

Manual

The SeisView Project

Jens Meßinger

30th October 2002



Contents

1	Introduction	3
2	The 'sbase' database	3
2.1	Installing	3
2.1.1	Security information	3
2.2	The database structure	4
2.2.1	Users	13
3	ASP and C#	13
3.1	Installing	13
3.2	SeisBase	14
3.3	The home page	15
4	How to use the seisview webpage	15
4.1	The structure	15
4.1.1	Start page and main menu	15
4.1.2	Station info	15
4.1.3	List pages	16
4.1.4	The config file	16
4.2	The functions	17
5	Known problems and design deficiencies	17
6	ToDo-List	18
7	Recommended literature	18
8	Acknowledgments	19
9	Contacts	19

1 Introduction

'SeisView' is a information system for seismological stations. All data, e.g. equipment and reports, are stored in the database 'sbase'. The user can administrate this database with the help of the 'SeisView' webpage.

The manual gives some information about the structures behind the project and about using 'SeisView'.

2 The 'sbase' database

This chapter describes the database behind the seisview webpage, called 'sbase'. The database server which is used is 'MySQL'. Furthermore you will find some information for getting started with the database.

2.1 Installing

As mentioned above 'SeisView' is using the database server MySQL. So first you have to install MySQL on your webserver for creating your own database.

You can download MySQL from the MySQL home page (<http://www.mysql.com>). Please follow the installation instructions in the MySQL reference manual provided by MySQL. Also see the manual for any further information about how to use MySQL.

After finishing this you can run the 'create_sbase' script (file: create_sbase.txt). The easiest way is to copy this file into the 'bin' directory of MySQL ('./bin/') and start MySQL after that. Now you can use the 'source' command to run the script:

```
mysql> source create\_sbase.txt;
```

The complete structure of the database without any data (see also chapter 2.1.1) will be created. Use the 'insert_data' script (file: insert_data.txt) in the same way to add some example data to the database.

Furthermore you have to install the MySQL ODBC driver. You can also download this from the MySQL home page. After installing you should configure a ODBC data source like given below. Please use Data Source (ODBC) tool for making this (MS Windows control panel — administrative tools — Data Sources (ODBC)).

Datasource data:

```
Windows DSN name      : mySQL-seismic
MySQL host(name or IP) : localhost
MySQL database name   : SBase
User                  : seislog
Password              : seislog
Port                  : 3306
```

Also check the options:

```
Don't optimize ...
Return matching rows
Change BIGINT ...
```

2.1.1 Security information

At installation of MySQL, a system database containing tables for users that have access to MySQL and their privileges are created. This database is called 'mysql'. The only users created at installation are root and a blank username. None of them have a password assigned

to them. Both are super users which have access to the 'mysql' database and have all privileges on all databases in the system. First thing to do after installation is to assign new passwords to these two users.

Each application which want to use a MySQL database should now create new usernames and passwords for access to the database system. They should only be assigned the privileges which are strictly needed and with no GRANT OPTION. In the 'SeisView' system there are three usernames which will be automatically created by running the 'create_sbase' script. These are:

1. dbcon
2. known
3. unknown

Dbcon is not used at moment. **Dbcon** is assigned privileges to INSERT, UPDATE and SELECT on the 'sbase' database and nothing else. **Known** is used by the secure pages in the web application for connecting the database. This user is assigned UPDATE, INSERT, DELETE and SELECT privileges on the 'sbase' database. **Unknown** is used by the insecure pages in the web application for getting information from the database and are only assigned SELECT privileges on the 'sbase' database.

Note: These are the users which have access to the database system (the MySQL database server), but this doesn't mean that these are the users for the 'SeisView' webpage. The real users who are allowed to work with the webpage are stored in the 'authentication' table in the database 'sbase'. There are also some users which are automatically created by running the 'create_sbase' script (and that's why the 'authentication' table contains some data from the beginning). These are:

username	password	used for
dbcon	fnvpm321	not used now
seislog	seislog	superuser
root	lgg31a	superuser
unknown	guest	non-specified user with only read rights

Table 2.1: The automatically created users in the 'sbase' database. This data is stored in the table 'authentication'.

So new users which want to use the 'SeisView' webpage are added to the 'authentication' table. The things they are allowed to do on the webpage depend on the rights they have (see chapter 2.2.1). But these users have no access to the database server. So in reality the users are allowed to use the webpages (and to add or delete data there), but only 'SeisView' itself is allowed to use the database.

