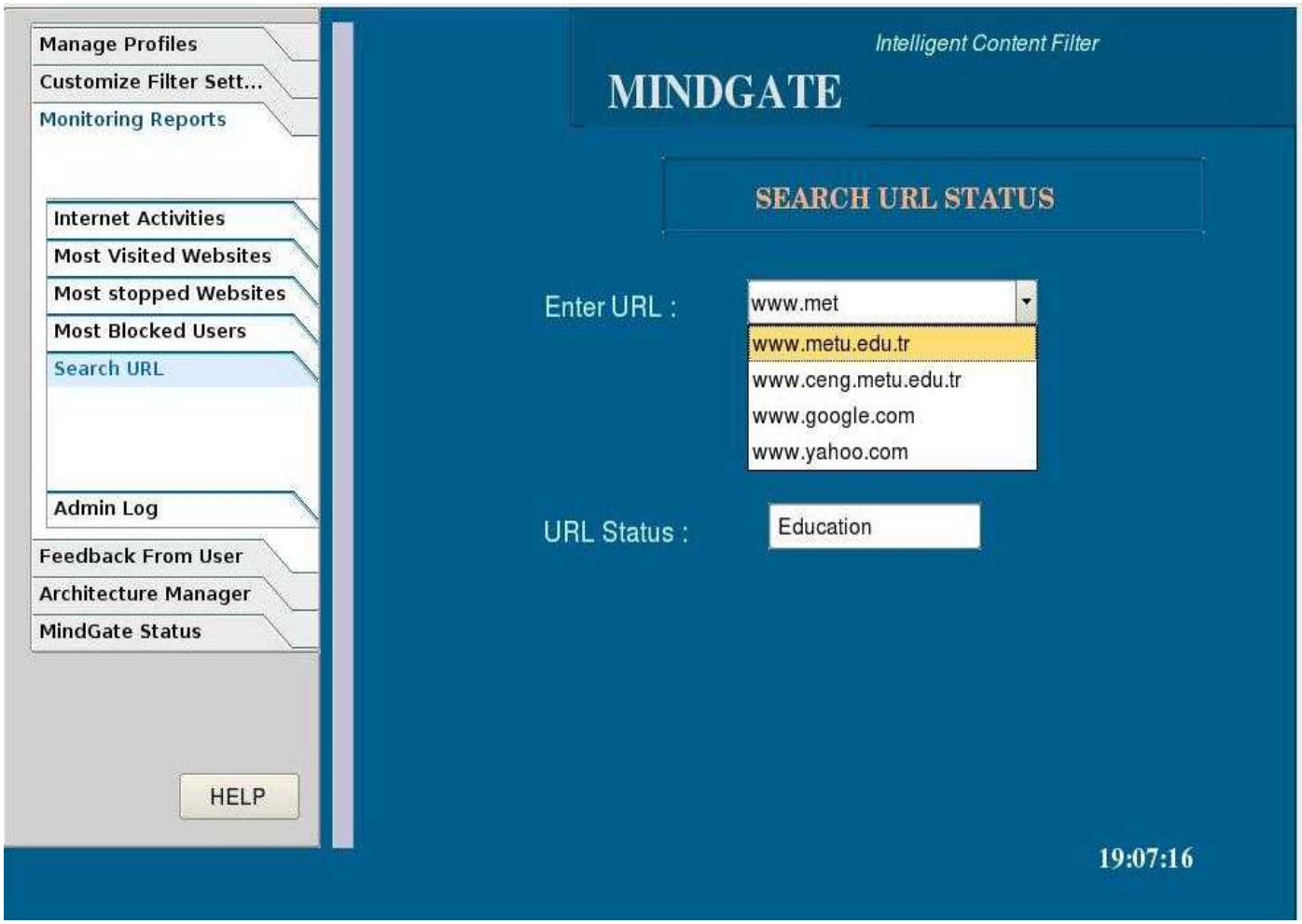


Figure27 - Querying URL Status screen

With the 'Search URL Function', admin can learn the status of the queried URL.

