UNIVERSITÄT OSNABRÜCK

Institute of Environmental Systems Research

GREAT-ER 1.0
Technical Manual

Jan-Oliver Wagner · Frank Koormann
http://www.usf.Uni-Osnabrueck.DE/projects/GREAT-ER/

January 25, 1999

Contents

List of Tables

List of Figures

1 Introduction

This document is meant to support maintenance, modifications, extensions and other development based on the software GREAT-ER.

This chapter gives some introductive information and should always be read. The next chapter will discuss software requirements that should be reviewed when installing the GREAT-ER development filetree. Chapter three is essential for adding furthers catchments or data elements to existing catchments. Chapter four helps to understand the actual end-user system and should be considered when planning further feature in GREAT-ER (e.g. an additional analysis routine). The last chapter offers additional information about the Avenue scripts and their interconnection and should be used for reference purposes when planning major modifications on the end-user system.

1.1 General information

GREAT-ER has been developed by a group of several european organizations in 1996 to 1999. The actual software part has been basically developed at the Institute of Environmental Systems Research of the University of Osnabrück, Germany. One exception is the simulation software which has been entirely developed at the University of Gent, Belgium. Hence, this document will not cover any description of the simulation software.

Besides the software, data (especially geographic data) are an essential part of GREAT-ER. Some parts of the data processing have been done at the Institute of Environmental Systems Research and information on the processing are a part of this document. Initial data collection and preparation (covering hydrological processing) have been done by the Institute of Hydrology, Wallingford, UK and are not part of this document.

GREAT-ER development (allthough a Windows NT software) has been performed basically on Unix workstations. It may be possible to transfer the whole filetree to a Windows platform, but the time for adapting to correponding Windows-typical features may be high and the results not as efficient (and simple) as on Unix platforms. One reason for this is an extensive use of makefiles and shell-scripts. Hence it is recommended to keep the whole development method on the current platform (some sub-parts may still quite easily be transfered to other platforms).

1.2 How to use this document

This document does not claim to be complete or fully correct in every detail. The reason for this is on the one hand the size of the project and on the other hand the permanent improvement of sub-parts. The very little time available for the authors to write this document is another reason.

However, the most accurate documentation of processing routines are the actual source codes, provided all key items are well described within the source codes. Hence, this document tries to collect overviews to provide information that lead to the source files to be modified/considered for a certain change. Furthermore, the basic understanding of data flows and the semantic and actual interconnection of the sub-parts is supported.

All this means that the reader of this document must be familiar with several aspects of software development in priciple as well as able to work on (or efficiently understand) a number of different software tools. The document should only be referred to when an actual task is to be solved.

1.3 Other information and documents

As mentioned above this document forms only a part of the descriptions to understand, handle and extend/modify GREAT-ER. The following information are recommended as additional information resources:

GREAT-ER User's Manual If you are new to GREAT-ER you should read this document for for a basic understanding of the software. Of course, you should then also extensively work with the program to get really familiar with it.

Technical description of the simulation software Please refer to this document to learn about the the implementation of the simulator. Please contact ECETOC to obtain this technical manual.

ARCVIEW Manuals For a basic understanding of ARCVIEW and its built-in language Avenue you should read the manuals as well as refer to the online-documentation.

ARC/INFO manuals For basic and deepened understanding of ARC/INFO and geographic operations in general.

GREAT-ER development filetree Read the source files for in-depth understanding of actual operations. All key operations in the source codes are provided with comments. All source files are handled with the Revision Control System (RCS). Hence, all changes (and the associated comments) can be traced back.

1.4 Contact

The main authors of GREAT-ER version 1.0 are Frank Koormann and Jan-Oliver Wagner. Some small parts have been developed by Carsten Schulze and Martin Hinsch. In every RCS-managed script you will find hints on the author(s). You can contact all authors via the email address

`greatdev@usf.Uni-Osnabrueck.DE`

This email address is not a user support hotline. Those who plan to modify/extend GREAT-ER may ask questions when they rush into major problems.

You may also refer to the GREAT-ER Web pages that were set up during the project development:

`http://www.usf.Uni-Osnabrueck.DE/projects/GREAT-ER/`

2 Software Requirements

The development of GREAT-ER at the Institute of Environmental Systems Research has been done on Workstations running under SunOS 5 (= Solaris 2). The underlying operating system is likely to be of no major importance, since the software dependencies are based on the applied software packages and tools of which most are available for several platforms.