2.2 The database structure

The following figures show the structure of the database and the relations between the tables. The meaning of the tables and their field is explained below.

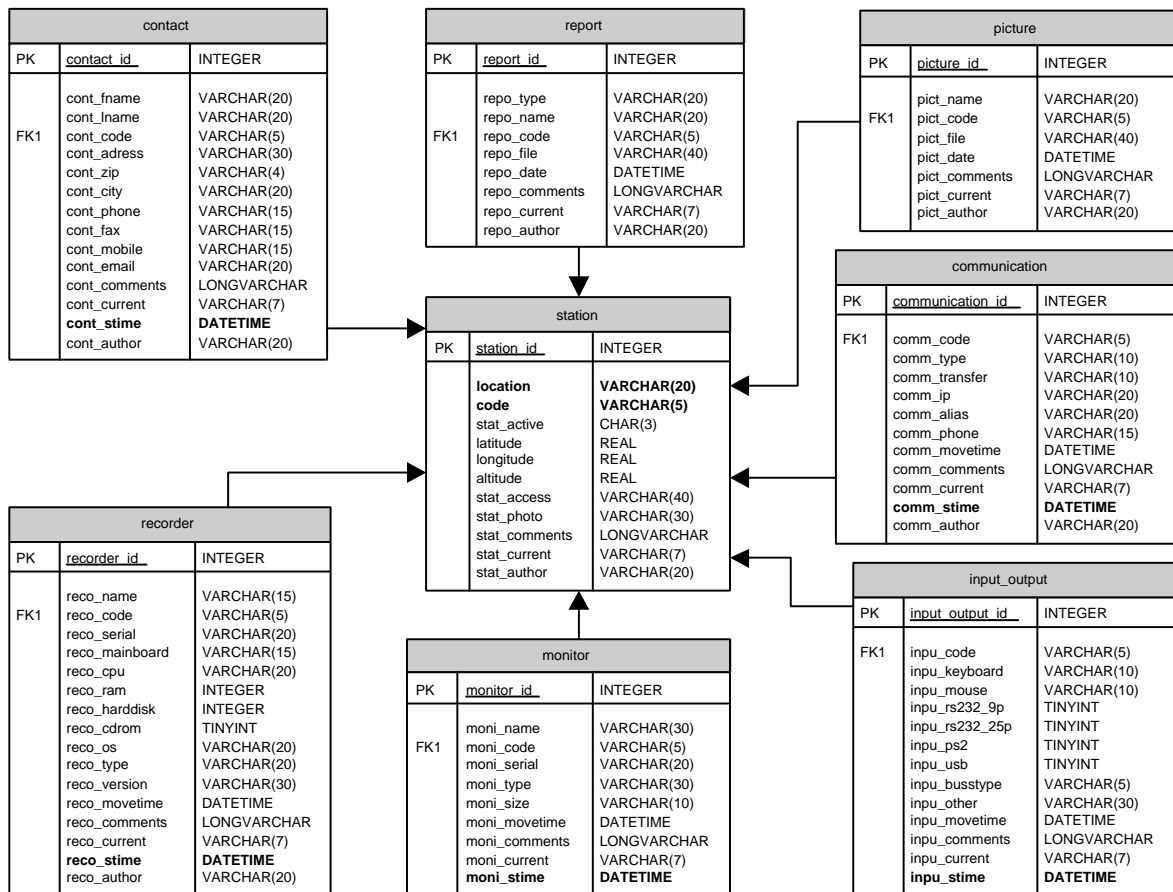


Fig. 2.1: Table 'station' and related contacts, reports, pictures, communication data and recorders