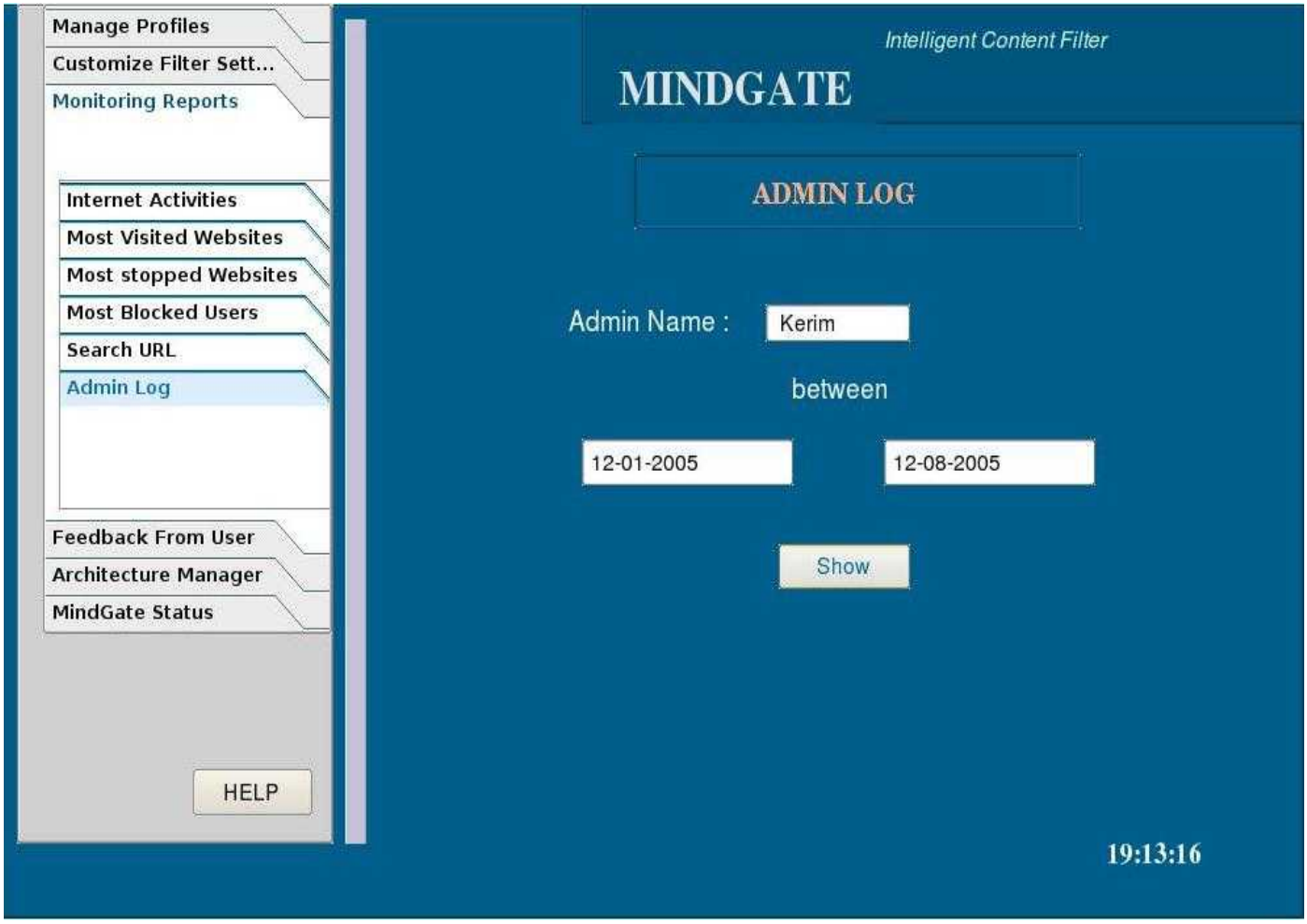


Figure28- Display Admin Log File Screen

Admin Log Function display administrator's log files between two time periods. Administrator's every act such as changing settings or removing user are saved to this log files. This log file is only readable, changing or removing administrator's log file is not allowed by MindGate. And also this log file can be examined by administrator or company manager. With this property of MindGate, manager or other company boss keep administrator under control.

Of course this is a bad property according for administrators of the system and can affect the popularity of MindGate and possibly for this reason may be many administrator will not want to use this product but there should exists such a mechanism in such a tool.

