siteMETA Manual

Table of Contents

1. Overview

1.1 Application area

siteMETA software is an easy-to-use and powerful search tool for full-text search within a web-site.

siteMETA can easily be configured to run on a certain server. User dialog with the system and administrating of the system is done with the use of web interface. **siteMETA** combines index and search modules. The index module automatically tracks changes in the existing documents and creation of new documents and constantly update the index. As a result the search presents the newest version of the site.

1.2 Specifications

siteMETA is an intelligent solution enabling comprehensive multi-language text processing and the only one unique system with the full-scale English language support. The service is being created on the basis of the the data search technologies developed and refined at the ground of the popular Ukrainian web search engine <META>, whereas the query processing time does not essentially depends on the index size and never exceeds a split second.

The detailed characteristics on every component of **siteMETA** are given below.

1.2.1 Index module

- Full text index with a high reindexing speed
- Automated file content analysis
- Automatic URL extraction for further indexing
- HTTP protocol for document access
- Local network documents access
- Option of setting up time intervals for balancing the server loading
- Option of adding front-end features for multiple file format supported
- Automatic language and encoding recognition
- Recognition of formatting elements in HTML documents
- Detection of the meta-tag **"robots"** of an HTML-file
- Duplicated documents detection
- Automatic tracking of changes in existent documents and creation of new ones (if they are linked), thus keeping the index up-to-date all the time long.

1.2.2 Search module

- Qualitative search result rankings according to the relevance algorithm
- Search ability with date limits and presentation of the result depending on the date.
- Complex query language (Boolean, refinement, field restriction, date operators)
- Comprehensive morphological support for the English, Russian and Ukrainian languages
- User-defined tunable results page layout
- Multiple pages results presentation.
- Quotation of relevant text fragments with underlined keywords
- Rich options to set up the query syntax
- Presentation of the most relevant documents fragments.
- Option to add a stop word list (obscene vocabulary)

1.2.3 Query language

- Exact phrase search option
- Context limit search option
- Boolean operators: AND, OR, AND NOT, sequential operator - "()" option
- Truncation search operator "*" option
- Document title search option

1.2.4 Search service

- Option to index while searching
- Custom schedule mode switch option
- Compatible with all web servers supporting CGI scripts

1.3 System requirements

siteMETA is developed to run under
Windows NT 4.0 Service Pack 6a (i386)
Windows 2000 Professional or Server (i386)
Windows 2003 Server (i386)
Linux (i386)
FreeBSD (i386)

The installation of **siteMETA** requires 1 to 10 MB hard drive space. Additional space on the hard drive in required for storing the index.

While indexing, free space should be a half (if stored in the main index) to three (if indexed and stored in the temporary index) sizes of the indexed site files, this can take from 0.5 to 3 times of the indexed files size.

To build a 100000 text file index with the average document size of 25Kb one requires approximately 2Gb disk space and another 2Gb for index updating.
The minimum memory size is 128Mb.

1.4 The principle of work.

siteMETA system consists of three main modules: the index module, search service module, CGI-script for web interface and CGI-script for administration interface.
The index and search service are independent applications for simultaneous indexing and searching documents.
Service:

- receives administrative commands and search queries via HTTP,
- delivers (downloads) documents via HTTP or from the local network,
- indexes the delivered documents, transmits the results into the search index,
- updates the search index according to the schedule,
- provides the option to searches within the search index.

The search CGI-script exists for the web server with the index and search service connection. It receives the search query and generates the results page.
CGI-script of the administration interface manages the search server via web interface. The administration interface is used to set up the list of directories to be indexed, the indexation restrictions for documents, the reindexation schedule, etc.

2. Installation and tuning

2.1 Archive set

siteMETA distributive includes:
/siteMETA

./etc/ - folder with search service setting files,
./db/ - search index folder,
./doc/ - documentation folder,
./lib/ - system libraries (linguistic support modules, format filters),
./www/ - CGI-scripts (search and administration interface),
./bin/ - search service,
./license.txt – license agreement.

2.2 Installation and quick start

2.2.1 Installing under Windows NT/2000

Run the installer **siteMETA-w2k-en-<version number>.exe.** Having accepted the license agreement, follow the installation wizard to complete the installation process.

A new program group **"siteMETA"** will be created**.** If you are using **Internet Information Server** version 5.0 and higher, the virtual directory **siteMETA** will be created automatically, linked to the search core folder (**"./www/"**). This virtual directory will have the execution **"Scripts and Executables"**) and read (**"Read"**) rights. If you are using another web server, you should manually create a virtual directory named **siteMETA**, link it to the folder **"./www/"** and assign the necessary rights.

The search service is installed in the system under the service name **siteMETA**. To start the search service, do the following:

-start the search service using the **"Start service"** icon in the menu **Start/Programs/siteMETA**, or from the folder **Services Windows,**

-load the administration interface (by default **http://localhost/siteMETA/admin/mssAdmH.exe**), set the index parameters and click on **"Reinitialize"**.

2.2.2 Installing under Windows 2003 Server

The installation of **siteMETA** under Windows 2003 Server with IIS 6.0 a bit differs from the installation under Windows 2000 with IIS 5.0.

Hereinafter it is supposed to be used

- Windows 2003 Server
- IIS 6.0
- and siteMETA is installed in C:\Program Files\META\siteMETA-2.1.10

After the **siteMETA** installation completed, the following should be done:

- configure the web service to execute the CGI-scripts of the search service
- create a virtual directory for execution of the search service CGI-scripts.

In the section **Start/Administrative Tools/Internet Information Services (IIS) Manager** choose the **Web Services Extensions**.

Choose the link **Add a new Web service extension...** in the opened window**.**

Enter the extension name and the CGI-scripts file names in the dialog window:

Extension name:
siteMETA-CGI

Required files:
C:\Program Files\META\siteMETA-2.1.10\www\mssCgi.exe
C:\Program Files\META\siteMETA-2.1.10\www\admin\mssAdmH.exe

Set the flag **Set extension status to Allowed**.

To create the virtual directory the following should be done:

in the section **Start/Administrative Tools/Internet Information Services (IIS) Manager** choose **Web Sites/site_name**, where **site_name** is the name of the site where the virtual directory is supposed to be created.

In the **Action/New** menu choose **Virtual Directory...**

In the opened wizard set:

- Alias – siteMETA
- Path – C:\META\siteMETA-2.1.10\www\
- Access rights: **Read**, **Execute**
- close the wizard.

Then, the access rights to the directory **/siteMETA/admin/** must be set so that only the search service administrator has the rights to administer the search service.

2.2.3 Installing under FreeBSD and Linux

siteMETA distributive for these operation systems is an archive to be extracted and to inserted the directory tree into the appropriate directory (e.g. /usr/local/).

To unpack use the command:

tar -zxvf {tgz-archive-name}

Copy the files from the folder **/usr/local/siteMETA/www/** into the web server directory that is used to launch the CGI-scripts (normally it's the virtual directory cgi-bin). The search and administration URLs will then look like:

Task	*URL*
Search	http://{www server name}/cgi-bin/mssCgi
Administration	http://{www server name}/cgi-bin/admin/mssAdmH *

* - It's recommended to limit the access to this URL using the web server functions and only allow the access to the users with the **siteMETA** administration rights.

The index and search module is initialized using the config file, whose full path should be given in the environment variable *siteSearchCfg*. It's also possible to specify an alternative config file using the key -init: (see example)

The commands to launch the index and search module are:

/usr/local/siteMETA/bin/mssSvrs -pid:{pid file name} -start
or
/usr/local/siteMETA/bin/mssSvrs -pid:{pid file name} -init:/ usr/local/siteMETA/etc/siteSearch.ini -start

Stop the index and search module:
/usr/local/siteMETA/bin/mssSvrs -pid:{pid fiel name} -stop

The service can be also stopped via the administrator's interface.

The administrator's interface and the CGI search script are initialized from the config files. The config files can be moved from the virtual directory **/cgi-bin/**, while it's necessary to specify the environment variables *mssCgiCfg* for mssCgi and *mssAdmHCfg* for the administration module (mssAdmH). By default the config files are sought in the current directory.

NOTE*:* while moving the config files, all the keys with the template file paths inside the ini files should be changed, or the templates should be moved together with the ini files. The paths to the log files should also be changed to correspond with the new location of the ini files. All the paths in the ini files are relative by default and point to the external files relative to the path to the config file location.

2.2.4 Search service configuration via the administrator's interface.

The administrator's interface is used for the search service configuration management.

To start indexing you need to form the search documents collection, configure indexing setting and downloading .

For detailed description of indexing settings and downloading tuning configuration see item **2.3.**

2.2.5 Search form integration into a web page.

For Windows NT/2000:

Make sure the virtual directory **"siteMETA"** is linked to the files from the distributive directory **"www"**, and the directory has all the required rights. The search URL for your site will be as follows:

http://{your_site_name}/siteMETA/mssCgi.exe

The administrator's interface URL is:

http://{your_site_name}/siteMETA/admin/mssAdmH.exe

Add the following code to the page your want to integrate the search form to:

```
<FORM action="/siteMETA/mssCgi.exe" method="get">
 <INPUT maxLength="80" name="q" size="35">
 <INPUT type="submit" value="Search">
</FORM>
```

For Unix-systems:

The path to the CGI search and administration applications is defined according to item **2.2.3 Installing under FreeBSD and Linux**.

2.2.6 Start

After all the above preparing operation took place the siteMETA search system will be available for work. For the search service to start do the following:

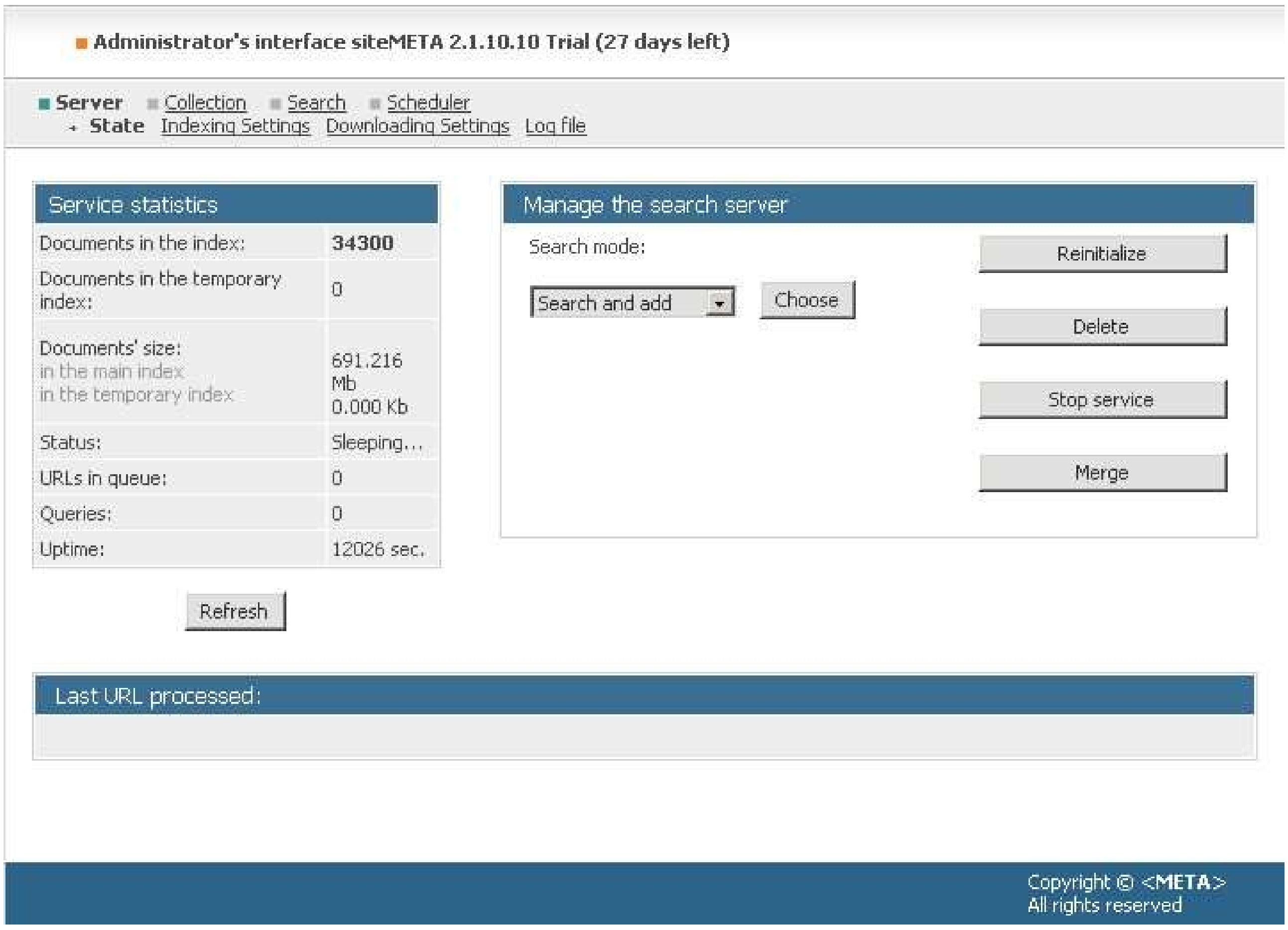
- start the search service using ***"Start service"*** icon in the menu **Start/Programs/siteMETA** or from **Services Windows** folder.
- Run the administrator's interface (by default it is: **http://localhost/siteMETA/admin/mssAdmH.exe)** choose the index parameters and press the **"update"** button.

After the above actions took place the siteMETA search system will start indexing your site content. It is not necessary to wait till the end of the indexing process, it is already possible to use the search service while the system is still indexing your site content. If you receive an error for the search results, please, make sure that you have correctly configured the CGI scripts and if your IIS server (or another server installed) has the rights to access them. More details on this questions you will be able to find in the **"./db/"** file. In case you are still experiencing some difficulties use the appropriate part of this manual or contact our technical support team at: support@meta.ua

2.3 Administering search service with use of administrator's interface.

The administrator's interface is a web-based administrative interface enabling:

- to set the rules to form the collection of the documents to be indexed
- to set the time parameters of the collection reindexing cycle,
- to change the parameters and modes of indexing, downloading and storing,
- to schedule the periodic events of the indexing process,
- to monitor the search service status.


Administrator's interface siteMETA 2.1.10.10 Trial (27 days left)

Server | Collection | Search | Scheduler

State | Indexing Settings | Downloading Settings | Log file

Service statistics

Documents in the index:	**34300**
Documents in the temporary index:	0
Documents' size: in the main index in the temporary index	691.216 Mb 0.000 Kb
Status:	Sleeping...
URLs in queue:	0
Queries:	0
Uptime:	12026 sec.

Refresh

Manage the search server

Search mode: Search and add | Choose

Reinitialize | Delete | Stop service | Merge

Last URL processed:

Copyright © <META> All rights reserved

Screenshot: Administrator's interface start page

2.3.1 Documents collection forming

NOTE: it is meant that the search service was successfully installed and launched.

Forming of the elements collection that will be included into the index is 2 staged

establishing the URLs to start from,

-establishing the permission and stop rules for URLs.

The documents to be indexed are collected by tracking the valid URLs, according to the permission rules. New URLs are found while scanning the downloaded documents and detecting links to other documents from the collection.

To begin indexing, the URLs should be defined to start from. These URLs are the starting point for updating the search index.

Every update cycle begins from checking the start URLs, detecting links to other documents and verifying their belonging/accessory to the collection.

URL belongs to the current collection if:

-it points to a document on one of the sites indicated as starting URL, i.e. it passes the primary masks,

-there is no stop rule that applies to this URL;

-this URL passes at least one of the permitted rules.

The service tries to download and process all URLs belonged to the collection.

The allowed characters in the rules are “*” and “?”, while “*” substitutes for any substring (also an empty one), and “?” is any single character.

NOTE: the characters “*” and “?” themselves cannot be substituted.

Example:

-the stop rule for indexing all documents from the directory /stat/ of the site *somesite*, should be written as ***http://somesite/stat/****

-the permit rule for indexing all documents with the extension .htm is ****.htm***.

NOTE*:* the case of letters DOES matter for the URL collection rules.

In case if the distributive supports the archived files, as RAR and ZIP, processing the permit rules for this should have been applied for the documents found in the archives. The syntax of the rule is following:

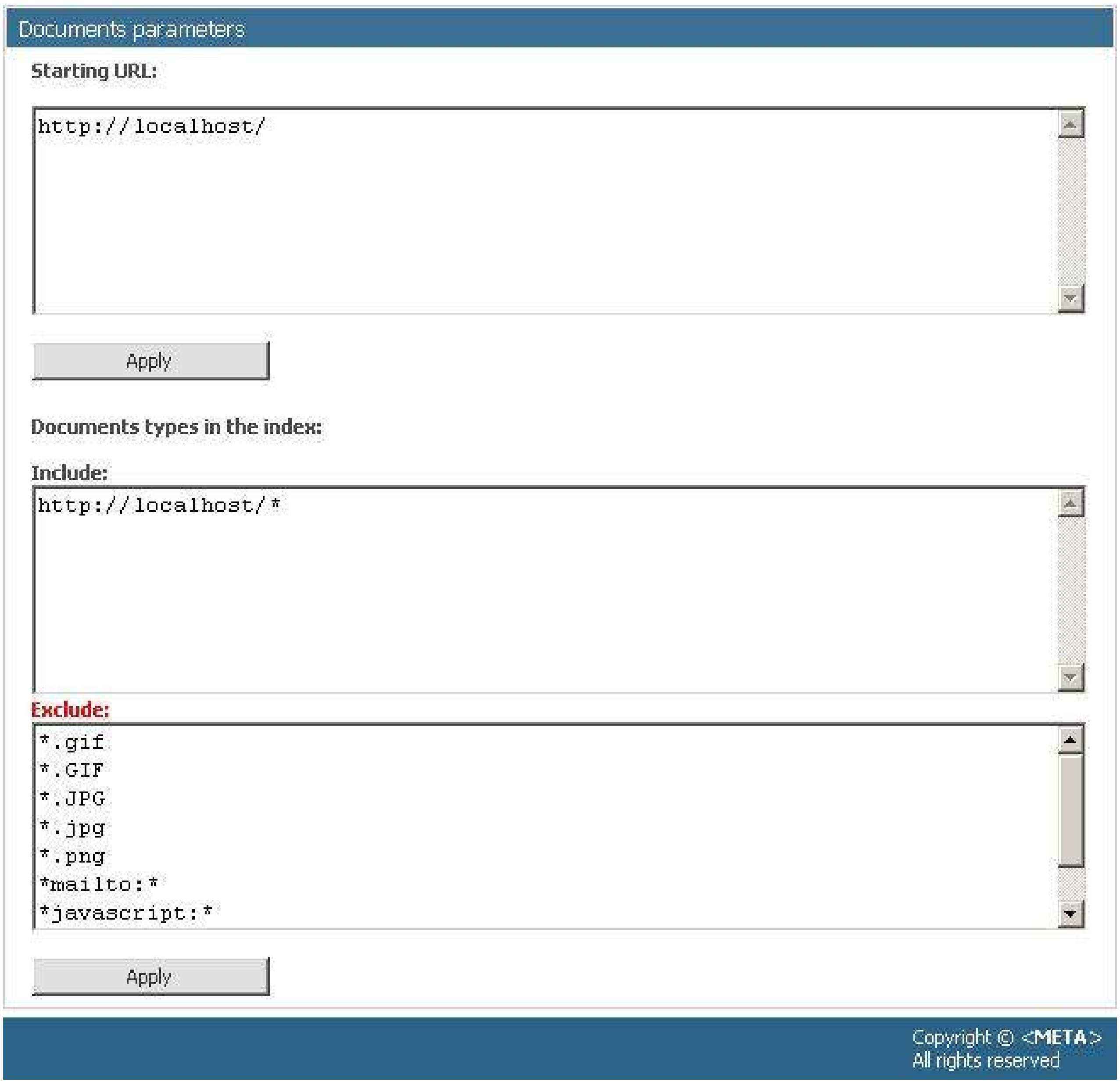
***container_path=*.<extention>**

For instance, to permit indexing files .htm, .html, .txt the following rules for the archives must be applied:

***container_path=*.htm**
***container_path=*.html**
***container_path=*.txt**

NOTE*:* the rules of belonging URLs from the file archive to the collection ARE NOT dependent on the case of the letters.

Screenshot: forming the documents collection


Documents parameters

Starting URL:

```
http://localhost/
```

Apply

Documents types in the index:

Include:

```
http://localhost/*
```

Exclude:

```
*.gif
*.GIF
*.JPG
*.jpg
*.png
*mailto:*
*javascript:*
```

Apply

Copyright © <META>
All rights reserved

Indicate the start URLs of the collection in the field "*Start URL*" on the collection management page **Collection/Settings**, and push "*Apply*" button.

The permit and stop rules are specified at the same page in two different fields (include/exclude) "*Documents types in the index*" and confirmed by clicking "*Apply*".

2.3.2 Forming documents collection in Windows local network

The documents on the network mapped drives are indexed using the URL format: ***file://computername/foldername/*** , where ***computername*** is the network computer name, ***foldername*** is the shared folder name.

All documents with the **guest** access rights will be indexed**.**

Add ***file://*/*** to the permitting masks to enable indexing of sub-folders.

NOTE*:* to get the files in the local area network indexed you need to configure the search service the way that it will be started as user that have rights to access the network and the system files at the same time. Otherwise the service starts as LocalSystem, which does not have access right to the network.

To change this press **Start>Control panel>Administrative tools>Services,** choose properties of siteMETA, at bookmark **Log file** choose point **Lon on as>This account** and choose the user that has all the necessary access rights (access to the network resources, access to the system files, access to the search index files, rights Lon on as a service).

Restart the search service.