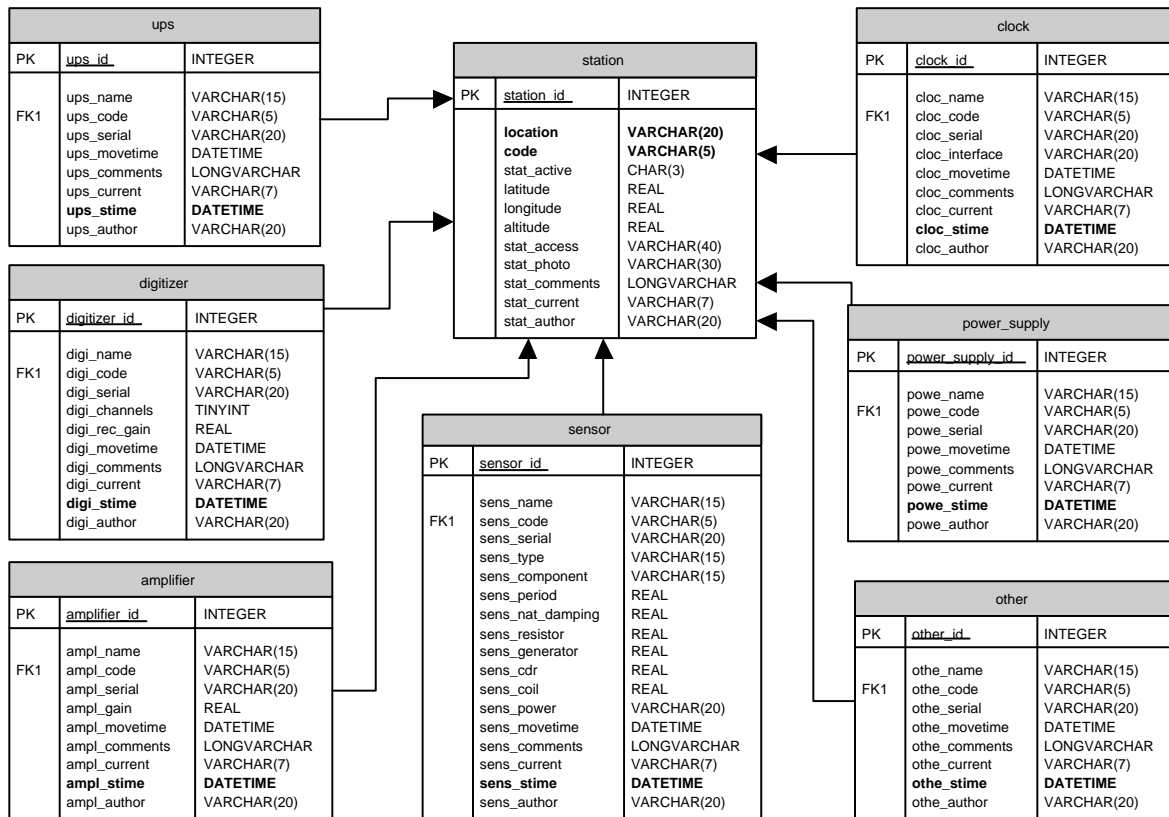


Fig. 2.2: Table 'station' and related equipment (part. I and II)

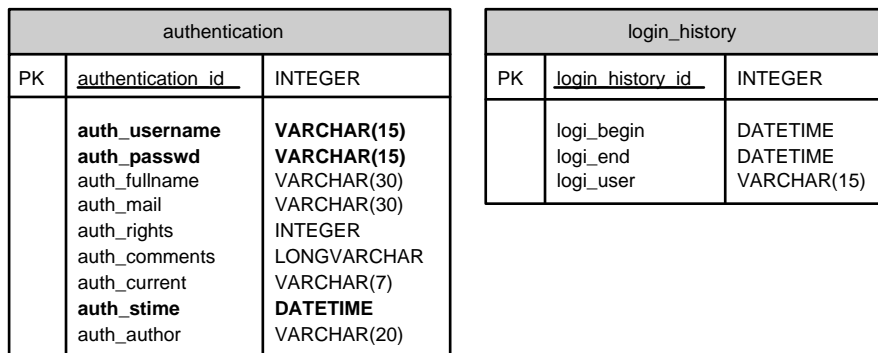


Fig. 2.3: Tables used for user administration

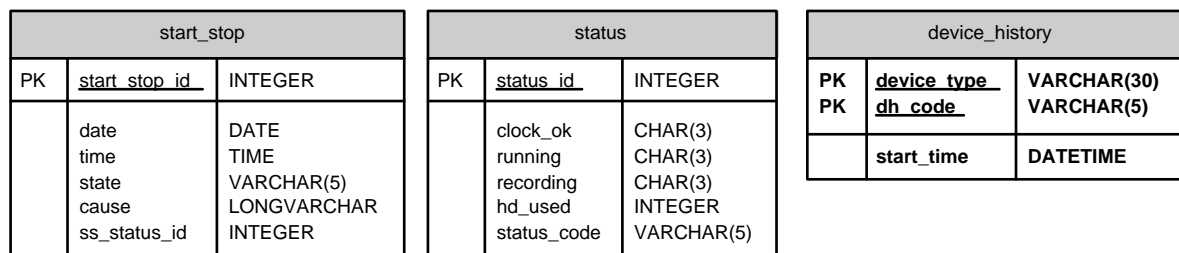


Fig. 2.4: Tables which are currently not used

station_id	int	primary key
location	varchar(20)	location of the station
code	varchar(5)	the station code
stat_active	char(3)	is station active ('YES' or 'NO')
latitude	float	latitude (+N)
longitude	float	longitude (+E)
altitude	float	altitude (meters)
stat_access	varchar(40)	how to access station; link to a file
stat_photo	varchar(30)	filename; contains the photo for the current status of the station
stat_comments	text	field for comments, max 65535 characters
stat_current	varchar(7)	status of this entry: current(YES) or for adding a new entry (ADD)
stat_author	varchar(20)	user, who put the data into the database