7.3.5. Feedback From User

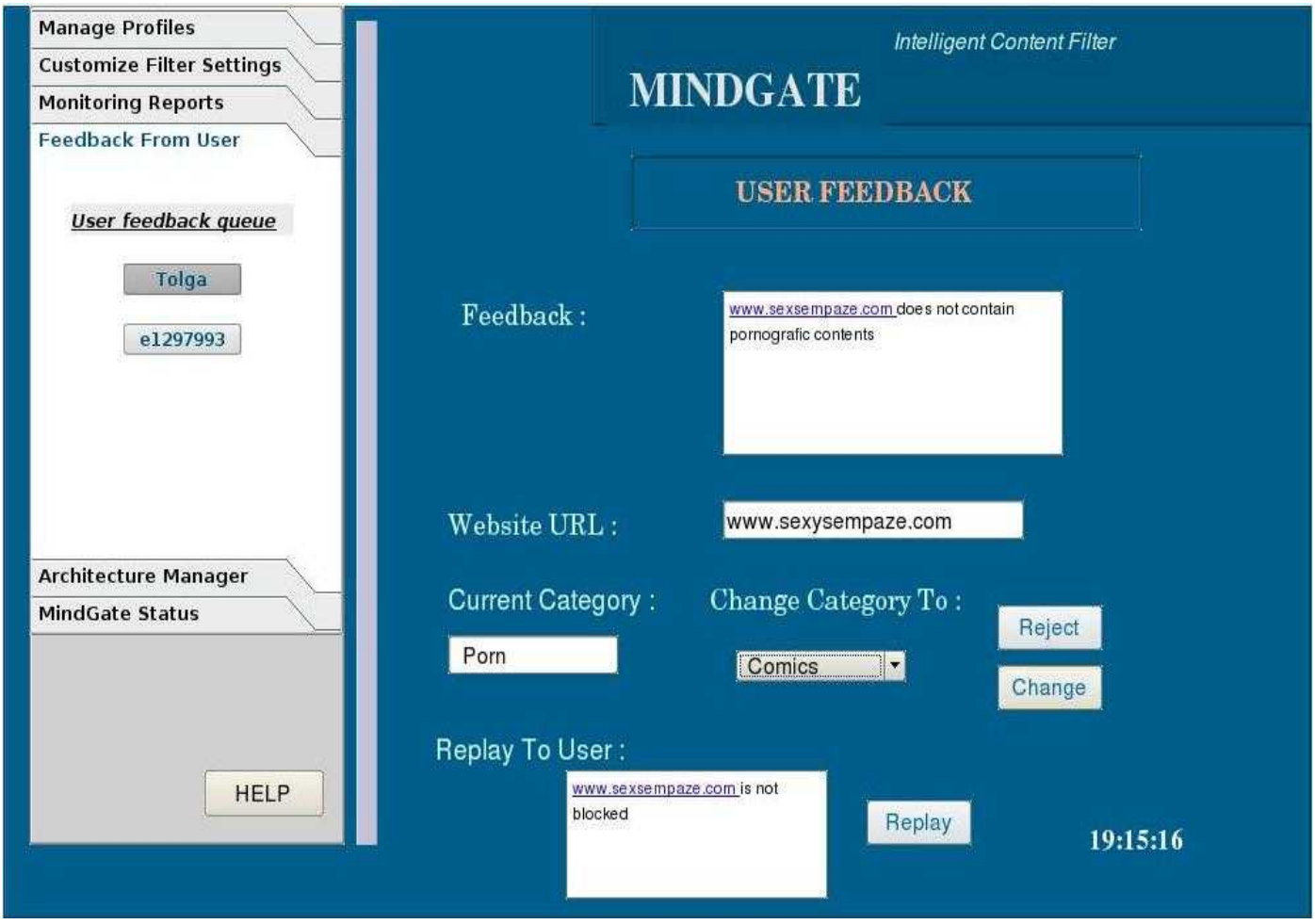


Figure29- Feedback from User to Administrator Screen

Even so MindGate is developed content filter; it also can do mistakes. To minimize these mistakes we construct a feedback form. If MindGate categorize a website under a wrong category and this site blocked, users can send a feedback form to administrator. In this form user can tell Administrator that website does not contain unwanted content with including his/her name and websites URL in this form.

Figure29 shows feedback form that received by administrator. There is also a queue under the 'Feedback From User' tab and shows how many user sent feedback. When administrator clicks the user name his/her feedback form is displayed in screen. Websites URL and its current category are shown in the screen. Administrator clicks the URL and if websites is categorized improperly, administrator changes its category by hand. And also sends a replay message to user whether or not he/she reject or approve user's feedback.

7.3.6. Architecture Manager

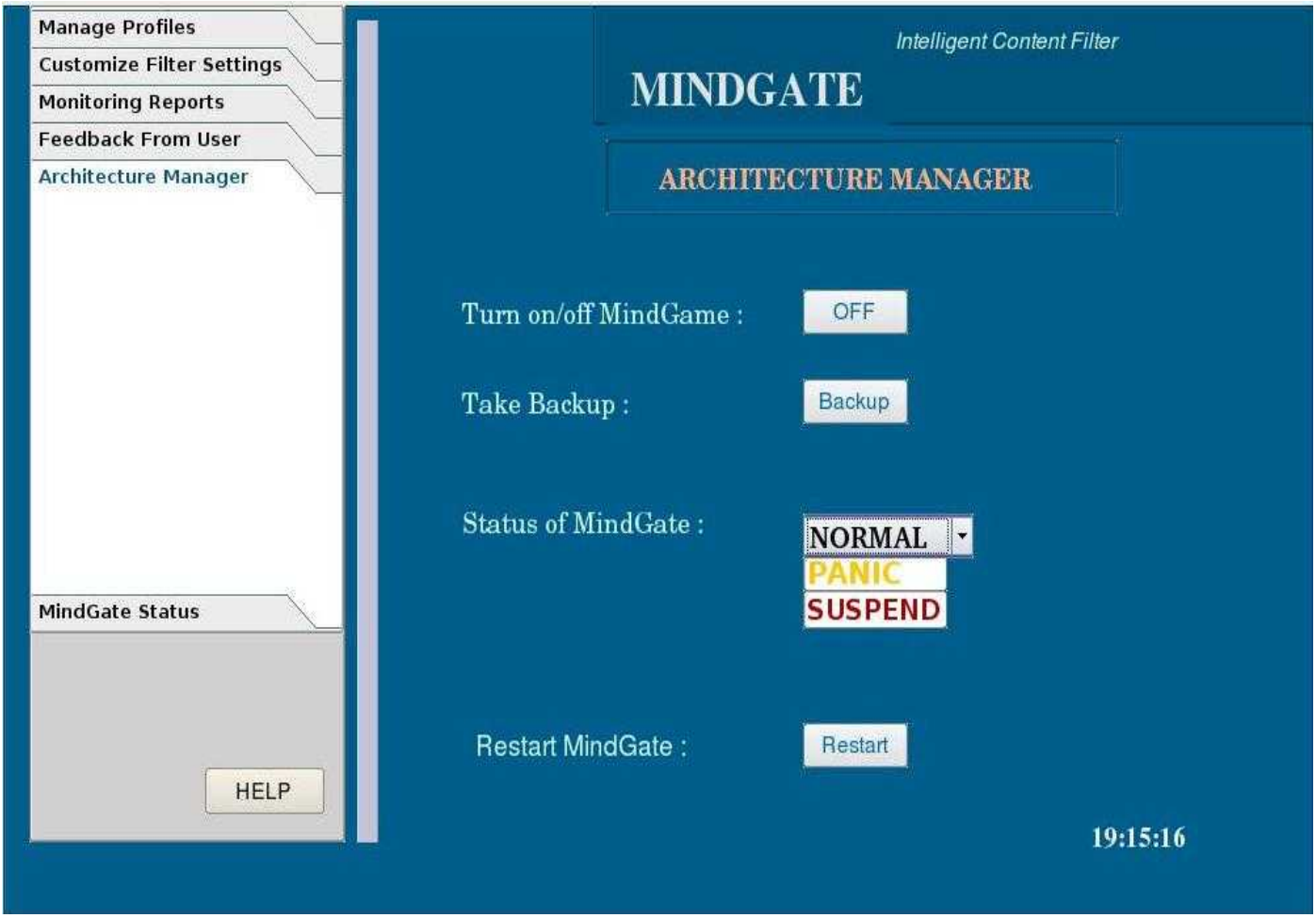


Figure30- Architecture Manager Screen

In the Architecture Manager screen, administrator manages MindGate and can turn on/off it. Also this screen allows administrator to take backup of system resources which located in RAM. Even administrator does not take backup of the system, MindGate periodically takes backup of sources. Software allows administrator to change system status. These statuses are NORMAL mode, PANIC mode or SUSPEND mode. In Normal mode, MindGate runs regularly and it filters user requests according to defined policies. In Panic mode MindGate stop all connections and does not allow any request to handle. In Suspend mode, software does not filter any request and bypass all requests coming from users.