Before starting to work on the source codes (i.e. executing makefiles and other scripts), please make sure that the following software is available. The versions which were used at the Institute of Environmental Systems Research are listed, possible other (higher or even lower) versions will work, too.

Due to the complexity of the system and the diversity of the used tools it is likely that the following list is incomplete and that some information is not sufficiently detailed. Please report any problems to the authors.

Several tools are licenced with the General Public Licence (GPL). Please refer to
`http://www.gnu.org/copyleft/gpl.html`
for the full licence text.

arc **Version:** ARC Version 7.1.1 (Thu Feb 6 23:26:50 PST 1997)
Licence: commercial
Description: A full GIS system to be purchased from ESRI or an official ESRI dealer.
In GREAT-ER: Various GIS methods are used during the generation of the catchment coverages. The produced coverages can be used directly by ARCVIEW.

awk **Version:** GNU Awk 3.0.3
Licence: GPL
Description: Awk is a pattern scanning and processing language.
In GREAT-ER: it is often used for format conversions and some spreadsheet-like data processing in several processing routines. It is also required for the end-user system where simulation results produced by the simulation software are processed before read into the ARCVIEW system.

cut **Version:** cut - GNU textutils 1.11
Licence: GPL
Description: Cuts columns from ASCII tables.
In GREAT-ER: This tool can easily be used via makefiles and is needed for generating catchment river data for the simulator.

echo **Version:** echo - GNU sh-utils 1.12
Licence: GPL
Description: The GNU version of the well known `echo` command.
In GREAT-ER: this version is needed in several makefiles.

join **Version:** join - GNU textutils 1.11
Licence: GPL
Description: Joins tabular textfiles. This tool can easily be used via makefiles.
In GREAT-ER: it is used to support the generation of the catchment river data for the simulator.

latex2e **Version:** LaTeX2e <1995/12/01> patch level 1
Licence: free software
Description: Text (type-setting) processor.
In GREAT-ER: used for several documents including the user's and the technical manual. Some packages such as `a4.sty` and `epsf.sty` were also used.

make **Version:** GNU Make version 3.77
Licence: GPL
Description: This make tool is needed to process the makefiles.
In GREAT-ER: it is required for several processing and installation routines. Most makefiles should also be processed correctly by other make implementations, but some need the special features of the GNU make.

sed **Version:** GNU sed version 2.05
Licence: GPL
Description: Sed is a powerful stream editor.
In GREAT-ER: it is used in some makefiles for simple format conversions (find-replace operations).

Tcl/Tk **Version:** Tcl 8.0, Tk 4.2
Licence: free software
Description: Powerful macro-language.
In GREAT-ER: Used for several prototyping purposes. Part of the end-user-system: Tcl/Tk is used to properly execute some system commands.

zip and unzip **Version:** Zip, version 2.1 ; UnZip, version 5.31
Licence: free software
Description: Tools for archiving files. These are the versions of the 'Info-Zip' project, see

`http://sunsite.cnlab-switch.ch/ftp/mirror/infozip/`
In GREAT-ER: The two commands are required for the end-user system. They are used for the Export/Import functions.

Several other (small or even tiny) tools such as `sort, tail, cp, rm, mkdir` and `cat` are also used in makefiles for data processing and installation routines. These tools/OS-commands should be available on any UNIX system. No special implementations are needed (these should all work the same).

3 Preparation of Catchment Data Sets

This chapter describes how to prepare and process catchment data to create a new catchment data set for GREAT-ER. This includes a description of the file's location as well as a description and examples for the data preparation and final processing.

The information in this document are not complete. The most detailed descriptions can only be found in the script files (placed as comments). Furthermore, some of the presented examples and processing descriptions may be substituted by other adequate methods, provided one is aware of the complex interconnection of the processing routines. Major changes should only be done if you feel familiar with the whole complex.

3.1 Introduction

For such complex and comprehensive data as required to build a GREAT-ER catchment data set, it is to be expected that the obtainable data are in several different formats and/or resolution. This problem is managed by the definition of an intermediate file format. For each data group a GREAT-ER pre-defined format is given. The demanded files have to be produced from the original raw data manually or with special routines which likely have to be developed anew.