Table 2.2: TABLE STATION: contains some general information about a seismic station

contact_id	int	primary key
cont_fname	varchar(20)	operators first name
cont_lname	varchar(20)	operators last name
cont_code	char(5)	references station
cont_adress	varchar(30)	address
cont_zip	char(4)	zip code
cont_city	varchar(20)	city
cont_phone	varchar(15)	phone number
cont_fax	varchar(15)	fax number
cont_mobile	varchar(15)	number of mobile phone
cont_email	varchar(20)	email address
cont_comments	text	field for comments, max 65535 characters
cont_current	varchar(7)	status of this entry: current(YES), not current(NO) or for adding a new entry (ADD)
cont_stime	timestamp	time when the entry was put into the database
cont_author	varchar(20)	user, who put the data into the database

Table 2.3: TABLE CONTACT: information about contact persons (operators) for a station

picture_id	int	primary key
pict_name	varchar(20)	name or description
pict_code	char(5)	reference station
pict_file	varchar(40)	link to the picture file (in the pictures directory)
pict_date	datetime	date, when the picture was added to the database
pict_comments	text	field for comments, max 65535 characters
pict_current	varchar(7)	status of this entry: current(YES) or for adding a new entry (ADD)
pict_author	varchar(20)	user, who put the data into the database

Table 2.4: TABLE PICTURES: pictures which belong to a station (i.g. photos of the station)

report_id	int	primary key
repo_type	varchar(20)	type of the report
repo_name	varchar(20)	name or description
repo_code	char(5)	reference station
repo_file	varchar(40)	link to the report file (in the reports directory)
repo_date	datetime	date, when the report was added to the database
repo_comments	text	field for comments, max 65535 characters
repo_current	varchar(7)	status of this entry: current(YES) or for adding a new entry (ADD)
repo_author	varchar(20)	user, who put the data into the database

Table 2.5: TABLE REPORTS: reports for a station

recorder_id	int	primary key
reco_name	varchar(15)	name or description
reco_code	varchar(5)	reference station
reco_serial	varchar(20)	serial number
reco_mainboard	varchar(15)	mainboard
reco_cpu	varchar(20)	CPU
reco_ram	int	how much RAM (in Mb)
reco_harddisk	int	size of the harddisk (in Mb)
reco_cdrom	smallint	number of installed CD-ROM's
reco_os	varchar(20)	operation system
reco_type	varchar(20)	recorder type (i.e. PC)
reco_version	varchar(30)	used software version
reco_movetime	datetime	time of the last movement
reco_comments	text	field for comments, max 65535 characters
reco_current	varchar(7)	status of this entry: current(YES), not current(NO) or for adding a new entry (ADD)
reco_stime	timestamp	time when the entry was put into the database
reco_author	varchar(20)	user, who put the data into the database

Table 2.6: TABLE RECORDERS: contains information about the recorders at the stations

input_output_id	int	primary key
inpu_code	varchar(5)	reference station
inpu_keyboard	varchar(10)	keyboard
inpu_mouse	varchar(10)	mouse
inpu_rs232_9p	smallint	number of 9 pins serial ports
inpu_rs232_25p	smallint	number of 25 pins serial ports
inpu_ps2	smallint	number of PS2 ports
inpu_usb	smallint	number of USB ports
inpu_busstype	varchar(5)	used busstype
inpu_other	varchar(30)	other ports
inpu_movetime	datetime	time of the last movement
inpu_comments	text	field for comments, max 65535 characters
inpu_current	varchar(7)	status of this entry: current(YES), not current(NO) or for adding a new entry (ADD)
inpu_stime	timestamp	time when the entry was put into the database
inpu_author	varchar(20)	user, who put the data into the database