8. SOFTWARE METHODOLOGY

8.1. SOFTWARE DESIGN CRITERIA

MindGate is designed in a flexible fashion. The whole structure and all classes will be designed to be modular with the aim of adding new components on demand. To satisfy this goal, several modifications applied on classes. At the first step, functions and input-output data on the general perspective were defined. Related Data Flow Diagrams (DFD) sketched and several modifications applied in order to increase the accuracy of the design. Main goal is to design the project in every detail without significant design errors.

In the DFD drawing process, unnecessary data flows eliminated to increase the local simplicity and modularity of the program. The Data Dictionary formed parallel to the DFD drawing process. In the Data Dictionary, the names of the data the host and destination processes and the explanation of the data are shown. In the explanation project team has tried to clarify each data by showing the format and kind of the data.

In the design process, MindUs Team searched other useful libraries and tools in order to reach excellence in the implementation and design activities. Aim is to embed extra tools into the project and concentrate on detailed issues rather than general applications which can be handled by libraries.

In the design report, as mentioned before, aim is to complete design process which can be ready to implement. So project team will continue working on the structure and other tools to develop the design.

8.2. SOFTWARE DESIGN PROCESS

In the software design process, project team have followed the democratic decentralized approach and assigned tasks to each member of the team. However, project team has worked together on the general design of the whole architecture initially. Purpose was to understand the idea behind the design and capable of every general concept of aimed product. Getting feedback from the members of the team, the structure changed several times.

In the next step, every member of the team took the responsibility of one module and formed initial drafts on these structures. Making various meetings and defining several formats on concepts, which enabled developing the system.

In the last step, all the work combined, misunderstandings and errors eliminated in the drafts. After completing the whole design, the design report was completed and project team tasks finished in this phase.

8.3. TECHNOLOGY CRITERIA

MindGate system is not a desktop publishing project, or a general purpose tool. MindUs plans to deliver the system as a complete system with hardware, operating system, and server application itself.

MindGate is a large scaled project, with various features and efficiency constraints described before and will have various external interfaces. As a four people team MindUs have to select various APIs and tools for implementation and use reusable code which is available. For specifications of the external interfaces see Detailed Description of External Components in MindGate Architecture section.

In the implementation of the project, JAVA programming language will be used. JAVA is a well known and highly portable programming language. However in MindGate project portability is not a constraint. The reason to choose JAVA for this project it is extensive support for network technology, with various APIs. All of the external interfaces planned to be used in project are implemented in JAVA for JAVA applications. Object oriented approach in this language also a plus for this language. Efficiency might seem to be a handicap, but MindGate will run on a powerful server machine, which is specially configured for this purpose so efficiency probably will not be problem.

Operating System where MindGate runs on will be a Linux server distribution, probably Debian since team members have many experiences with this distribution and it is used for similar purposes for years all around the world. License of the Debian system also allows free distribution of the software. If the MindGate goes in to market MindUs plans to distribute product on these system but only demand payment for MindGate not for operating system and external interfaces since this will be a license violation.

For an operable system, JAVA virtual machine will be installed and properly configured with custom settings for the MindGate system and its interfaces.

8.4. SOFTWARE MAINTENANCE

MindUs design methodology is Component Assembly Model. MindGate is designed to be modular for easier development and improvement. User feedback will be important after release of the core system for further evolution of the system. Since the system is developed in object oriented fashion making changes in the product will be relatively easier to other methods. Software development documentation policy was established during analysis phase to enable team to make revisions and changes at any time during development.

MindGate system must be supervised by a system administrator and the aimed customers for this product surely have large networks and administrators for these networks. MindUs plan to extensively interact with these administrators since they are the real users

of the system, to reach the development excellence. With the released system there will be documentation of the system specially prepared for the administrators.

If MindUs becomes a corporation project team plans to establish a support center for the product. In this center call support, email support and updates of the product may be found. In addition, on purpose build categories such as textile category for a textile company will be prepared and delivered. Knowledge base growing structure of the MindGate system described before. MindUs plan to collect index information of the user organizations, to merge indexes and build a large categorization index of the frequently accessed web pages. This will minimize the dynamic analyzing, auto categorization errors and grow the knowledge base of the product exponentially. Moreover this index will be corrected by user feedbacks and administrator supervision as time passes. As a result users will contribute the improvement of the system by just using it. This model will be applied by Google Inc. successfully and became a model for MindUs.