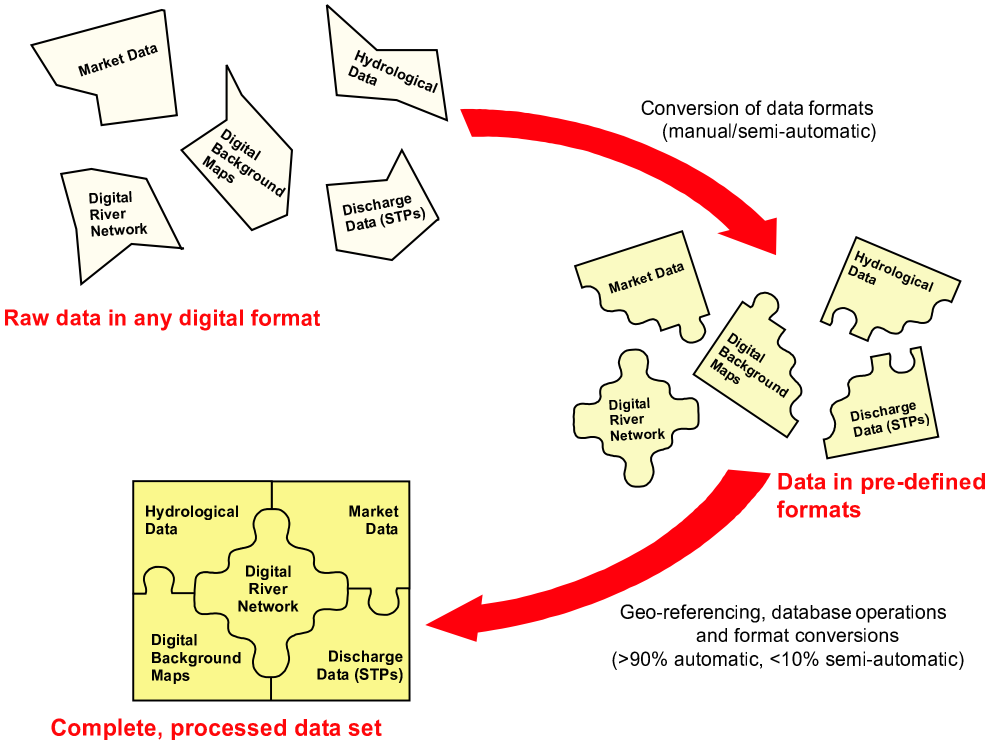


Figure 3.1: *Two steps of data processing*

Once you have converted all the required data, the second step is quite simple. Most of the work is done automatically via makefiles and several scripts. These will direct format coversions, data joining, etc. and finally the installation of the new catchment.

Performing these steps (especially step 1) requires some knowledge about software engeneering basics and of course about the usage of the applied software.

Please always feel free to examine the existing routines (e.g. for MicroLowFlows datasets) to learn from them. The processing of the raw data can be very individual and hence can not be described as detailed as it should be for an easy recipe.

For adding a new catchment, you have to perform the following steps:

1. `mkdir ~/src/data/raw/yourcatchment`
2. Copy your raw data to that directory and perform all processing in that directory (or subdirectories) to end up with the files in the pre-defined formats. If you are automizing parts of your processing, feel free to add e.g. awk-scripts in the directory `~/src/data/raw/awk` and makefiles in `~/src/data/raw/mak/`. This does make sense when applying your routines to more than one catchment.

   Your catchment directory must at least contain a makefile with an empty target (e.g. 'all:'). This is required for the automatic installation being the first step of the final data processing.

   When copying raw files to your catchment directory avoid the names 'yourcatchment.riv', 'yourcatchment.bnd', 'yourcatchment.stp', 'yourcatchment.bgd', and 'yourcatchment.att'). These will be the files you have to prepare in the pre-defined formats.