Table 2.7: TABLE INPUT_OUTPUT: the input/output-devices for the stations

monitor_id	int	primary key
moni_name	varchar(30)	name or description
moni_code	varchar(5)	reference station
moni_serial	varchar(20)	serial number
moni_type	varchar(30)	type of the monitor
moni_size	varchar(10)	the size of the monitor (in inch)
moni_movetime	datetime	time of the last movement
moni_comments	text	field for comments, max 65535 characters
moni_current	varchar(7)	status of this entry: current(YES), not current(NO) or for adding a new entry (ADD)
moni_stime	timestamp	time when the entry was put into the database
moni_author	varchar(20)	user, who put the data into the database

Table 2.8: TABLE MONITOR: the monitors used for the stations (respectively the recorders at the stations)

communication_id	int	primary key
comm_code	varchar(5)	reference station
comm_type	varchar(10)	type
comm_transfer	varchar(10)	transfer type
comm_ip	char(20)	ip address
comm_alias	char(20)	alias for the ip
comm_phone	varchar(15)	phone number for connection
comm_movetime	datetime	time of the last movement
comm_comments	text	field for comments, max 65535 characters
comm_current	varchar(7)	status of this entry: current(YES), not current(NO) or for adding a new entry (ADD)
comm_stime	timestamp	time when the entry was put into the database
comm_author	varchar(20)	user, who put the data into the database

Table 2.9: TABLE COMMUNICATION: contains information about how to communicate with the station

sensor_id	int	primary key
sens_name	varchar(15)	name or description
sens_code	varchar(5)	reference station
sens_serial	varchar(20)	serial number
sens_type	varchar(15)	type of the sensor
sens_component	varchar(15)	component
sens_period	float	free period
sens_nat_damping	float	natural damping
sens_resistor	float	damping resistor
sens_generator	float	generator constant
sens_cdr	float	cdr
sens_coil	float	coil resistance
sens_power	varchar(20)	power (i.e. 12 V or +/- 12 V)
sens_movetime	datetime	time of the last movement
sens_comments	text	field for comments, max 65535 characters
sens_current	varchar(7)	status of this entry: current(YES), not current(NO) or for adding a new entry (ADD)
sens_stime	timestamp	time when the entry was put into the database
sens_author	varchar(20)	user, who put the data into the database

Table 2.10: TABLE SENSOR: contains information about the different sensors in use

digitizer_id	int	primary key
digi_name	varchar(15)	name or description
digi_code	varchar(5)	reference station
digi_serial	varchar(20)	serial number
digi_channels	smallint	number of channels
digi_rec_gain	float	gain
digi_movetime	datetime	time of the last movement
digi_comments	text	field for comments, max 65535 characters
digi_current	varchar(7)	status of this entry: current(YES), not current(NO) or for adding a new entry (ADD)
digi_stime	timestamp	time when the entry was put into the database
digi_author	varchar(20)	user, who put the data into the database

Table 2.11: TABLE DIGITIZER: contains information about the different digitizers in use

amplifier_id	int	primary key
ampl_name	varchar(15)	name or description
ampl_code	varchar(5)	reference station
ampl_serial	varchar(20)	serial number
ampl_gain	float	amplifier gain
ampl_movetime	datetime	time of the last movement
ampl_comments	text	field for comments, max 65535 characters
ampl_current	varchar(7)	status of this entry: current(YES), not current(NO) or for adding a new entry (ADD)
ampl_stime	timestamp	time when the entry was put into the database
ampl_author	varchar(20)	user, who put the data into the database

Table 2.12: TABLE AMPLIFIER: contains information about the different amplifiers in use

clock_id	int	primary key
cloc_name	varchar(15)	name or description
cloc_code	varchar(5)	reference station
cloc_serial	varchar(20)	serial number
cloc_interface	varchar(20)	interface
cloc_movetime	datetime	time of the last movement
cloc_comments	text	field for comments, max 65535 characters
cloc_current	varchar(7)	status of this entry: current(YES), not current(NO) or for adding a new entry (ADD)
cloc_stime	timestamp	time when the entry was put into the database
cloc_author	varchar(20)	user, who put the data into the database

Table 2.13: TABLE CLOCK: contains information about the different clocks in use

ups_id	int	primary key
ups_name	varchar(15)	name or description
ups_code	varchar(5)	reference station
ups_serial	varchar(20)	serial number
ups_movetime	datetime	time of the last movement
ups_comments	text	field for comments, max 65535 characters
ups_current	varchar(7)	status of this entry: current(YES), not current(NO) or for adding a new entry (ADD)
ups_stime	timestamp	time when the entry was put into the database
ups_author	varchar(20)	user, who put the data into the database