If the service did not start and the following error occured:

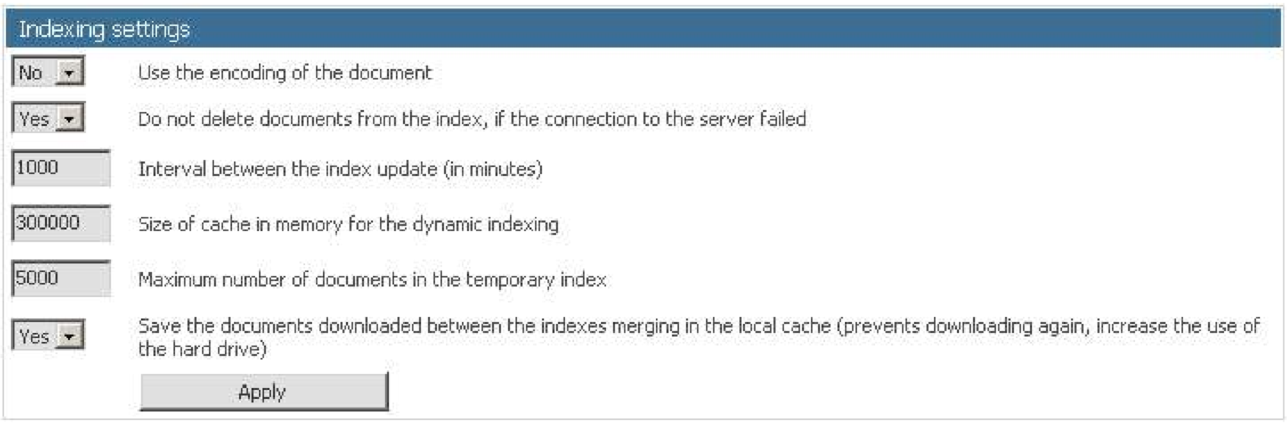
Could not start metaSE-2.1.10 service on Local Computer.
Error 1069: The service did not start due to a logon failure
This means that the chosen user does not have rights to Log on a a service. To add this right to the chosen user use: **Start > Control Panel > Administrative Tools > Local Security Policy** choose **Local Policies > User Rights Assignment**, choose Log on as a service and add the chosen user to the users list.

2.3.3 Index and download parameters set up

The index and download parameters can be specified in the sections **Server/Indexing settings** and **Server/Downloading settings**.

Indexing settings

Indexing settings



No	Use the encoding of the document
Yes	Do not delete documents from the index, if the connection to the server failed
1000	Interval between the index update (in minutes)
300000	Size of cache in memory for the dynamic indexing
5000	Maximum number of documents in the temporary index
Yes	Save the documents downloaded between the indexes merging in the local cache (prevents downloading again, increase the use of the hard drive)

Screenshot: Indexing settings

Use the encoding of the document
The setting lets the service preserve the original encoding of the document to be indexed. If the setting is set to ***NO*** or the encoding is undefined, the service tries to determine it using given statistical algorithms.

Do not deleted documents from the index, if connection to web-server failed. The setting determines the behavior of the service if the transport module fails to connect to the host server. If ***YES*** is chosen, the documents will NOT be deleted from the index. Otherwise such documents are assumed as non-existent and removed from the index.

Interval between index updates (in minutes)
The field determines the interval in minutes between the termination of one update cycle and the start of the next one.

Size of cache in memory for the dynamic indexing
Specifies the size (the total number of entries) of cache containing the coordinates of every indexed document. Using cache, you can essentially decrease the disk transfer and accelerate the inclusion of documents into the temporary index. The default size is 200000, the recommended cache size for large sites is 1000000. The maximum possible value is 2000000 entries. A one-million-entry cache

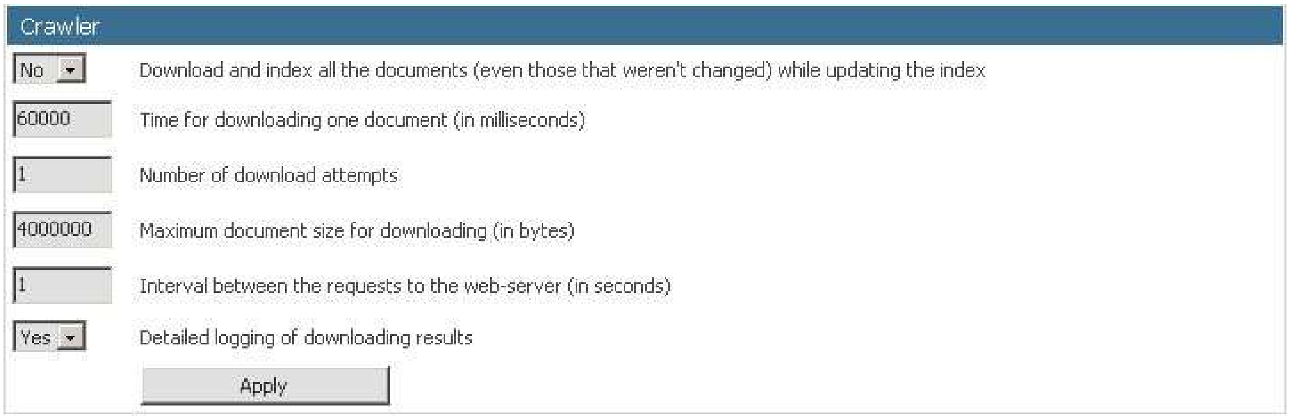
requires appr. 40 Mb of random access memory.

Maximum number of documents in the temporary index
Limits the size (in documents) of the temporary index. When the limit is reached, the temporary index is merged with the main search index.

Save the documents downloaded between the indexes merging in the local cache

(prevents downloading again, increase the use of the hard drive)
If *Yes*, all documents that were downloaded and indexed during the merging, are stored in the local cache. It saves network traffic while eliminating recurrent downloads if the temporary index was lost due to software or hardware failure. The documents can be thus reindexed from the local cache. The local cache size is equal to the size of the downloaded documents.

2.3.3.2 Downloading settings


Crawler

No	Download and index all the documents (even those that weren't changed) while updating the index
60000	Time for downloading one document (in milliseconds)
1	Number of download attempts
4000000	Maximum document size for downloading (in bytes)
1	Interval between the requests to the web-server (in seconds)
Yes	Detailed logging of downloading results

Screenshot: downloading settings

Detailed logging of download result

If ***YES*** is set, then for each failed to download file HTTP return code will be written.

Download and index ALL the documents, were not changed, while updating the index. If ***YES is set***, all documents in the index will be downloaded and reindexed during the update procedure. Otherwise only recently modified documents will be updated.

Time for downloading one document (in milli seconds).
The time limit in milliseconds to download one document. Download is terminated when time is out. Minimum timeout is 10000, maximum - 180000 milliseconds.

Number of download attempts.
The number of attempts to connect to the server after the first attempt is failed. By default it is one attempt, without repeating connection requests. Maximum value is 20.

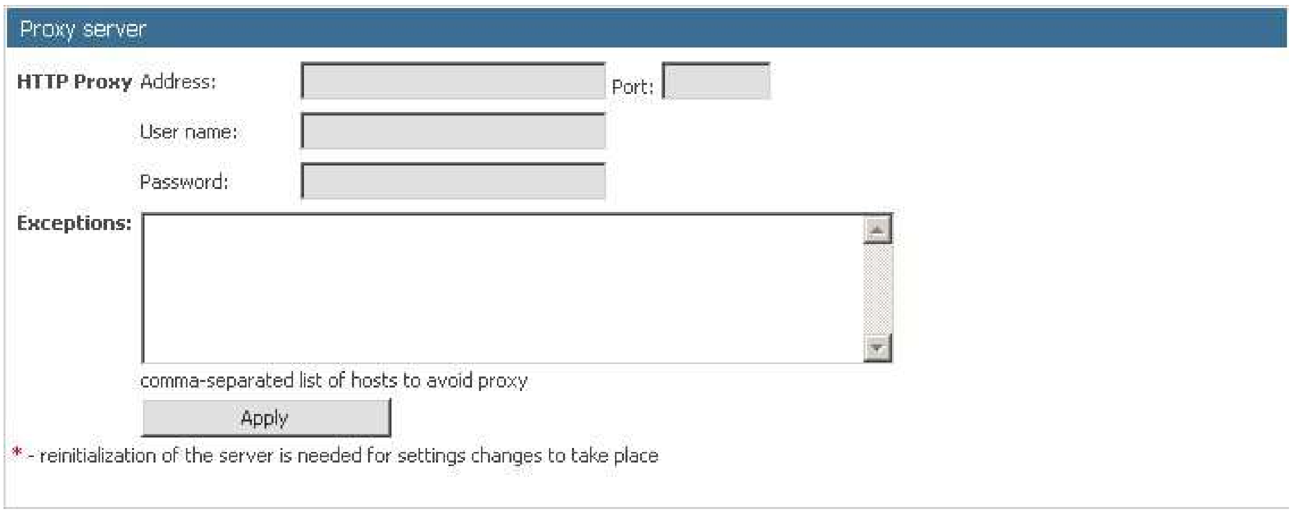
Maximum document size for downloading.
Limits the document size to be downloaded (in bytes). If the document exceeds the limit, it will be downloaded partly.

Interval between queries to the web-server (in seconds)
Sets the minimum pause between queries to one web-server in seconds.

When the parameters were set, click *"Apply"*.

Proxy server

Screenshot: Proxy server settings



Proxy server

HTTP Proxy Address:	Port:
User name:	
Password:	
Exceptions:	comma-separated list of hosts to avoid proxy

Apply

* - reinitialization of the server is needed for settings changes to take place

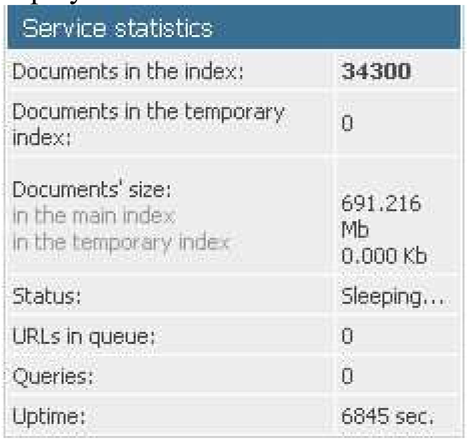
The downloading settings page also allows to choose the proxy server for the documents download.

Enter the host name and the proxy port. If the proxy server required authorization, enter the login and the password. You can also specify sites that do not require proxy server to download documents from.

2.3.4 Search service state

The status and numerical data on the service and documents collection can be monitored from the main page of the administrator's interface **Server/State**.

The statistic items are displayed in the section “Service statistics”.



Service statistics

Documents in the index:	**34300**
Documents in the temporary index:	0
Documents' size: in the main index in the temporary index	691.216 Mb 0.000 Kb
Status:	Sleeping...
URLs in queue:	0
Queries:	0
Uptime:	6845 sec.

Screenshot: server statistics

Documents in the index
The total number of documents in the main and temporary indexes.

Documents in the temporary index
The number of documents in the temporary index.