   Discharge data (when used in more than one catchment) can also be placed in the directory `~/src/data/raw/discharges`.
3. `mkdir ~/src/data/catchments/yourcatchment`
4. Create the file `~/src/data/catchments/yourcatchment/makefile`:

   ```
   # the name of this catchment:
   CATCHMENT = yourcatchment

   # the country of this catchment
   COUNTRY = yourcountry

   # choose extension for data source file
   # (empty = default extensions)
   SRCDAT_TYPE =

   # undefined variable PICTURES means NO
   #PICTURES = YES

   include ../mak/makefile.final.inc
   ```

   The country for your catchment is to be specified in a two-letter code. (at the current state the following codes are used: `be` for Belgium, `de` for Germany, `it` for Italy, `uk` for the United Kingdom).
5. Add the new catchment identification to the catchment data base

   `~/src/data/catchments/catchmentsDB/catchments.txt`

   and install the database (`make install`).

6. `mkdir ~/src/data/overview/yourcountry/yourcatchment`
7. `mkdir ~/src/data/raw/wq/yourcountry/yourcatchment`
8. In that directory create the file `yourcatchment.wq` containing only one line:
   `#Name,Mapname,Type,Legendfile,ctch_fl`

3.2 Establishing the pre-defined format

3.2.1 The digital river network

The digital river network describes the structure of the river network of interest. It consists of lines (or so-called 'polylines') of which each is representing a river segment and nodes which represent the point of connection of two segments. Lines are expressed as a sequence of coordinate pairs, nodes are expressed as exactly one coordinate pair.

Lines have to be considered as vectors, the first point of the sequence is the starting point and the last the end-point. Thats how the segments contain the information about the flow direction. The length of a segment can be computed from the coordinates.

Each transition from one segment to a subsequent one (sequence, confluence or bifurcation) is defined by a node which contains the information 'from' and 'to'. Those are the ID's of the upstream ('from') and the downstream ('to') segments. The geographic position of a node is uniquely given with the last point of the upstream and the first point of the downstream segment. Are these two points not identical, the two strechtes are not connected and no corresponding node exists.

Principles on creating a digital river network The most important fundamentals for the creation of a digital river network are maps of the corresponding area. For this, traditional printed maps can be used, but nowadays usually those maps already exist in a digital format.

Pictures taken from air planes or satellites offer information in a raster format. That means, surface waters consist of single points (squares) and hence do not contain information about e.g. flow direction, but on the other hand may offer information about aereal aspects (e.g. width of rivers, shape and size of lakes) and possibly further properties (temperature, depth). The raster information must be converted to vector data for gaining a digital river network as described above. Such a conversion can partially be automized, manual rework by a human expert is unavoidable.

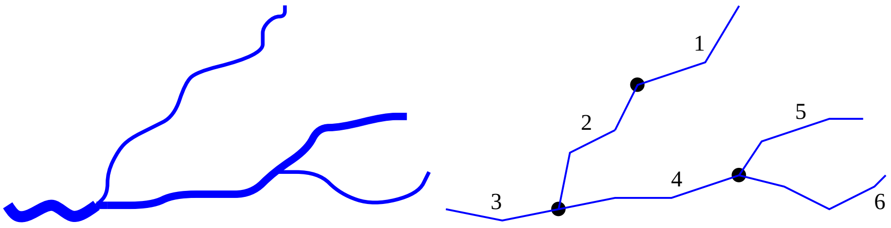


Figure 3.2: *Real river course and digitized river with vectors and nodes*

When the vectorization has been finished or an already prepared dataset was available, the river network is now present in any format, dependend on the origin of the data and/or the used GIS.

Usually the data have then still to be tested for the following demands and may have to to be adapted to fit the demands:

- All river segments must be directed downstream.
- Should two segments be connected, it is necessary that the end-point of the upstream and the start-point of the downstream segment are identical. That also means that for confluences or bifurcations always the upper segment(s) end at that point and one (or two) new segments start from that point.
- All segments must have a unique ID.

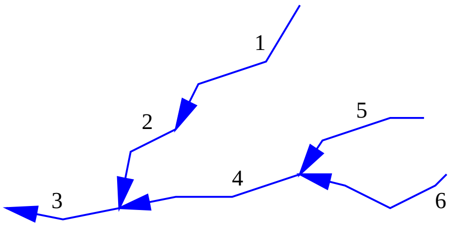


Figure 3.3: *Downstream directed segments*

For the segmentation of the river stretches it should also be considered that each segment represents a homogenous item. Hence, if it is planned to provide a long linear segment with different attributes (e.g. flow or flow velocity) it is already now necessary to split up the long segment into a number of smaller segments (building a sequence).