9. CLASS DIAGRAMS

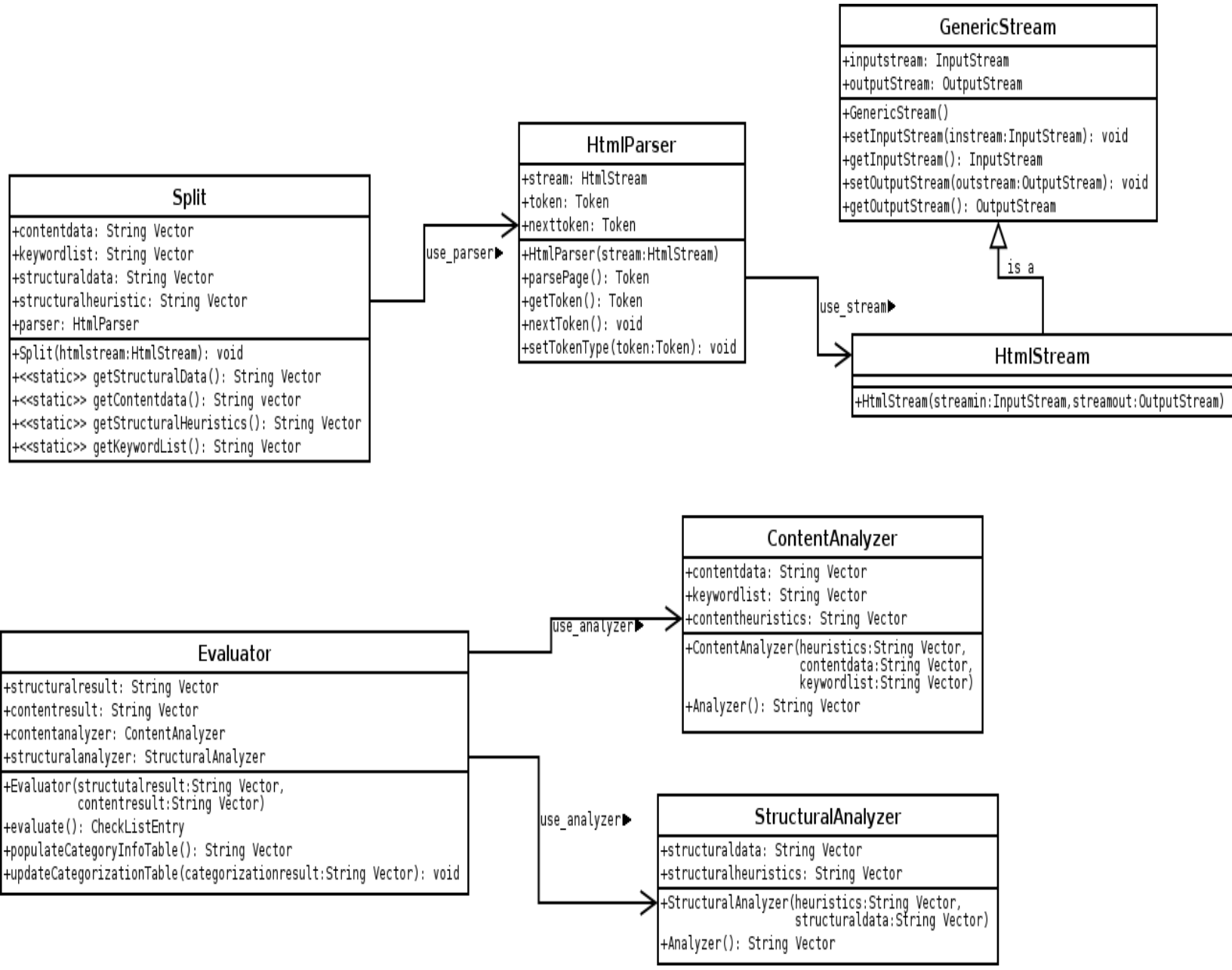


Figure31 – Class Diagram of Categorizer Unit