Documents' size
The total size of documents

in the main index
in the main index

in the temporary index
in the temporary index

Status

Indicates the current state of the search service. Possible modes:

Initializing	the process is being initialized
Updating	the search index is being complemented/updated
Flushing	the local cache is being copied to the disk
Merging	the temporary index is being integrated into the main one
Sleeping	no modifications on the search index at the moment
Reindexing	the documents in cache are being reindexed. It happens when indexing failed and the temporary index was lost.

Queries

The number of queries processed since the service is started.

URLs in queue

Shows the quantity of the URLs in the queue

Uptime

Time (in seconds) from the start

Last Updated URL

In the *Updating* mode, shows the last processed URL.

To retrieve the actual statistics, use "Refresh" button.

2.3.5 Search service administration

Changing operation mode

Choose the needed mode from the drop-down list and click the "*Apply*" button.

Mode description:

Search and update

The full update cycle of the search index.

-new URLs from the queue are being added,

-if the URL queue is empty, the URLs already included into the index are being updated,

-queries to the web-server are separated by a pause defined by the setting *"Interval between queries to web-server"*;

When the add/update cycle is over, the service is suspended for the time specified by the setting *"Interval between index updates".*

Search and add

New URLs are added to the search index. If the URL queue is empty, the service is switched to the *Sleeping* mode. The index is not updated.

Search only

No modifications of the search index. If this mode is activated while indexing, updating or merging are running, these processes are terminated, the data is synchronized, i.e. local cache is copied to the disk, the service is switched to the *Sleeping* mode.

Server-stress

The same as *"Search and update"*, but the parameters of waiting are ignored.

Reinitialization

For the new settings of the search service to take effect, use the "*Reinitialize*" button on the main administrator's interface.

Delete index

For unrecoverable removal of the index, push "*Delete*" button on the main administrator's interface.

Stop the service

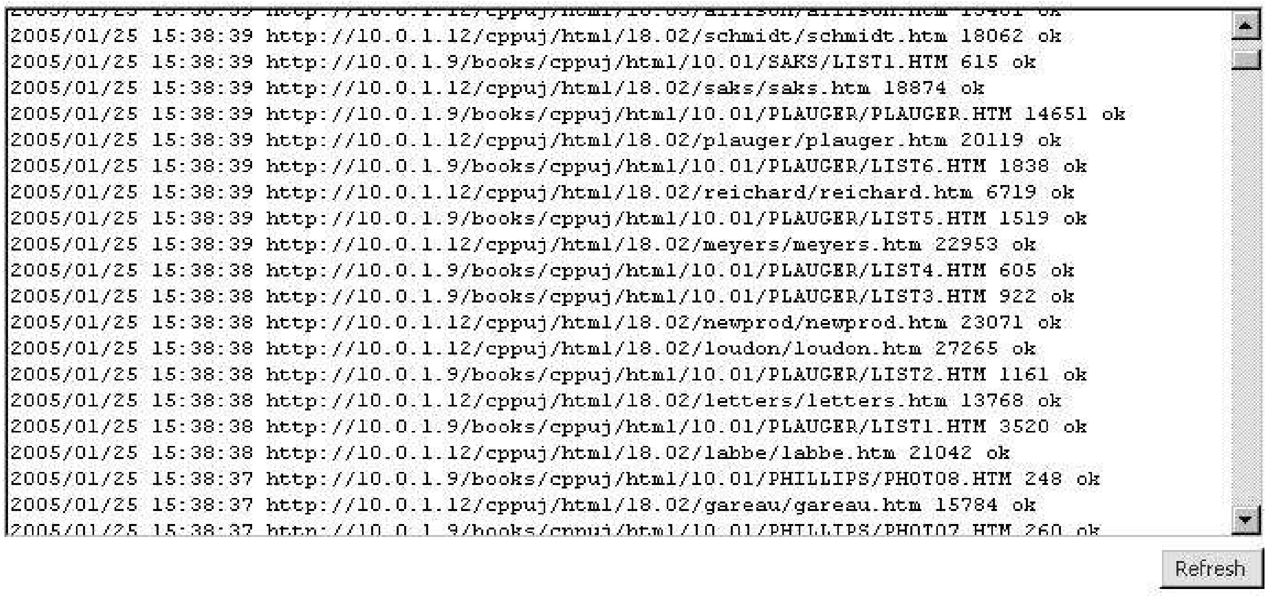
The button "*Stop service*" on the main administrator's interface will terminate the search service.

Merging the main and temporary indexes

Merging accelerates the search, optimizes hard drive transfers and reduces the search index size. Merging is the final stage of the index cleaning process; the data for search modes, i.e. grouping by sites and filters, are updated (the grouping of results is possible in the main index only).

To start merging, click on "Merge" button on the main administrator's interface.

2.3.6 View log file of the search service



```
2005/01/25 15:38:39 http://10.0.1.12/cppuj/html/18.02/schmidt/schmidt.htm 18062 ok
2005/01/25 15:38:39 http://10.0.1.9/books/cppuj/html/10.01/SAKS/LIST1.HTM 615 ok
2005/01/25 15:38:39 http://10.0.1.12/cppuj/html/18.02/saks/saks.htm 18874 ok
2005/01/25 15:38:39 http://10.0.1.9/books/cppuj/html/10.01/PLAUGER/PLAUGER.HTM 14651 ok
2005/01/25 15:38:39 http://10.0.1.12/cppuj/html/18.02/plauger/plauger.htm 20119 ok
2005/01/25 15:38:39 http://10.0.1.9/books/cppuj/html/10.01/PLAUGER/LIST6.HTM 1838 ok
2005/01/25 15:38:39 http://10.0.1.12/cppuj/html/18.02/reichard/reichard.htm 6719 ok
2005/01/25 15:38:39 http://10.0.1.9/books/cppuj/html/10.01/PLAUGER/LIST5.HTM 1519 ok
2005/01/25 15:38:39 http://10.0.1.12/cppuj/html/18.02/meyers/meyers.htm 22953 ok
2005/01/25 15:38:38 http://10.0.1.9/books/cppuj/html/10.01/PLAUGER/LIST4.HTM 605 ok
2005/01/25 15:38:38 http://10.0.1.9/books/cppuj/html/10.01/PLAUGER/LIST3.HTM 922 ok
2005/01/25 15:38:38 http://10.0.1.12/cppuj/html/18.02/newprod/newprod.htm 23071 ok
2005/01/25 15:38:38 http://10.0.1.12/cppuj/html/18.02/loudon/loudon.htm 27265 ok
2005/01/25 15:38:38 http://10.0.1.9/books/cppuj/html/10.01/PLAUGER/LIST2.HTM 1161 ok
2005/01/25 15:38:38 http://10.0.1.12/cppuj/html/18.02/letters/letters.htm 13768 ok
2005/01/25 15:38:38 http://10.0.1.9/books/cppuj/html/10.01/PLAUGER/LIST1.HTM 3520 ok
2005/01/25 15:38:38 http://10.0.1.12/cppuj/html/18.02/labbe/labbe.htm 21042 ok
2005/01/25 15:38:37 http://10.0.1.9/books/cppuj/html/10.01/PHILLIPS/PHOTO8.HTM 248 ok
2005/01/25 15:38:37 http://10.0.1.12/cppuj/html/18.02/gareau/gareau.htm 15784 ok
2005/01/25 15:38:37 http://10.0.1.9/books/cppuj/html/10.01/PHILLIPS/PHOTO7.HTM 260 ok
```

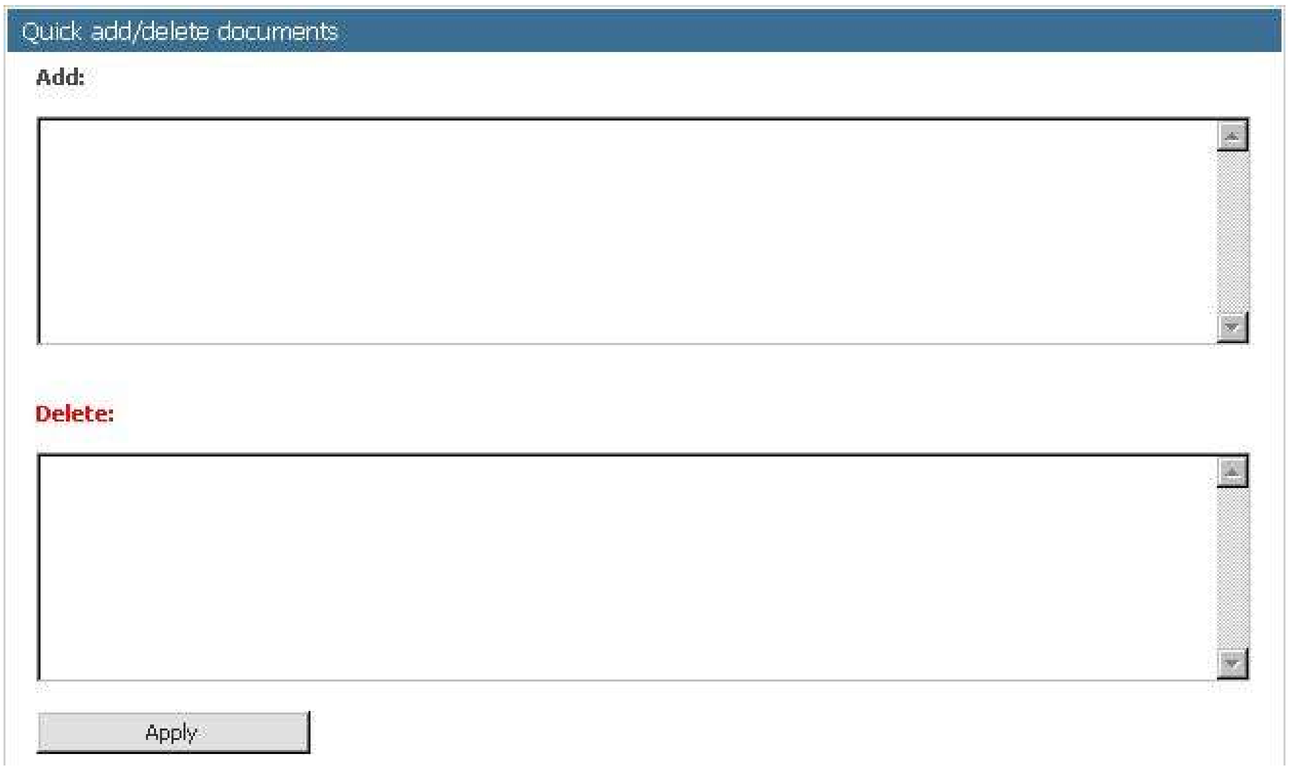
Screenshot: Log file view

The latest records in the log file can be viewed using the **Server/Log file** section on the administrator's interface.

2.3.7 Add/remove documents to/from the collection

URLs can be added/removed to/from the search index instantly, out of turn, in the section **Collection/Quick add**

Screenshot: Quick add



To put an URL to the first position in the indexing queue, insert it into the “*Add*” field.

Press “*Apply*” to send the command to the server.

The URLs can be added to the indexing queue only if they satisfy the permitting and stop masks.