GREAT-ER pre-defined format '.riv' In the following the file format for the data of a digital river network is described. Those are so-called 'ASCII' or 'text' files and should use the character set ISO 8859-1 (ISO latin 1).

The format is line-oriented; each line starting with a '#' (comment line) and all empty lines have no meaning and can be added anywhere.

This counts also for all other described formats below ('.att', '.bnd', '.stp' and '.bgd').

Each river segment is introduced with its ID `StretchID`. All following line-wise comma-separated coordinates define this segment until a new ID appears. The first coordinate pair is the start, the last coordinate pair the end of the segment. The order of the segments is a matter not to be cared for. The order of the coordinates is of course of high importance.

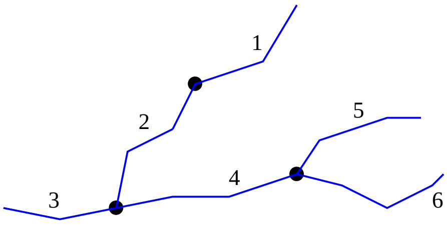


Figure 3.4: *Geographic data*

1
5.9,1.9
5.4,1.3
4.96,1.1
4.5,0.8
2
4.5,0.8
[...]

and the source code

All coordinates must be given in the same projection which has to be ’geographic projection’ with the unit ’dd’ (decimal degrees). As the decimal point, a point and not a comma must be used. Numbers are not allowed to contain any other delimiters. Exponential numbers must be expressed using the letter ’E’ or ’e’.

These demands for coordinates are also required for the below described formats ’.bnd’ and ’.stp’.

3.2.2 The river network attributes

Depending on the demands by the applied exposure simulation models, information about certain properties of the river network are required. Usually these are floating point numbers (e.g. flow in $[m^3/s]$ or flow velocity $[m/s]$). Other attributes such as boolean (shipable [yes|no]) and others are conceivable. Most of the traditionally measured data are expressed as statistics due to natural fluctuations and seasonal rythmic. For example, the flows are usually expressed as log-normal distribution provided with the mean and standard deviation (or mean and low percentile (e.g. 5th)).

However, all attributes have in common that they are corresponding with a certain river segment. A parameter for a river network can hence be given in tabular format.

Necessary attributes for GREAT-ER The in the following listed attributes represent only a small part of possible river parameters. It will be those that are required or helpful for the GREAT-ER prototype.

Some of the attributes are obligatory, some can be specified optionally. The latter ones are typically those which are difficult to obtain and estimation routines are available to compute the optional values from the obligatory ones.

Flow The most important river attribut for GREAT-ER is the flow. Hence, it is obligatory for all river segments. Of course the flows are not constant values, because they are underlying fluctuations which are basically driven by the annual seasons.

This makes it necessary to handle flows as statistical parameters. For annual statistics the distribution type is usually log-normal. This distribution is defined by the mean and the standard deviation. The standard deviation is often replaced by the 5th percentile. Both values have to be given in $[m^3/s]$.

In hydrology for flows often the low flow scenarios are of interest. Hence, the statistics are given with the mean and the 95th low flow percentile. In the distribution curve the 95th low flow percentile is actually the 5th percentile. To avoid any further confusion, in this document always the mathematical correct term (i.e. 5th percentile) will be used.

Flow velocity The flow velocity is the most important parameter for the main process of chemical fate in rivers: advection. Since methods are available for estimating the flow velocity from the flow, it is an optional attribute. Analogous to to the flow, the flow velocity is also a statistical attribute (assumed as log-normal distribution and given by the mean and 5th percentile, both in $[m/s]$).

Missing values are to be estimated with this estimation formular:

$$v = 10^{-0.583} MF^{0.283} (\frac{Q}{MF})^{0.495}$$

(refer to GREAT-ER Working Note No. 5, IH, Wallingford, UK).

Depth The depth does influence the chemical elimination process of 'volatilization'. Of course the depth is correlated with the flow. Hence, it is possible to calculate the (optionally given) depth where no other information were available. Again, a log-normal distribution is assumed. Mean and 5th percentile have to be given in $[m]$.