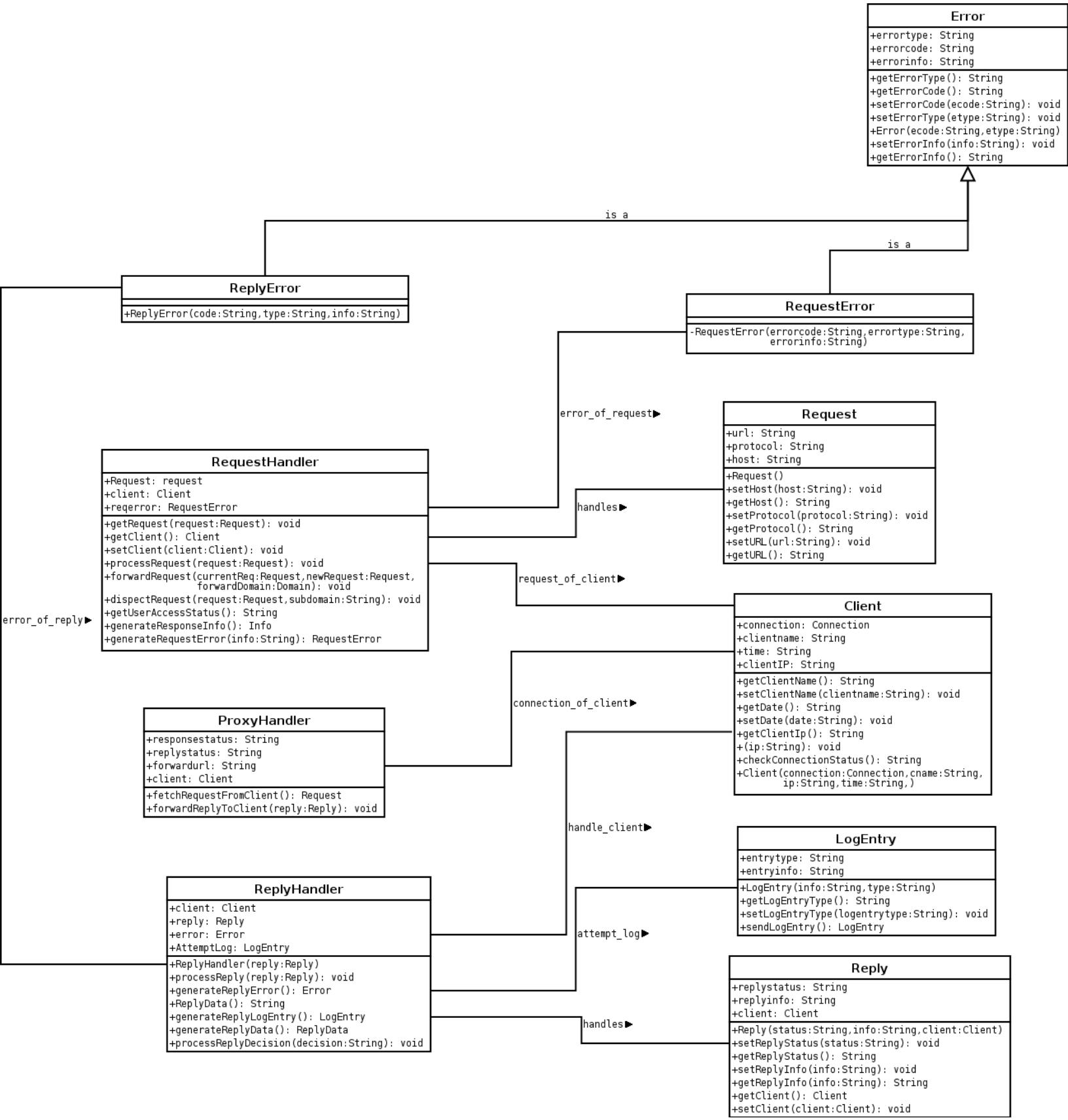


Figure32 – Class Diagram of Connection Unit

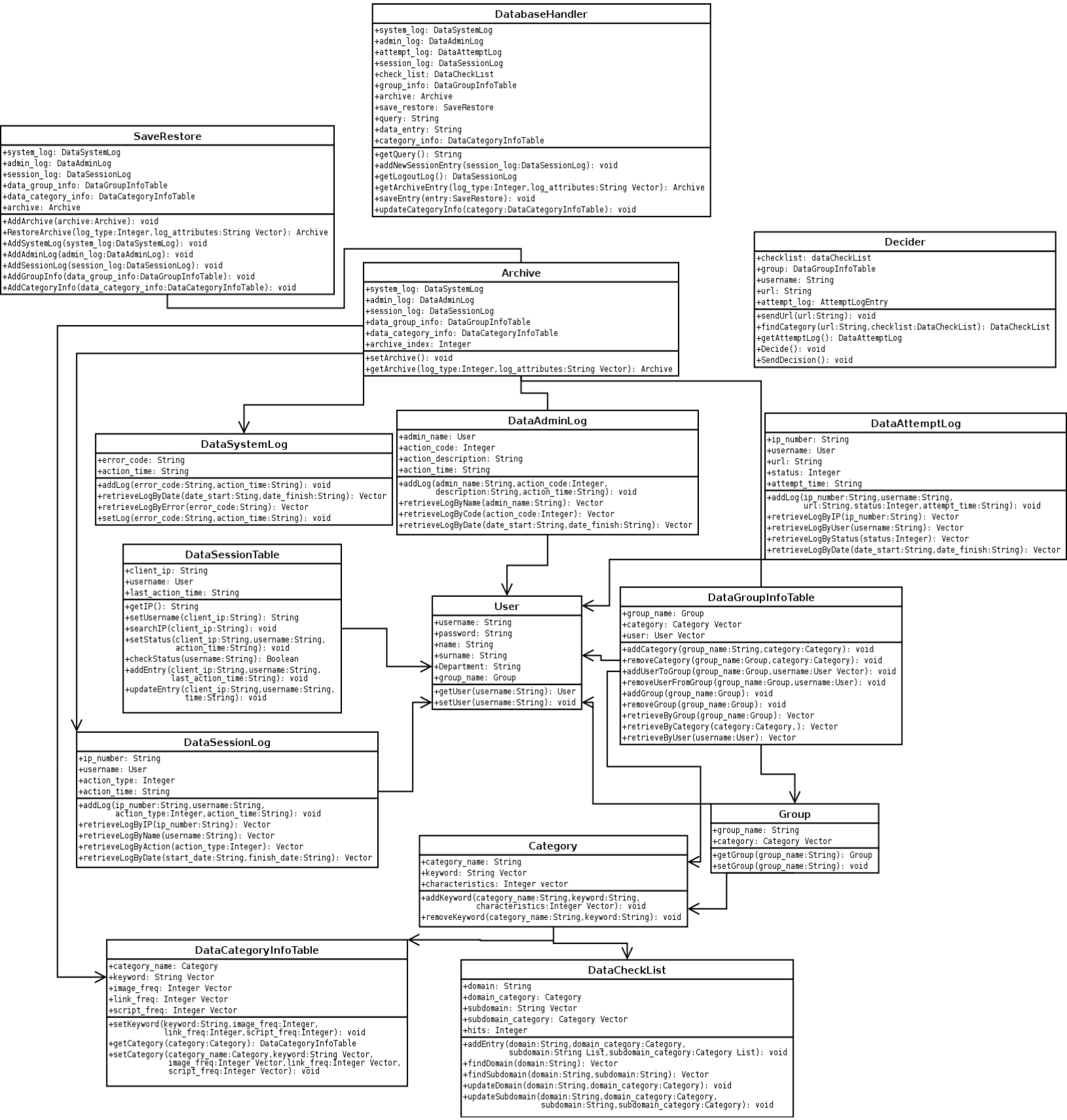