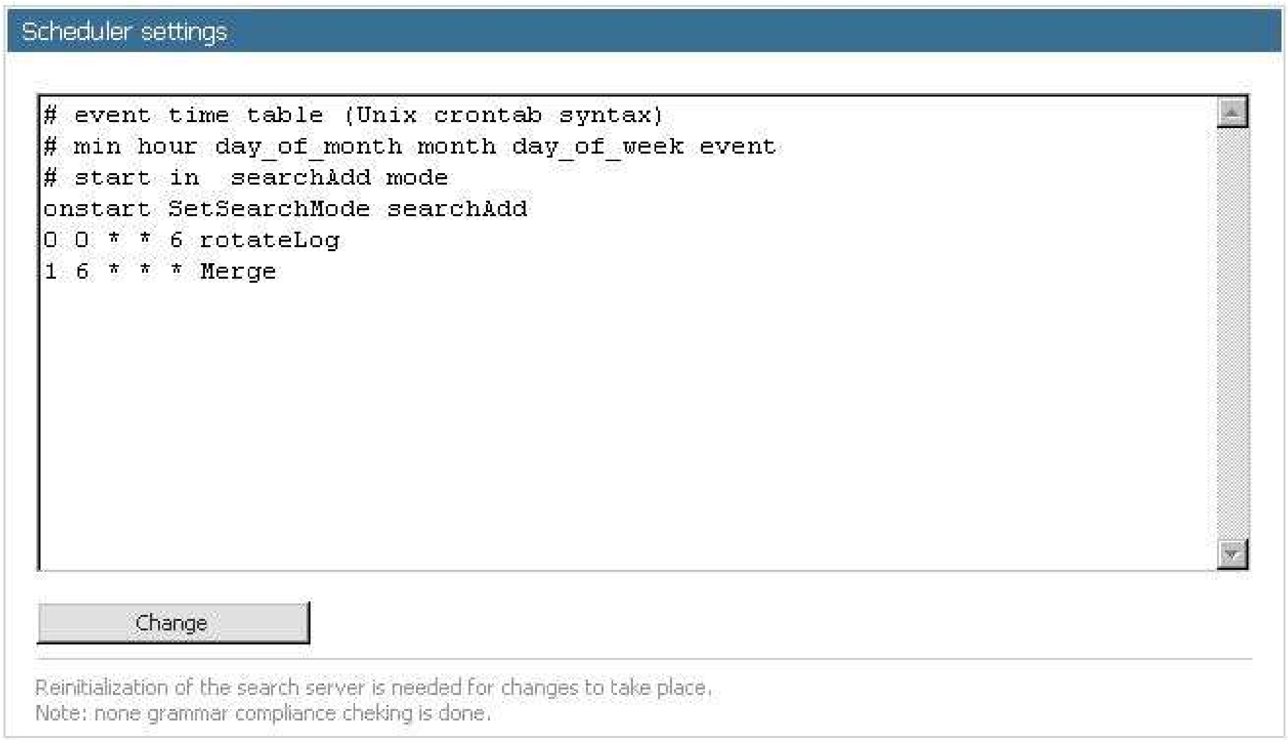
To remove an URL from the search results and enqueue it for instant removal, insert it in the “*Remove*” field, then press “*Apply*” to send the command to the server.

The quick add option is inactive during merging . Documents cannot be deleted before merging is over, don't check the documents as to be deleted.

2.3.8 Service scheduling

The search service offers the alternative option to balance the server load by scheduling the service activity.

The scheduler settings are accessible under the **Scheduler** section on the administrator's interface.



```
# event time table (Unix crontab syntax)
# min hour day_of_month month day_of_week event
# start in  searchAdd mode
onstart SetSearchMode searchAdd
0 0 * * 6 rotateLog
1 6 * * * Merge
```

Reinitialization of the search server is needed for changes to take place.
Note: none grammar compliance cheking is done.

Screenshot: Scheduler settings.

The scheduler file format is described in **Appendix A.**

To change the settings, make necessary changes in the input box in compliance with the syntax rules, and use the "*Change*" button.

NOTE: The changes will take effect only after reinitialization.

2.4 Customizing web interface and result pages

siteMETA allows the administrator to individually set the result page view and the errors page, using the template building option. The standard templates **template.htm** and **template_e.htm** are stored in the distributive folder **"./etc/"**. They can be modified to match your site design. The template names are determined by the variables **template** and **errtemplate** in section **[SSCGI]** of the ini file **mssCgiCfg.ini,** for example:

```
...
[SSCGI]
...
template=./yourtemplate.htm
errtemplate=./myerrtemplate.htm
...
```

A template is an HTML-file combining common hypertext mark-up elements with special pseudo-tags that are used by the CGI-script to generate the result page.
(See Table 1)

Pseudo tags can be global or internal. Internal pseudo tags can be only inside operator brackets made by their parent tags, whereas the global ones can be used outside any others.

Tag name	**Description**
Global tags	
<\`Query\`>	Query text
<\`NGroups\`>	Number of found groups of documents (one group may contain more than one document).
<\`FoundDocuments\`>	Number of found documents
<\`NextPageLink\`>	Link to the next search result page
<\`#include filename\`>	Include filename directive. The file must exist to be included.
<\`IdFirst\`>	Number of the first group of a portion
<\`IdFinal\`>	Number of the last group of a portion
<\`SearchForm\`>	Search form default tag
<\`form\`> </\`form\`>	Opening and closing tags of the search form. Inside them the search form HTML code must be entered
<\`ForEachDocument\`>- <\`/ForEachDocument\`>	Opening and closing tags of document iterator
Internal tags for the Form tag	
<\`Form.Site\`>	Transforms to *"selected"* in case the sites grouping mode is selected
<\`Form.Rank\`>	Transforms to *"selected"* in case the "Sort by relevance" mode is selected
<\`Form.Time\`>	Transforms to *"selected"* in case the "Sort by ascending date" mode is selected
<\`Form.Rtime\`>	Transforms to *"selected"* in case the "Sort by descending date" mode is selected
<\`Form.Site.Checked\`>	Transforms to *"checked"* in case the site grouping mode is selected
<\`Form.Rank.Checked\`>	Transforms to *"checked"* in case the "Sort by relevance" mode is selected
<\`Form.Time.Checked\`>	Transforms to *"checked"* in case the "Sort by ascending date" mode is selected
<\`Form.Rtime.Checked\`>	Transforms to *"checked"* in case the "Sort by discending date" mode is selected
<\`Form.Query\`>	Transforms to the encrypted text of the previous query
<\`Query\`>	Query text
<\`Form.MinTime\`>	Lower time limit (DD/MM/YYYY)
<\`Form.MaxTime\`>Internal tags for the ForEachDocument tag	Upper time limit (DD/MM/YYYY)
Internal tags for the ForEachDocument tag	
<\`Doc.Number\`>	Ordinal number of the found document (by default results are sorted by relevance)
<\`Doc.Title\`>	Document title description tag
<\`Doc.Language\`>	Document language description tag
<\`Doc.Quote\`>	Relevant text fragment description tag
<\`Doc.FoundTime\`>	Last document modification date
<\`Doc.IndexingTime\`>	Document indexing date

<\`Doc.Id\`>	Document identifier. It is used for making a request to reconstruct a document
<\`Doc.GroupName\`>	Site name used for the site grouping mode. This name is used to display all the site documents meeting the current query terms.
<\`Doc.GroupSize\`>	Number of documents found on one site
<\`Doc.Rubrics\`>	Text line with data on rubrics, if any in the report. String format: *<rubricID> <space> [<description> [newline]]*
<\`BaseUrl\`>	CGI script URL
<\`Doc.ForEachURL\`>- <\`/Doc.ForEachURL\`>	Opening and closing tags of URL iterator
Internal tags of the Doc.ForEachURL tag	
<\`URL\`>	Document URL
<\`URL.CodePage\`>	Document code page
<\`URL.IndexingTime\`>	Time of the last indexing of a document
<\`URL.Size\`>	Document size (in KB)

Table 1. Reserved pseudo tags.

To embed the search form into the result displaying template, you can use either a default form or a tunable form creating it with the help of the tag **<\`form\`>:**

```
<`form`>
<form action="" method="get">
<input type="text" maxlength="80" name="q" size="35" value="<`form.query`>">
<input type="submit" value="Search!">
<select name="mode">
<option <`form.rank`> value="rank">By relevance </option>
<option <`form.time`> value="time">By ascending date</option>
<option <`form.rtime`> value="rtime">By discending date</option>
</select>
</form>
<`/form`>
```

In **mssCgi** of the application the following parameters can be sent using the query:

Parameter	***Description***
q	query text
mode	search modes (see below) (calculating query mode)
site	Indicates the group with settings of a result display template in the initialization file. (*)
docID	Document identifier (only for queries of the "text" type)
siteurl	assigns a list of sites (separated by comas) which documents are to be displayed in the result pages
filter	assigns document selection condition according to document filters
grouptag	name of the field for grouping according to the filters
page	result page number
ds	lower time limit for document modification date (**)
dt	upper time limit for document modification date (**)
tlist	identifier for the rubric to be filtered
showrubric	If ***'on'*** is set, the search service receives a query to fill the fields with data on rubrics containing the found documents

Table 2: a list of parameters acceptable by the application

(*) – for the **mssCgi** application: in case a parameter is not specified or a group with this name is absent, the result display template can be set according to the settings in the **SSCGI** group.

(**) – used for searching for a specified period of time (date is to be in DD/MM/YY[YY] format).

Search modes (*mode* parameter value):

site	group results by site
rank	display results by document, sort by relevance
time	display by document, sort by document modification date (ascending)
rtime	display by document, sort by document modification date (descending)
text	display by document text that satisfies the *q* query specified by *docID*
grouptag	group results by filter specified by the *grouptag* parameter

Table 3: Search modes

3. Fine tuning of siteMETA

siteMETA search service is a powerful and flexible tool. The administrator's interface is not the only way to tune **siteMETA** search system. Fine tuning can be also done via the configuration files stored in the folder **"./etc"**. All the configuration files are text files consisted of lines. Usually, each line has two parameters: a name (or a key) and its meaning after the equal sign. If a line starts with "#", it is a comment so it is ignored. See the structure and purpose of the configuration files:

3.1 Setting up configuration files

3.1.1 File searchService.ini searchService.ini file stores the search server settings.

Key	Description
Section [System]	
Administration	Number of the port for the search server and administrator's interface cooperation
SearchExchange	Number of the port for the search server and CGI script co-operation
Log	Variable indicating the path to the log file
ListenInterface	Interface for the search server and CGI script co-operation, by default, it is ***localhost***.
Timeout	Timeout value to estimate the search query duration in milliseconds. By default is set 150000.
Section [UrlManager]	
RootUrlList	Variable indicating the path to the file containing the start URL(s)
UrlValidate	Variable indicating the path to the file with URL permit/deny masks URL
SaveUrlList	Variable indicating the path to the file with a list of retrieved URLs to be indexed
ForceUpdate	If the variable is set to *no* position, the documents with unchanged modification time will not be reindexed while updating the index. In order to reindex statical documents after changing the set of the allowed URL masks, download limitations or linguistic processing, the variable should be set to *yes*.
NoRemove	The variable determines how to process the event when the transport module fails to connect to the server hosting the required documents. If it is set to *yes*, the documents will not be deleted from the index. Otherwise such documents are considered as not found and will be removed from the index.
DeniedParams	If the variable is set to *on*, the URL parameter list will be cleared from PHP-session specific parameters.
DumpBadUrls	If ***'yes'***, by download failures the server records the URL and the return code of the web server in the log file.
Section [Indexing]	
CronTab	The Variable contains the path to the scheduler ini file (optional)
TimeBetween	Pause between documents indexing (in seconds), minimum 1 sec.
reindexTime	Pause between reindexing cycles (in minutes)
Mode	sets the operation mode of the search service. Possible values: ***All*** - search and index updating (by default); ***SearchAdd*** - search and add documents; ***Stress*** - search and index updating without pauses; ***SearchOnly*** - search only, without indexing.