Missing values are estimated applying this equation:

$$r = 0.57 \cdot e^{0.36 \cdot log(Q50)}$$

$$d = (r > 2.13?0.610 + 0.93 * r : 1.21 * r)$$

(Regime Theory (Simons & Albertson, 1960) Simons DB & Albertson NL (1960) Uniform water conveyance channels in alluvial materials. ASCE, Vol 86, No. HY5, 1960.)

Length Theoretically there is of course no problem to calculate the length of each segment based on the coordinates. These values are naturally (sometimes much) lower than the real length of the river segments. These differences are clear when taking into account that even detailed digitizing (e.g. one point every 300m) can not reflect the actual shape of a very curvy river. Additionally, the digitizing itself reveals uncertainties.

Together with the flow velocity the length basically determines the transport of a (chemical) load. The travel distance should hence be given as accurate as possible. Obtaining the actual length of rivers should generally be not a big problem, because it is relatively easy to measure. Sometimes canoeing books offer these information in a high quality. All values have to be given in $[m]$. This attribute is obligatory.

Name The names of rivers are provided for the user's orientation only. They are optional and have to be given as character strings.

GREAT-ER pre-defined format '.att' For the '.att' format has to be applied the same technical framework as for the '.riv' format. The attributes can be stored in a very simple tabular structure. Column separator is the comma. The columns have to be given in an order, the following example will illustrate:

```
# Aire/Calder Catchment, Yorkshire
# StretchID,Qmean(m^3/s),Q5(m^3/s),vmean(m/s),v5(m/s),
# ... RealLength(m),depthmean(m),depth5(m),Name
26954,34.751000,12.280656,,,2491.655000,,,Unnamed
22954,0.009000,0.002359,,,1648.222000,,,Unnamed
7333,34.739000,12.276415,,,1629.885000,,,Unnamed
22932,0.005000,0.001088,,,540.190000,,,Unnamed
26950,34.731000,12.274977,,,2336.740000,,,Unnamed
22923,0.017000,0.002167,,,2447.520000,,,Unnamed
...
```

In this example no information were available for the flow velocity and the depth and the corresponding columns were left empty.

3.2.3 The catchment boundary

The catchment of a river network is defined by the most downstream point, i.e. the most downstream segment. The catchment boundary represents the natural drainage area. Any point in that area will drain to that most downstream segment.

GREAT-ER additionally offers to join several separated river networks. This does make sense for flat coastal regions or for complete islands or even whole countries. In such cases the catchment should be the union of all the single sub-catchments.

The information about the actual border of a catchment is used in GREAT-ER in two cases. First as an optional opportunity to select discharge sites for a certain region out of large database using the site coordinates. This method reveals some uncertainties, especially when the sites are close to the borderline (e.g. sometimes the effluent is pumped against the natural slope).

Furthermore, the boundaries are displayed in the GREAT-ER end-user system for visual orientation and interpretation support.

A catchment boundary must always be a closed polygon.

GREAT-ER pre-defined format '.bnd' For the '.bnd' format has to be applied the same technical framework as for the '.riv' format. The polygon is to be described linewise by the single coordinate pairs (x,y). The following example should be sufficient to illustrate the format:

```
-2.052912,53.683811
-2.060483,53.683807
-2.060477,53.679314
-2.068047,53.679310
...
-2.045346,53.688309
-2.045342,53.683804
-2.052912,53.683811
```

This example shows also that the polygon must be closed, i.e. the last and the first point must be identical.

3.2.4 The background data collection

To support a quick and easy access to valuable geographic background information a list of background data sets can be defined for each catchment.

In the '.bgd' file the names of additional background data that shall be loaded by the menu entry DISPLAY/Add Background Data must be specified. Basically, the data can be given in two ways. In a first simple case a background data set is only a geo-data set ARCVIEW can load as a theme, i.e. an ARC/INFO Coverage, an image, a grid or a shapefile This case must be indicated by the value 'yes' in the field 'ctch_flg' (see below). In the more complex case a background data set consists of a set of geo-data, indicated by the value 'no' in the field 'ctch_flg', thus leading to a set of themes. In this case, not the name of the data set must be specified, but an ID for a script that knows how to load the geo-data sets.