Table 2.14: TABLE UPS: contains information about the different ups's in use

power_supply_id	int	primary key
powe_name	varchar(15)	name or description
powe_code	varchar(5)	reference station
powe_serial	varchar(20)	serial number
powe_movetime	datetime	time of the last movement
powe_comments	text	field for comments, max 65535 characters
powe_current	varchar(7)	status of this entry: current(YES), not current(NO) or for adding a new entry (ADD)
powe_stime	timestamp	time when the entry was put into the database
powe_author	varchar(20)	user, who put the data into the database

Table 2.15: TABLE POWER_SUPPLY: contains information about the different power supplies in use

other_id	int	primary key
othe_name	varchar(15)	name or description
othe_code	varchar(5)	reference station
othe_serial	varchar(20)	serial number
othe_movetime	datetime	time of the last movement
othe_comments	text	field for comments, max 65535 characters
othe_current	varchar(7)	status of this entry: current(YES), not current(NO) or for adding a new entry (ADD)
othe_stime	timestamp	time when the entry was put into the database
othe_author	varchar(20)	user, who put the data into the database

Table 2.16: TABLE OTHER: contains information about the 'other equipment' in use

authentication_id	int	primary key
auth_username	varchar(15)	username
auth_passwd	varchar(15)	password
auth_fullname	varchar(30)	full name for the user
auth_mail	varchar(30)	the users mail address
auth_rights	int	the access rights for the user (see chapter 2.2.1)
auth_comments	text	field for comments, max 65535 characters
auth_current	varchar(7)	status of this entry: current(YES), not current(NO) or for adding a new entry (ADD)
auth_stime	timestamp	time when the entry was put into the database

Table 2.17: TABLE AUTHENTICATION: contains information about all the users for the 'SeisView' webpage

login_history_id	int	primary key
logi_begin	datetime	time when the user logged in
logi_end	datetime	time when the user logged out (not used)
logi_user	varchar(15)	name of the user who was logged in

Table 2.18: TABLE LOGIN_HISTORY: the login history for the database

2.2.1 Users

All registered users for the 'SeisView' webpage are stored in the table 'AUTHENTICATION'. It contains fields for the username, the password and some additional information. A very important field for the user administration is the field 'auth_rights'. There are three different levels of access rights which a user can have:

Level	Description	
0	read only	the user is only allowed to read data
1	read/write	the user is allowed to read, edit, add and delete all non-critical data
2	full	read/edit/add/delete access for all data

'Critical data' means the login history and the users passwords. Only a user with 'full' access rights is allowed to see this data. These users are also the only ones which have the possibility to add or delete users.

3 ASP and C#

The 'SeisView' home page was developed with the help of ASP.NET Web Development Tool from Microsoft. There are two parts of this home page. One part consists of the webpages themselves and C# script behind them. The other part is a project called 'SeisBase' which is used as an interface between the webpages and the the 'sbase' database.

3.1 Installing

You can download Microsoft's ASP.NET Web Development Tool from the official home page of ASP.NET (<http://www.asp.net>). You will also find all necessary information about installing and getting started there.

After this you should add two additional parts of the .NET Framework. You can also download this from the ASP.NET home page. Please install the components in the following order:

1. MSDAC 2.7 or later; use default settings
2. ODBC .NET Data Provider

3.2 SeisBase

As mentioned before 'SeisBase' is responsible for the communication between the database and the webpages of the 'SeisView' project. 'SeisBase' contains a class called 'DBWork', which is used for this communication with the help of SQL commands. Within the class, there are definitions of functions for getting data from the database or putting data into the database. The typical structure of such a function is shown here:

```
// function used for getting data from the database
// parameters are:  what - which fields should be selected ...
//                  tables - ... from which tables ...
//                  condition - ... with which condition ...
//                  order - ... ordered how
public DataSet getInformation ( string what, string tables, string
    condition, string order )
{
    // The SQL query is made from the parameters.
    string query = "";
    query = "SELECT "+what+" FROM "+tables;
    if ( condition != "" ) query += " WHERE "+condition;
    if ( order != "" ) query += " ORDER BY "+order;
    query += ";";

    // The result of the SQL query is put into the C$\sharp$
    // class 'DataSet'.
    DataSet ds1 = new DataSet();

    // These are the C$\sharp$ objects for connecting the
    // database and sending the SQL command to it (and
    // receiving the result).
    OdbcConnection myCon = new OdbcConnection( conStr );
    OdbcDataAdapter myAD = new OdbcDataAdapter( query, myCon );

    try
    {
        myCon.Open(); // opens the connection
        myAD.Fill( ds1); // the result of the SQL command is put
                        // into the 'DataSet'
    }
    catch( OdbcException e )
    {
        // Error handling take place here. Message is written in a file.
        StreamWriter
            st1 = new StreamWriter( "C:/Error/DB_error_log.txt", true);
        st1.WriteLine( e.Message.ToString() + "::" + query );
    }
}
```

```
        st1.Close();
    }
    finally
    {
        // closes the connection
        myCon.Close();
    }
    // return value is the 'DataSet'
    return ds1;
}
```