LoUrls	If ***'YES'***, converts the symbols of all URLs found at the indexed pages to lower case. Doesn't influence the start URLs and the permitting/stop masks.
MaxDynDoc	Maximum number of documents in the supplementary database.
StoreTasks	If the variable is set to ***YES***, the downloaded documents will be stored in the local cache to prevent downloading them repeatedly by a program failure.
UsePredictedCP	If the variable is set to ***YES***, when the document format filter retrieves the supposed document encoding and the latter matches the list of supported encodings, the document will be recoded according to the encoding specified in the filter.
Section [Database]	
StPath	path to the search index
RecCacheSize	cache size (number of entries) to build the coordinates (by default it is zero, the recommended value for extensive sites - 1000000)
LinkList	Path to the text file with the link list description. If this file is indicated and exists, it can be used to generate binary file ***<StPath>.link.bin,*** that is used to initialize rubric-restricted search. File ****.link.bin*** is generated from the administrator's interface or with the utility ***mLinkList***
LinkList_Descr	Path to the text file with the rubric identifier description. Every rubric identifier in the file specified by *LinkList* parameter can be matched with a one-line description (name) to be used by result presentation (see **showrubric** and **<`Doc.rubrics`>**). The file content can be modified using the administrator's interface, section Rubrics.
Section [Parser.Settings]	
TextLimit	maximum size of the document text (in bytes, without formatting) that will be processed by the indexing module.
Wildcards	(De-)/Activation of wildcard support. Possible values: ***YES, NO***. For large indexes ***NO*** should be set
Section [Format filters]	
html	path to HTML document filter
Section [Format filters cfg]	
html	path to HTML document filter parameters (optional)
Section [http]	
BrowserName	Name used for the indexing module during uploading document from your website.
AcceptMask	the acceptable MIME-coding
MaxDownload	maximum data volume that can be downloaded by the transport module (by default 4Mb)
HTTP_Proxy	IP and port of the proxy-server in the format hostname:port (optional)
Proxy_User	User name for authentication (optional)
Proxy_Passwd	User password for authentication (optional)
No_Proxy	List of hosts, separated by commas, which should be referred by-passing the proxy-server (optional)
Timeout	Timeout value for downloading documents, in milliseconds. By default - 10000. Maximum value - 180000
Tries	The variable sets how many times the service should try to connect to the web-server. By default - 1. Maximum value - 20.
X-Metadata	Assigns names list for document metadata extraction from the HTTP-header while downloading
Section [Quotes]	

MaxChars	maximum length (in characters) of the quotation if selected the quotation mode **fulltext**, or by text reconstruction
MaxWords	maximum length (in words) of the quotation if selected the quotation mode **quote** (by default - 48)
Section [Structures]	
TagTable	path to the ini file of the tag and grouping table (available only for search through structured data)
Section [Query.Settings]	
StopWords	path to the ini file of the stop word vocabulary
Sections [DbModule.Static] and [DbModule.Dynamic]	
DatabaseModule	path to the database administration module file is used as a statical or dynamical index database, accordingly.

3.1.2 File mssAdmHCfg.ini

mssAdmHCfg.ini file contains parameters of the administrator's interface.

Key	Description
	Section [System]
Administration	Number of the port for the search server and the administrator's interface connection. The same number must be set for ***Administration*** key in ***[System]*** section of the file ***searchService.ini***.
ListenInterface	Interface used for the administrator's interface and search server connection. By default it is ***localhost***.
Template	path to the administrator's interface template file
Log	path to the log file

3.1.3 File mssCgiCfg.ini

File **mssCgiCfg.ini** stores the CGI script parameters forming search queries and result pages.

Key	Description
Section [System]	
SearchExchange	Port number for interaction between the CGI-script and the search service. The same port number should be set for the key ***SearchExchange*** in the section ***[System]*** of the ***searchService.ini*** file.
ListenInterface	is the interface between the administrator's interface and the search service. By default it is ***localhost***.
Log	the path to the log file
Section [SSCGI]	
template	the variable defines the path to the template of the result page generated by the CGI-script
errtemplate	the variable contains the path to the template of the error page. The error page is shown when no matches were found. If this variable is not set, the error message will be shown in the standard format specified in the ***template*** file.
grouptemplate	path to the result page template for result grouping requests (by sites, by filter). If omitted, the template specified in the ***template*** variable is used by default.
ResultCount	number of documents displayed on one page
MaxUrlCount	maximum number of URLs one document may have
QuoteHiliteOpen	opening HTML tags used for text reconstruction during the extraction of a relevant fragment

QuoteHiliteClose	Closing HTML tags used for text reconstruction during the extraction of a relevant fragment

NOTE: For text reconstruction mode it's possible to navigate through the found document words. To activate this feature, you should redefine the tags QuoteHilite*. For example:

QuoteHiliteOpen=<a name='$META_CUR$' href='"#"$META_PREV$'><b><font color=red>&lt";"&lt";"||</font></b></a>&nbsp";"

QuoteHiliteClose=&nbsp";"<a href='"#"$META_NEXT$'><b><font color=red>|| &gt";"&gt";"</font></b></a>

The used temporary templates will be replaced by the template processor with strings of the format METANNNNNN, where N are decimal numbers, describing the order number of the found word or phrase, and can be used to generate named links on the page. Temporary templates are replaced with strings like that:

$META_CUR$ - number in the string – current order number of the highlighted phrase;
$META_NEXT$ - number in the string – the one following the current one;
$META_PREV$ - number in the string - previous, except for first entering.

NOTE: Remember that the semicolon (;) and sharp (#) symbols used in the ini files should be put in double quotes. Otherwise they are interpreted as a beginning of a string comment. Double quote symbols (“) should be introduced by slash (\).

3.1.4 File rootlist.txt

The file **rootlist.txt** stores a list of URLs to be indexed. In fact, this file is an analog of the start URLs field in the administrator's interface.
NOTE: for a site to be available for indexing, its start URL of the format http://somesite/<somepath> should be included into the start URL list. Otherwise all documents from this site will be checked as not satisfying the mask list, thus not available for indexing.

3.1.5 File urlmasks.txt

The file **urlmasks.txt** contains the stop mask (template) list. It is a standard text file with permit and stop URL masks. Mask syntax is quite simple:
«*» - the wildcard symbol allows to index documents according to the mask. For example, the mask ***meta.ua*** allows indexing all sites from the site “**meta.ua**”.

«!» - index restriction mask. For example, the mask **!*meta.ua/en/*** forbids indexing documents stored in the directory “**en**” of the site “**meta.ua**”.

The order of the masks implementation is set according to their query in the file **urlmasks.txt**.

3.1.6 File saveurls.txt

The file **saveurls.txt** stores a list of extracted URLs to be indexed. In case the search service shuts down, this file is used to continue indexing from the page it stopped indexing at.

3.1.7 File cron.ini

siteMETA offers an alternative mechanism of controlling the server load through the embedded scheduler. The file **cron.ini** stores the settings of the **siteMETA** scheduler.

For more details on the scheduler file please see **Appendix A.**

3.1.8 File tagtable.ini

The file **tagtable.ini** stores the settings for the tag and result grouping table.
For more details on the file format of the tag table settings please see **Appendix B.**

This file determines the way the index module treats the tags inside the document.

Currently the search core supports the following tags:

- text mark-up tag;
- tag filter;
- tag filter for metadata.

Each tag is set in the ini file **tagtable.ini** by a group with a unique name according to the tag name in lower case. The first two tag types require also the field **id**, describing the unique tag identifier (a natural number from **1 to 65536**), while every identifier corresponds to only one tag name. These types of tags should be listed in the group **[tags]** of the ini file of the HTML document filter:
tag_name=corresponding_identifier.

NOTE*:* The list of the allowed mark-up tags for HTML documents is fixed and cannot be extended. These tags can be activated only by using the key **search**, i.e. included or excluded from the search. The tag identifiers in the range **1 to 20** are reserved and should not be used in the user-defined filters.

In the tag filters for meta-data, the tag identifier is defined by the search core, so the field **id** can be ignored.

3.1.8.1 Text mark-up tags

This group of tags define the way the index module treats the mark-up tags. This class of tags should necessarily contain the tag identifier (field **id**). This type of tag is set by either the key **format**, or the key **zone** (the value is an integer constant within the range **[1, 32]**). These keys are normally fixed-preset and cannot be changed, hence it is no use to describe them in details here. The description of these tags can also include the key **search** with the possible values **yes** (allow the search within the tag content) or **no** (forbid the search within the tag content). **yes** is the default value.
The **search** setting is the indication for the index module that the tag content should be ignored while searching, though included into the extracted result phrase. This setting can be useful, for instance, to exclude from the search a document portion, which does not belongs to the content (is an element of the site design or a site section,recurring on every page/section).

For example, for news sites it would be expedient to exclude the tags **<form>** and **<a>.**

3.1.8.2 Tag filter

This class of tags serves for filtering documents by a certain numerical parameter, e.g. signed integers, unsigned and real integers - s. parameter **type**). The tag filters are defined by a group including the fields:

```
[tagname]
id=tag_identifier
filter=yes
type={ int | word | float }
```

Further on, while searching the documents containing tag filters, the filtering by the filter-fields can be applied.
Available operations: **==, !=, <, <=, >, >=.**

NOTE: filtering parameters cannot be defined through the CGI search application. You should use a search COM object for MS IIS 5.0 or higher to be able to apply filters. Description of the COM object can be found in Appendix C.

3.1.8.3 Tag filters for metadata

These are tag filters allowing grouping the search results by the tag value. The tag filters for metadata can be used after additional tuning of the site pages subject to grouping.

Example of a tag description in **tagtable.ini**:

```
[gid]
metaname=gid
filter=yes
type=int
```

The values of these grouping tags are specified in the section HEAD of the HTML-document:

<meta name="gid" content="value_type_int">.

The grouping rules depending on the meta data tag filters are given in **Appendix C.**

3.1.9 Initialization file format for the stop-word list

By default, the **siteMETA** system searches all words, including spurious and non-informative ones, e.g. prepositions, conjunctions, certain adverbs, etc.). However, coordinate data extraction can be rather time-consuming by overloads and large indexed data volumes. In such situation, a list of stop-words to be ignored is a big relief.