Figure33 – Class Diagram of Database Control Unit

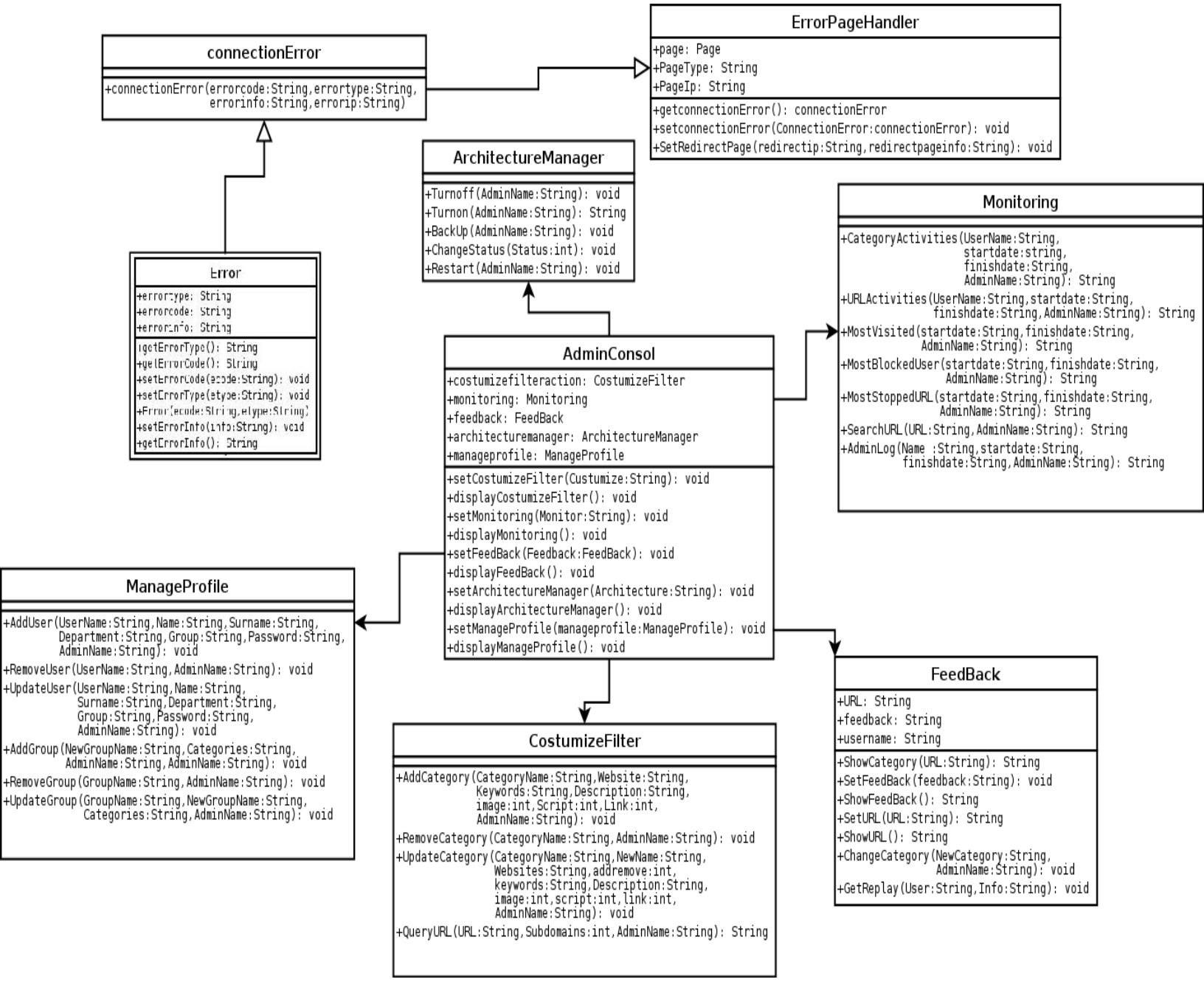
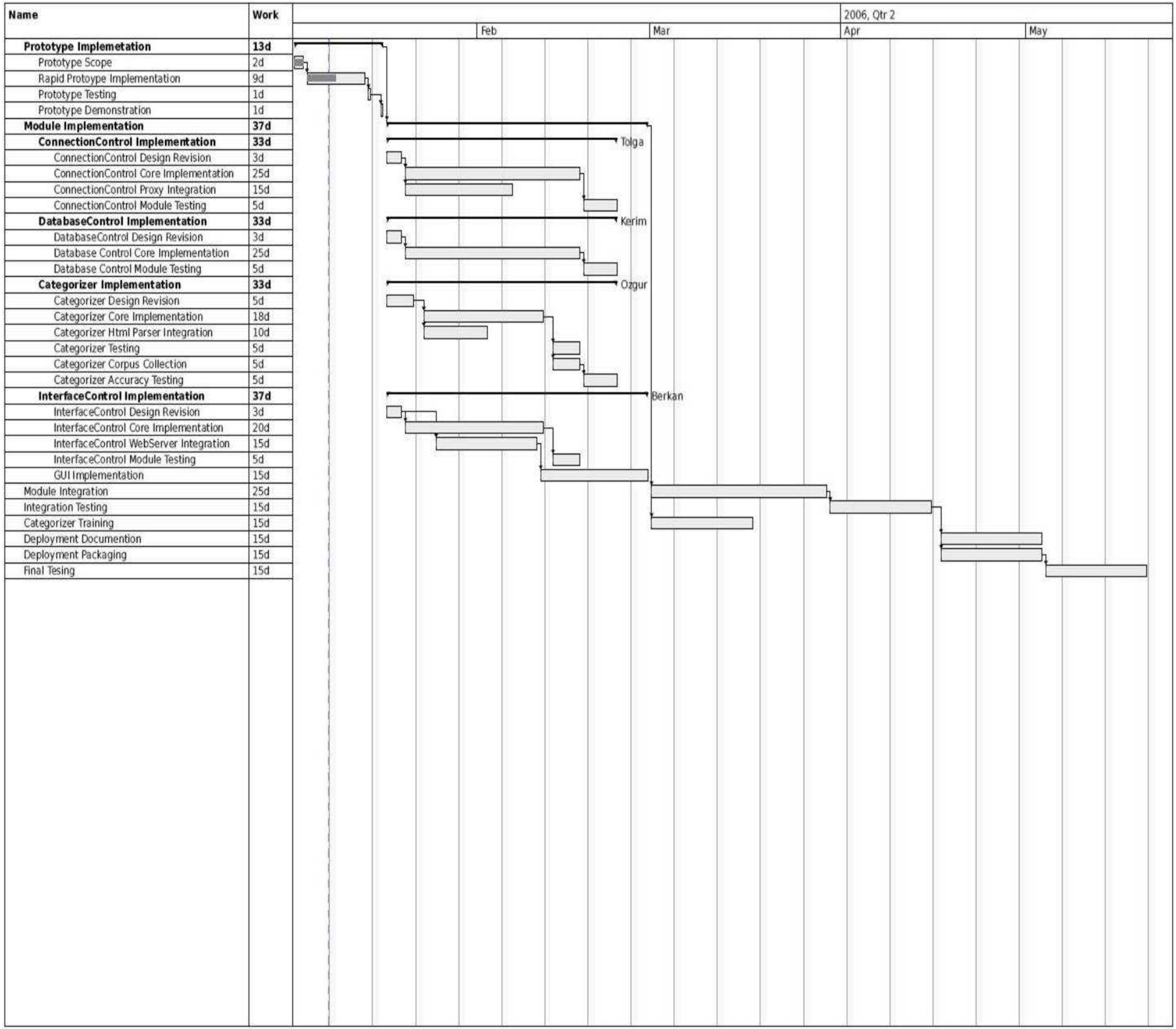