3.3 The home page

Every page of the 'SeisView' web site consists of two files - the page file itself (an asp or aspx file) and the script file behind this page. The language which is used for the scripts is Microsoft's C#.

The structure and language of a asp or aspx file is very similar to a normal html file, only parts of the code are asp specific. So it should be quite easy to understand these files.

The C# files are not so easy to read, but there are a lot of comments.

4 How to use the seisview webpage

This chapter gives you a introduction for using the 'SeisView' home page. It describes the general structure and functions, but it describes not each single page.

4.1 The structure

The home page of the 'SeisView' project consists of three parts. The first part which the user will see contains the start page and the main menu. All pages which show grouped information for one specified station belong to the second part. The third part means all pages where the user can see complete lists of equipment and other data. These are also the pages for the data administration.

Besides the contents of some of the shown titles, for example in all page headers, are not fixed. The information is read from a config file which also will be described here.

4.1.1 Start page and main menu

On the start page you see a list of all active stations in you network. You can select one of the station for getting all information about this station (see chapter 4.1.2) by using one of the hyperlinks. Besides there is a button for jumping to the main menu.

The main menu gives the possibility to go directly to one of the pages for the complete lists (see chapter 4.1.3). So it also shows which kind of data you can administrate with the webpage.

4.1.2 Station info

In these area you will find pages which show all the current information for this station. The information is grouped. You can chose the group of data by using one of the blue buttons. These are the groups:

- **Stationinfo:** the general information about the station, e.g. the location, and the communication data

- **Operator:** contact information for the station's operator
- **Pictures/Photos:** list of pictures and photos for this station
- **Reports:** list of reports; a report can be a comment text, a file or both of them
- **Recorder:** the recorders, monitors and input/output devices for the station
- **Equipment I:** the sensors, digitizers and amplifiers
- **Equipment II:** the clocks, ups's, power supplies and other equipment on this station

You have also the possibility to jump to the complete list pages (chapter 4.1.3) by using the function buttons, which are explained in chapter ??.

4.1.3 List pages

Each list page shows a complete list of one kind of data. Normally this is a list of current data. But you can also use the most of these pages for getting some other information.

1. You can look for all equipment in the special 'stations', called *stock*, *trash* and *other*.
2. It's possible to see the time history for one equipment entry or for current kind of equipment for one station.

For information about how to use this possibilities see chapter ??.

Furthermore the list pages are used for the data administration. So you can add, edit and delete entries to the database.

4.1.4 The config file

As mentioned above some of the shown titles and graphics are not fixed. The data is read from a config file. This file is a xml formatted file and called 'config.xml'. There should be two equal files, one placed in the main directory of the home page and the other one placed in the 'secure' subfolder. So it is easy to change some of the shown information in the home page.

The following contents are stored in this file:

header

- `h.iconfile`: the small picture top-left in the header; relative path to a file
- `h.icontext1`: first line of the icon caption
- `h.icontext2`: second line of the icon caption
- `h.title1`: the big header title
- `h.title2`: the smaller header title
- `h.label`: the black label at the bottom of the header

start page





- `sp.image`: the big image on the first page; relative path to a file

footer







- `f.label`: the black label in the footer

4.2 The functions

Here you will find some short descriptions of the functions on the web site. Most of this functions can be used by click on the corresponding icon. The following functions are available for the pages at the part 'station info'.







	current list	shows the complete list of the current data
	edit this entry	jumps to the corresponding list page and starts the edit mode for the current entry
	add entry	jumps to the corresponding list page and adds an new entry
	time history	jumps to the corresponding list page and shows the time history for this station

And these are the functions for the pages at the 'list pages' part.

	delete entry	deletes this entry
	edit this entry	changes to the edit mode for the current entry
	cancel	finishes the edit mode without saving of the changes
	correct entry	finishes the edit mode and corrects the current item with the new data; this means that there will be no time history for this change
	update entry	finishes the edit mode and updates the current item with the new data; this means that there will be a time history for this change - the old data remains in the database (but is not a current data anymore) and new data is added to the database (and is set as current data)
	time history	shows the time history for this station or for this entry (it depends on which of the clock buttons you use)
<u>Add ...</u>	add entry	adds an new entry and sets this entry into the edit mode

Of course not all of this functions are available for all sites. There is no time history for the tables station, picture and report. That's why you cannot see the time history button for these tables. Besides there is no need for a correction button when you edit a entry in this tables.