The ini file of the stop-word vocabulary is a simple text file, every line containing a lexeme subject to delete from the search query.

NOTE: that if you add a word from a morphological dictionary into the stop-word list, all its word forms will be automatically considered as stop-words.

It is also possible to exclude an exact word form. Syntax example:

<word>/<refining word form>

Example: "*simple/simplest*" - the adjective "**simple**" and all its derivatives will be added to the stop-word list. The lexemes "**simple(subj.)**" and "**simplify(verb)**" will not be considered stop-words.

3.1.10 Document metadata transfer through HTTP-headers.

In case that some metadata that is specific for a certain document cannot be transferred because of some reasons by means of the document itself, the metadata can be added to the HTTP web-server header.

Headers that have numerical values for transfer must be pointed according to the following grammar:

X-METASE-<NAME>: <NUMBER>

where *NAME* is metadata tag filter name from the file tagTable.ini which was also initialized in the HTTP-transport initialization group (see 3.1.1 **x-metadata** key description)

([http]::x-metadata = comma-separated-name-list)

NUMBER – numerical value from range of a certain tag-filter type according to the description in tagTable.ini file.

For each name assigned in the [http]::x-metadata list at each document downloading http-transport creates search header named X-METASE-<NAME>, found value pairs are kept for further indexing use.

Also HTTP-header can be used for sending a line which will be used as a title of a document while indexing and searching. Such a header has the following view:

X-METASE-TITLE: <TITLE-STRING>

where

TITLE-STRING – line, pointing an alternative document title.

TITLE-STRING must be pointed according to the following grammar.

TITLE-STRING:

=?windows-1251?U?<url-encoded-string>?=

<url-encoded-string>:

the line is in the windows-1251 encoding, all format forbidden URL characters which are encoded into sequences %<hex-number> where %<hex-number> is a symbol made of 16 numbers in windows-1251 encoding.

NOTE: only printed characters can be used for alternative document header line creation.

Example of an HTTP-header:

```
X-METASE-TITLE: =?windows-1251?U?Substitution%20of%20a%20title?=
X-METASE-TOPIC: 255
```

The value of the document title will be set in the value “Substitution of a title”, value of the tag-filter of metadata – 255.

Initialization files of the service need to be changed for processing this type of headers:

searchService.ini:

```
[http]
...
x-metadata=title,topic
```

tagTable.ini:

```
[topic]
metaname=topic
filter=yes
search=no
type=int
```

For changes in the tagTable.ini file to take place search database needs to be deleted.

3.2 Special features of siteMETA

3.2.1 Indexing documents that have several versions in different encodings

siteMETA search engine uses algorithms developed for page encoding and document language identification as well as advanced technologies for finding duplicate documents. That means that documents different only in titles and encodings will be grouped and placed in the index as one document that has several URLs.

3.2.2 Removing documents from the index. Deleting the index

Sometimes you may need to remove certain files from the **siteMETA** index, or completely renew the index content without switching off the search service. If you need to delete a single file, just insert its URL into the stop rules list using the administrator's interface. The file will be deleted during the next full indexing cycle.

Depending on the settings and the site size, the complete reindexing may take from several minutes to several hours or even days. If you have done any essential changes in the search service settings and do not want to wait till the complete indexing cycle is over, you can speed up the process by deleting the entire index.

3.2.3 Using cache while indexing

To increase the efficiency of indexing multi-page sites, we offer the feature of indexing with caching. The side effect is that the document are indexed very quickly, but they are available for searching only when all cache content is added to the database.

The cache size can be determined using the key **RecCacheSize** of the group **Database** in the ini file **searchService.ini.**

The recommended key value for large sites is 1000000 elements (required memory – about 40 Mb).

The maximum possible value is 2000000 entries.

3.2.4 Indexing modes

siteMETA supports the modes**:**

«Search and update»

The search service indexes newly detected files, reindexes the index content and searches. The mode «Search and reindex» is set by default.

«Search and add»

The search service adds newly detected documents and searches. The existent index is not reindexed.

«Search only»

The search service runs search over the index content. No adding, no reindexing.

«Server-stress»

Newly detected documents are added to the index, the index content is updateed, search is allowed. Indexing and reindexing go with the maximum possible rate without pauses.

3.2.5 Setting-up pages

While indexing, **siteMETA** collects the metadata of the HTML-pages, e.g. “**Robots”**, defining the document processing mode.

The acceptable values of this meta-tag are:

Name	Value
ALL	default value. The file will be indexed, tracking all found links
INDEX	index this page

FOLLOW	track all links found in the page
NOINDEX	do not index this page
NOFOLLOW	do not track the found links
NONE	do not index and do not track the links

For example, an HTML-page, containing in the section **HEAD** the line

<META NAME=”ROBOTS” CONTENT=”NOINDEX, FOLLOW”>,

will not be indexed, but the found links will be tracked and the linked files will be indexed.

Additionally, it is possible to set the document date for sites with dynamic pages. The default date of the document is the date indicated in the HTTP-header DATE of the page, or the indexation date when no DATE header is available. To change the document date, fill in the HTTP-header “DocumentDate”.

Acceptable date formats are:

Format RFC 822/RFC 1123:
Sun, 06 Nov 1994 08:49:37 GMT

Format RFC 850/RFC 1036:
Sunday, 06-Nov-94 08:49:37 GMT

Format of function asctime() from the standard library of the C language:
Sun Nov 6 08:49:37 1994

4. Query language

The search service has an advanced query language that includes Boolean operators, refinement operators, field restriction operators. Unlike most search engines, **siteMETA** correctly works with the user-defined stop word list, ignoring unwanted results.

Boolean operators

Operator	Description
+	Boolean AND. This operator is applied by default. It searches for records that contain both of the words it separates, i.e. the query ***cocktail recipe*** is equal to the query ***cocktail + recipe***.
-	Boolean AND NOT. It lets you exclude documents where an entered word is present. For instance, for the query ***cocktail -vodka*** results will contain documents where there is the word ***cocktail*** but there is no word ***vodka***.
|	Boolean OR. It lets you find documents that contain any of given words. For instance, the query ***seafood | shrimp*** reveals documents which contain ***seafood*** or ***shrimp*** or both words.

Operators priority may be altered with the sequential operator "( )". If you need to find documents containing phrases ***vacation at the Bahamas*** or ***vacation at the Caribbean***, you can use the brackets to specify the operators order: ***vacation at the (Bahamas | Caribbean)***.

Refinement operators

Operator	Description
"..."	The double quote signs let you search for documents containing exact phrases. An double quoted query returns documents which contain words of the same form and in the same order as the keywords. Also one symbol constructions that are parts of query language like: *** ? / ! ( ) [ ] =,** will not be found without this operator.
{...}	The search operator **{...}** allows to find phrases which are close to that in the brackets, i.e. unlike the operator **"..."**, the query **{dress shop}** will return documents containing phrases: **dress shop, dressing shop, dress shops, dress shopping**, etc., i.e. the grammatical flexions are not taken under consideration.
[n, ...]	This operator lets you revise you query by limiting the distance between keywords. For instance, the query ***[5, mobile phone]*** returns documents containing both words ***mobile*** and ***phone*** with the distance up to 5 words between them.

Field restriction operators

Operator	Description
title	This operator searches only across document titles, but not the entire document content. For example, the query ***title(prices)*** finds documents containing the word ***prices*** in their titles, and the query ***title("technical documentation")*** returns documents containing the phrase ***technical documentation*** in their titles.
heading	This operator searches only throughout documents headings For example, the query ***heading(requirements)*** returns documents containing ***requirements*** in the tags **heading**.
url	This operator searches only document URLs. For example, the query ***url=www.meta.ua/doc/about.asp*** returns documents with this exact URL. The operator supports the wildcard operator ***"*"***, i.e. the query ***url=www.meta.ua/doc/**** returns documents containing URLs meeting this query.

Appendix A

Scheduler initialization file format

The file may contain empty lines, comments and commands.
A comment is a line starting with the symbol “#”.

The commands can be divided into start (executed at the launch) and periodic (time of execution set by a field list).

A command is set with the pair “**type-event**”.

Type

Name	Description
onstart	Start command
Fields list	Periodic command

The event is set with the pair **“name-value”**.

Event

Name	Value	Description
SetSearchMode	*searchOnly*	switches the search service to the mode “**Search only”**
	searchAdd	switches the search service to the mode “**Search and add”**
	stress	switches the search service to the mode “**Server-Stress”**
	All	switches the search service to the mode “**Search and update”**
SetInterleave	*number*	sets the minimum pause interval between web server requests (in seconds). The minimum value is 1 second
Merge		Forced merging of the temporary and the main indexes. **NOTE**: index should be merged to provide correct search with site or filter grouping. In these modes, the search in the temporary index is unavailable, except for the cases when the main index is empty.
ClearQueue		Clears the URL indexing and updating queue. Stops updating and starts adding new documents (***searchAdd*** mode). New URLs to be indexed are sought in the set of start URLs list.
AddUrl	*URL-to-index*	Adds URL to the queue for immediate indexing.

Periodic events are described with a list of fields setting minutes, hours, days of month, months, and days of week:
Field list - minute hour day_of_month month day_of_week

Field values

All the time units described above may be set with an exact value, a list, a period, a mask, and a **(step value).**
A period is set with 2 numbers separated by a dash.

For example, the value **8-11** for the field “**hour”** sets a command execution at **8, 9, 10** and **11** o'clock.

A list is set with a set of numbers and periods separated by coma.

For example, "**1, 2, 5, 9**", "**0-4, 8-12**".

A mask is an asterisks "*****". If a parameter is set with a mask, it is not taken under cnsideration.

Step values can be used together with periods. A period that is followed by the combination "**/<number>**" sets the value of the step at which an event is executed during the period.

For example, "0-23/2" can be used for the field "**hour"** to set a command execution every two hours during the 12 am - 11 pm period. You can also do it using the mask "***/2**".

NOTE: A day of a command execution can be set by two fields: day of month and day of week. If the both fields are set and not equal to "*****", the command will be executed when any of the values match the real one.

For example, "**30 4 1, 15 * 5"** sets the command execution at the 1st and 15th days of every month plus every Friday.

Sample scheduler initialization file

```
# start in  searchOnly mode
onstart SetSearchMode searchOnly

# event time table (Unix crontab syntax)
# min hour day_of_month month day_of_week event
# set SearchMode to All on 01:01 everyday
1 1 * * * SetSearchMode All

# set SearchOnly mode on 06:01 everyday


# Clear URL queue
# (Use to stop adding or updating URLs from the cache.)
10 6 * * * ClearQueue
# Implicit index merging.
12 6 * * * Merge
```

Appendix B

siteMETA search system allows to tune HTML document indexing by indicating which document fields to index in a special file. **siteMETA** also supports an alternative way of grouping server pages, **for example**, for the servers with completely dynamic data grouping by URL is almost impossible. The file that stores these settings is specified in the key **TagTable** of the group **[Structures]** of the service initialization file.

Grouping and tag table initialization file format

The file is a standard initialization file containing empty lines, key groups, and "**key-value**" pairs.