Figure34 – Class Diagram of Interface Control Unit

10. SCHEDULE OF THE PROJECT

WBS	Name	Start	Finish	Work	Duration	Slack	Cost	Assigned to
1	**Prototype Implemetation**	**Jan 2**	**Jan 16**	**13d**	**13d**	**0**		
1.1	Prototype Scope	Jan 2	Jan 3	2d	2d	0		
1.2	Rapid Protoype Implementation	Jan 4	Jan 13	9d	9d	0		
1.3	Prototype Testing	Jan 14	Jan 14	1d	1d	0		
1.4	Prototype Demonstration	Jan 16	Jan 16	1d	1d	0		
2	**Module Implementation**	**Jan 17**	**Feb 28**	**37d**	**37d**	**0**		
2.1	**ConnectionControl Implementation**	**Jan 17**	**Feb 23**	**33d**	**33d**	**4d**	**0**	Tolga
2.1.1	ConnectionControl Design Revision	Jan 17	Jan 19	3d	3d	4d	0	
2.1.2	ConnectionControl Core Implementation	Jan 20	Feb 17	25d	25d	4d	0	
2.1.3	ConnectionControl Proxy Integration	Jan 20	Feb 6	15d	15d	19d	0	
2.1.4	ConnectionControl Module Testing	Feb 18	Feb 23	5d	5d	4d	0	
2.2	**DatabaseControl Implementation**	**Jan 17**	**Feb 23**	**33d**	**33d**	**4d**	**0**	Kerim
2.2.1	DatabaseControl Design Revision	Jan 17	Jan 19	3d	3d	4d	0	
2.2.2	Database Control Core Implementation	Jan 20	Feb 17	25d	25d	4d	0	
2.2.3	Database Control Module Testing	Feb 18	Feb 23	5d	5d	4d	0	
2.3	**Categorizer Implementation**	**Jan 17**	**Feb 23**	**33d**	**33d**	**4d**	**0**	Ozgur
2.3.1	Categorizer Design Revision	Jan 17	Jan 21	5d	5d	4d	0	
2.3.2	Categorizer Core Implementation	Jan 23	Feb 11	18d	18d	4d	0	
2.3.3	Categorizer Html Parser Integration	Jan 23	Feb 2	10d	10d	22d	0	
2.3.4	Categorizer Testing	Feb 13	Feb 17	5d	5d	9d	0	
2.3.5	Categorizer Corpus Collection	Feb 13	Feb 17	5d	5d	4d	0	
2.3.6	Categorizer Accuracy Testing	Feb 18	Feb 23	5d	5d	4d	0	
2.4	**InterfaceControl Implementation**	**Jan 17**	**Feb 28**	**37d**	**37d**	**0**	Berkan	
2.4.1	InterfaceControl Design Revision	Jan 17	Jan 19	3d	3d	0		
2.4.2	InterfaceControl Core Implementation	Jan 20	Feb 11	20d	20d	9d	0	
2.4.3	InterfaceControl WebServer Integration	Jan 25	Feb 10	15d	15d	0		
2.4.4	InterfaceControl Module Testing	Feb 13	Feb 17	5d	5d	9d	0	
2.4.5	GUI Implementation	Feb 11	Feb 28	15d	15d	0		
3	Module Integration	Mar 1	Mar 29	25d	25d	0		
4	Integration Testing	Mar 30	Apr 15	15d	15d	0		
5	Categorizer Training	Mar 1	Mar 17	15d	15d	55d	0	
6	Deployment Documention	Apr 17	May 3	15d	15d	15d	0	
7	Deployment Packaging	Apr 17	May 3	15d	15d	0		
8	Final Tesing	May 4	May 20	15d	15d	0		

Figure35 – Time Schedule of the Project


Name
Work
2006, Qtr 2
Feb
Mar
Apr
May
Prototype Implemetation 13d
Prototype Scope 2d
Rapid Protoype Implementation 9d
Prototype Testing 1d
Prototype Demonstration 1d
Module Implementation 37d
ConnectionControl Implementation 33d
ConnectionControl Design Revision 3d
ConnectionControl Core Implementation 25d
ConnectionControl Proxy Integration 15d
ConnectionControl Module Testing 5d
DatabaseControl Implementation 33d
DatabaseControl Design Revision 3d
Database Control Core Implementation 25d
Database Control Module Testing 5d
Categorizer Implementation 33d
Categorizer Design Revision 5d
Categorizer Core Implementation 18d
Categorizer Html Parser Integration 10d
Categorizer Testing 5d
Categorizer Corpus Collection 5d
Categorizer Accuracy Testing 5d
InterfaceControl Implementation 37d
InterfaceControl Design Revision 3d
InterfaceControl Core Implementation 20d
InterfaceControl WebServer Integration 15d
InterfaceControl Module Testing 5d
GUI Implementation 15d
Module Integration 25d
Integration Testing 15d
Categorizer Training 15d
Deployment Documention 15d
Deployment Packaging 15d
Final Tesing 15d
Tolga
Kerim
Ozgur
Berkan