Furthermore there are some other functions for using the webpage.

	MENU	jumps to the database main menu
	DATABASE MENU	jumps to the database main menu
	HOME	jumps to the start page
	LOGIN	user can log in
	LOGOUT	logs the current user out
	FILE	dumps the currently shown information to a file and shows this file

5 Known problems and design deficiencies

This is a list of some known problems and design deficiencies which are not solved yet but probably will be solved in the future. It is not a list of real bugs.

- It is impossible to store empty fields or fields only filled with space characters. That's why all empty or space character fields are automatically replaced by '-'. Responsible for the problem is probably the C#-object 'OdbcDataAdapter' which is used for getting the data out of the database.

- There is no check if the data the user put in has the correct type. So in the case of a wrong type an error will occur.
- There are no formatted comments in the complete lists (not the lists which are related to one station). For the pages which show the data for only one station, multiline textboxes are used for the comments (which sometimes may cause a strange page layout).
- The width of some lists is bigger than the page width. In this case the comment fields are quite small.
- You should use the paper orientation 'landscape' for printing pages in the browser you are using.
- Actually one entry in the input/output table is part of a recorder. But there still no relation between these two tables.

6 ToDo-List

The things mentioned here will probably be done in the future. Of course we also try to solve the known problems (see chapter 5).

- In the current database all equipment is linked directly to the station. Crosslinks between different types of equipment (for example digitizer, amplifier and sensor) are the most important improvement of the database in the future.
- Most of the equipment and other data tables have a field for the 'name'. So the user has to put in a name respectively a short description by his own. A predefined list of names which the user should use, e.g. by selecting one item in a dropdown list, would make it easier to administrate the data.
- A new page with a SQL-Interface would be useful. So the user can send SQL commands directly to the database.
- The connection tool 'dbCon' is not implemented.

7 Recommended literature

Because the online documentation of ASP and C# from Microsoft is quite bad you should look for some good books or independent online tutorials. The MySQL tutorial is much better, but there is only a short introduction to SQL itself and almost no information on how to create a database with an effective structure. That's why you find here a small list of some recommended literature.

- MySQL: MySQL manual (<http://www.mysql.com>)
- SQL: SQL tutorial (<http://w3.one.net/~jhoffman/sqltut.htm>)
- ASP.NET Web: official home page of Microsoft ASP.NET (<http://www.asp.net>)
- Deitel, H., M. et al (2002): C# - How to program, Prentice Hall
- MacDonald, M.: ASP.NET (2002): The Complete Reference, McGraw-Hill/Osborne
- Elmasri, R., Navathe, S., B. (2000): Fundamentals of database systems, 3rd ed., Addison-Wesley

8 Acknowledgments

The current 'SeisView' project is based on a project made in the spring 2002 by Jon Terje Tjønn and Tore Senneseth. Despite almost all web sites are completely new created for the new project. However the idea behind the structure of the project, the user interface and some of the functions are still the same. But of course the most functions are new created, too.

The new project was made by Jens Meßinger. He studies geophysics at the university of mining and technology in Freiberg/Germany (German name: TU Bergakademie Freiberg). During his IAESTE traineeship he worked at the Institute of Solid Earth Physics (University of Bergen) for 2 months.

Especially the ideas behind the project and the user interface were created and improved with the help of Jens Havskov and José Åsheim Ojeda, both working at the Institute of Solid Earth Physics.

9 Contacts

For questions about the National Norwegian Seismic Network and the Seisview project in general you should contact Professor Jens Havskov.

Jens Havskov
Institute of Solid Earth Physics
University of Bergen
Allégaten 41, 5007 Bergen
Norway

Phone: +47 5558 3414
Email: jens@ifjf.uib.no

Please contact José Åsheim Ojeda for detailed questions about the source code of the webpage or the database:

José Åsheim Ojeda
Institute of Solid Earth Physics
University of Bergen
Allégaten 41, 5007 Bergen
Norway

Phone: +47 5558 8780
Email:jose.ojeda@ifjf.uib.no