HTML tag settings

Each supported tag is set by a separate group with a unique name (e.g., tag name) and a standard key set.

There are two types of tags: index tags and filters.

The **Index tags** are set the following way:

```
[tagname]
id=<uniquetagid>
zone=<uniquetagzone>
search={yes|no}
quote={yes|no}
```

where:

tagname - name of the tag
id - tag identifier
search - indicates whether to take into account the tag value or not
zone - unique number in the range [1, 32] (for the tags used in search only)

The **filters** are set the following way:

```
[tagname]
id=<uniquetagid>
filter=yes
type={ int | word | float }
```

The filters can be signed integers, unsigned integers, and real numbers.

NOTE: The **tagname** name and the **id** tag identifier must match the corresponding pair in the group **[tags]** of the HTML filter initialization file. The path to this file is specified by the **html** key value of the **[Format filters cfg]** group of the search service initialization file **(searchService.ini)**. Otherwise, the tag settings will be ignored.

Setting alternative groupings

The alternative grouping settings are done by adding the following tag:

```
[groupname]
metaname=<groupname>
filter=yes
type=int
```

All the HTML documents of the **head** section containing the meta tag <**meta name="<groupname>" content="<integer>"**> can be grouped according to the value of the **content** attribute.

Appendix C

If you use **Microsoft IIS 5.0** and higher as a web server for the **siteMETA** search service, you will need a COM object kit for result page generation with script languages **(JScript, VBScript).**

Search interface object hierarchy

SFactory object is created by starting the web application.

The method **SFactory.CreateReport()** creates **SReport** object that transmits the user's request for processing and then returns the results to the web application.

SReport has an object collection **SGroup** for result presentation.

Every **SGroup** element bears information on documents in form of **SDoc** object collection sorted by the query relevance. **SDoc** returns the document's URLs collection.

Example:

```
Report  (SReport)
   Group(0)  (SGroup)
     Doc(0)  (SDoc)
       Url(0) (SUrl)
     ..............
     Doc(#)  (SDoc)
       Url(0) (SUrl)
       Url(1) (SUrl)

   Group(1)  (SGroup)
     Doc(0)  (SDoc)
       Url(0) (SUrl)
     ..............
     Doc(#)  (SDoc)
       Url(0) (SUrl)
       Url(1) (SUrl)
       Url(2) (SUrl)
    .................
```

Object SFactory

SFactory is a COM object for data exchange between search servers and clients.

Properties and methods of SFactory object:

Function

void AddServer(BSTR newServer, long Port)

Mission: adds a search server.

Arguments:

newServer – server's IP or **hostname**

Port – port number

Object property

Name	Value
long factoryTimeOut	timeout duration while waiting respond from the search server, in milliseconds. When the timeout is over in a method referring to the server, the method returns ***F_ERROR_RECEIVE***
long clipSize	number of groups in the report (default value for ***SReport*** objects)
BSTR qouteHiliteOpen	opening HTML tags, marking a relevant fragment in the text portion (initial value change for all created ***SReport*** objects)
BSTR qouteHiliteClose	closing HTML tags marking a relevant fragment in the text portion (initial value change for all created ***SReport*** objects)
long maxUrlTextLen	maximal length of resource locators
long maxTitleLen	maximal length of the document title
long groupMode	default grouping mode (default value for ***SReport*** objects)

Function

VARIANT CreateReport()

Mission: creates an empty search report.

Return value: a search report object. Afterward should be deleted by using the **Free** method of the report object.

Function

long ProcessQuery(VARIANT vReport)

Mission: processes the search queries.

Arguments:

vReport is an object containing the search query data. Should be created using the **CreateReport()** method**.**

Return value: if operation successful, returns **S_OK**, and **vReport** argument contains the search query processing results.

Possible errors in the return values

Name	Error code	Description
F_ERROR_CONNECT	101	connection to the search server failed
F_ERROR_SEND	102	query transmission failed
F_ERROR_RECEIVE	103	report reception failed
F_COM_ERROR	110	error in COM object (possible if the object class is not transmitted)
F_COM_NO_MEM	111	memory allotment error
F_COM_NO_DAEMONS	112	search daemon is not defined

Object properties

Name	Value
BSTR logFileName	path to the object's log file

Object SReport

SReport is a COM object for search query data transfer from the client to the server, and returning the results from the server to the client.

Contains query parameters, and after processing – all search match descriptions.

Object properties

Name	Value
long errCode	error code

Possible errors in the return values

Error code	Description
0	no error. User's query error
1	unary operator **NOT**
2	unary operator **AND**
3	unary operator **OR**
4	exact match quotes missing
5	closely adjacent phrase brackets missing
6	normal form identifier is not a word
7	empty set returned by specifying the normal form
8	missing comma in **[num, ...]**
9	missing brackets in operator **title(...)**
10	missing brackets in operator **heading(...)**
14	missing word following the hyphen (a word is ending with a hyphen)
15	this command is not available at this position
16	function is not supported
17	the lexeme is in the stop word list
18	**URL** length is less than allowed
19	query processing errors
101	connection to search server failed
102	query transmission failed
103	query reception failed
105	query processing error

Object properties

Name	Value
BSTR filter	defines filter values in the format compatible with the C language expressions: **"filter1 logic-op filter2 logic-op ... logic-op filterN",** where **filter1** is a logical expression for the filter in the form: **<filter_name><operation><value>** where **<filter_name>** is the filtering field name; **<operation>** is one of **> < >= <= == !=** **<value>** is the value within one filter mode **(int, word, float)**; **logic-op** – one of && || **NOTE**: filter names should correspond to the tag names specified in the ini file **tagtable.ini**.
long groupMode	grouping parameter
BSTR groupField	Meta tag name for grouping by value. Is used together with the grouping parameter *SortByField.*
BSTR strDate	Search time limitation. Receives a string of the format [dd/mm/yy[yy]-dd/mm/yy[yy]], determining the date range for the document modification date.
BSTR siteUrl	Assigns a list of sites, divided by comma, from which documents that satisfy the request will be shown.

Grouping parameter values

Parameter name	Value	Description
Undefined	0	default value
SortBySite	1	search result grouping by sites
SortByRank	2	search result presentation: documents one by one, sorting: by query relevance
SortByTimeInc	3	search result presentation: documents one by one, sorting: by ascending date
SortByTimeDec	4	search result presentation: documents one by one, sorting: by descending date
Reserved	5	*reserved to be defined later*
GetDocText	6	retrieve the document text. Is used together with **DocID**
GroupByField	7	result grouping by meta data tag filter value
LastIndexed	8	documents issue document after document, ranged by date of indexing, query calculation is not done.
GroupByField_LastIndexed	9	The same as G*roupByField* but documents sorting in a group is done according to the descending date, query calculation is not done.

Object properties

Name	Description
long docID	specifies the identifier of the document to be reconstructed. Is used together with the grouping parameter **GetDocText**

BSTR strQuery	query string
long first	first document or group number in the portion
long clipSize	portion size
long groupCount	number of groups in the portion
long groupFoundCount	total number of the groups found by this query
long docFoundCount	total number of the documents found by this query
BSTR quoteHiliteOpen	opening HTML tags marking a relevant fragment in the text portion
BSTR quoteHiliteClose	closing HTML tags marking a relevant fragment in the text portion
long minTime	Assigns search time limit. Accepts timestamp (time in seconds that has passed since January 1st 1970). Assigns the lower date border for documents modification date.
long maxTime	The same as *minTime* Assigns the lower date border for documents modification date.

Function
VARIANT Group(long lNum)
Mission: returns a group object by the number in portion. Group numbering starts from zero.

Function
void ExtendedInfo(ULONG flags, BSTR reserved)
Mission: Sets extended request flags by the search daemon.
Arguments:
flags
0x01 – retrieve rubric names.

Function
long ProcessQueryAt(BSTR searchDaemon)
Mission: sends request which has set up parameters in **SReport**, to the search service which is set by parameter **searchDaemon.**
Arguments:
SearchDaemon
<hostname:port> - points the host name and the port for sending the request to the search service.
Return value the same as **SFactory::ProcessQuery()**.

Object SGroup
Object SGroup is a group of documents that have a common parameter. The grouping parameter is specified in the query. For the groupings **SortBySite** and **SortByField** an object can contain more than one document. For the rest ones – only one document.

Object properties

Name	Description
BSTR siteLink	common part of the path for documents (site URL, document groups). While searching with the grouping parameter **SortByField** contains the meta data tag filter value, that is defined in the property **GroupField** of the object **SReport**, for this document group.
BSTR siteHref	link text to go through **sitelink**. To form a correct **html-**attribute of **href** link there is a slash in the end

long groupNum	group number in the report
BSTR quote	text portion of the first document in the group
single rate	group weight (relevance)
long docCount	number of document in the group
long docFoundCount	total number of relevant documents for a group

Function
VARIANT Doc(long lNum)
Mission: returns document from the group by its number.
Return value: object **SDoc.**

Function
BSTR Tag(BSTR tagName)
Mission: returns strings with rubric data.
Argument: parameter is the string **$rubrics**
Return value: string with rubric data in the form:
[<rubricID> <space> [<description[newline> ...]]]

Object SDoc
Object **SDoc** – document presentation.

Function
void WriteQuote(IStream* response)
Mission: prints the quote exactly to the output stream.
Argument: as a parameter accepts pointer to the stream object for record.

Function:
BSTR ExtendedQuote(BSTR cmdString, BSTR cmdParams)
Mission: is used for flexible information showing, that is somehow connected to quoting.
Arguments:
cmdString
name of command for showing.
cmdParams
the command parameters

Return value: returns a line corresponding to the entered parameters.

List of commands and parameters:

Command name	***Command parameters description***
"quote"	Returns fragment of quote for a certain group. If **cmdParam** is an empty line, then request is referencing to the value *"quote"*. In case the search service was configured for more than one quoting of document generation, parameter **cmdParam** can contain ID number (line number from 0 to 16) of a fragment, which needs to be shown. If the object does not contain of a fragment with such a number then an empty line will be returned.

Command name	***Command parameters description***
“lastMark”	Command “lastMark” is used to show the quantity of highlighted search terms in the quote and can be used to navigate to the last highlighted element in quote. The value of command “lastMark” will be different from null only in case of using in templates QuoteHilite* tags of placing $META_CUR$, $META_PREV$ и $META_NEXT$.

Object SDoc
Object SDoc is a representation of a document found.

Object properties

Name	**Description**
long docID	document identifier
single rate	document relevance
BSTR title	document title
BSTR lang	document language
long docNum	document number in the group
long size	document size (in bytes)
long urlCount	number of resource locators linked to the document

Function
VARIANT Url(long lNum);
Mission: returns document's URL by its number.
Return value: object **SUrl.**

Object SUrl
Object SUrl is a resource locator (URL) and linked data presentation.

Object properties

Name	**Description**
BSTR link	document's URL
BSTR cp	coding of the document found under this URL
BSTR accessed	last modification data, or last access date. Date format: **dd-mm-yyyy**

This description corresponds to the version mIISInt.dll 1.1.3.29