Name Name of background data to be shown. This is either a dataset, that ARCVIEW can load as a theme or a more complex data set of themes that needs to be loaded by a script. The first case is indicated by the value ’yes’ of the field ’ctch_flg’ (see below), the latter by the value ’no’.

Mapname In the case the background data can be loaded as a single theme this is the name of the coverage, image or shapefile that contains the data. Otherwise this field contains the script which can load the backgrounddata set.

Type In the case the background data can be loaded as a single theme this is the feature type of the data set. Valid entries are point, line, polygon, shape, image and grid.

Legendfile In the case the background data can be loaded as a single theme this is the name of an optional ARCVIEW legend file. Otherwise this field can contain (optional) parameter the script being given in the field ’Mapname” may need.

ctch_flg This field has one of the values ’yes’ or ’no’. ’yes’ indicates that this background data set consists of one theme that does not need a special script to load into the active view, ’no’ means a special Avenue script has been developed by the user which runs under GREAT-ER and loads the backgroud data set into the active view.

There is one special case to be considered. Due to the circumstance that the DCW data (Digital Chart of the World) by ESRI are usable in the GREAT-ER project for the whole of Europe, special scripts loading the DCW data were developed. So if the DCW data shall be a background data set for a catchment, the following line needs to be inserted into the file ’yourcatchment.bgd’ to automatically let the DCW data be a background data set:

```
DCW data,,,,
```

In the case a user installs the DCW data for not yet included countries, e.g. France, Spain or Denmark, you have to install them in the structure they are installed for Italy, Germany, Belgium and the United Kingdom.

GREAT-ER pre-defined format ’.bgd’ As an example file the one for River Itter is given:

```
#River Itter background data
#Name,Mapname,Type,Legendfile,ctch_fl
Cities,itter\cities,polygon,itter\cities.avl,yes
Tk25,itter\tk25tiffz.tif,image,,yes
Rhein,itter\rhein,polygon,itter\rhein.avl,yes
DCW data,,,,no
```

3.2.5 The discharge data

The discharge data are representing information about points through which substances are released into the river network, e.g. sewage treatment plants. In some cases it might make sense or even be necessary to aggregate several discharge sites into one (fictive) discharge point.

Please note that the discharges discussed here are point emmissions. Diffuse emmission will not be considered. Theoretically it is possible to represent a diffuse input in a discrete way, but for this a very high resolution would be required. In this case it does make more sense to adapt GREAT-ER to such situations (at the simulation level).

Coordinates (x,y) The coordinates of a discharge site are obligatory. Are these unknown, they should be available from detailed maps. In the case of real existing plants, the actual position of the plant and not the point of emission into the river should be taken. For a later interpretation of the simulation results and the comparison with monitoring the information whether a plant is on the left or right bank, can be quite valuable. The connection to the river network will be established with the additional specification of the corresponding segment (see below).

Connected population (Pop) For the estimation of the load of a chemical that will enter the discharge point, sometimes a per-capita consumption for a region forms the base. 'Per-capita' in such cases refer to the real population and not to population equivalents.

Hence, for industrial discharges this value must be omitted or set to 0.

Population equivalents (PopEquiv) The population equivalents are not required for a simulation, but may help a user for interpreting the results. This item is optional.

Dry weather flow (DWF) The dry weather flow describes the amount of water that passes the discharge site. As the name already indicates, the value for dry weather should be specified, the influence of precipitation is hence not to be included. This attribute must be given in $[m^3/d]$ and is optional when the actual daily flow has been specified.

Nondomestic Flow This value specifies the industrial flow in $[m^3/d]$. For pure domestic discharges its value should be 0.

Runoff Flow This value specifies the flow due to precipitation and subsequent run-off in $[m^3/d]$. If no further information are available, set this value to 0.

Actual Daily Flow (ADF) The actual flow of a discharge describes the amount of water that actually is released into the river network via the corrsponding discharge site. This attribute must be given in $[m^3/d]$ and is optional when the dry weather flow has been specified.

Type of discharge (Type) The type of a discharge site will be used in the simulations to determine the elimination efficiency. For some typical treatment plants special simulation models can be executed to calculate the removal of a chemical.

The type is to be given as a character string. Currently only few types are supported, but new types should be invented for other discharge types.

Table 3.1: *Discharge Types*

Type-Code	Description
AS	Sewage treatment plant with a biological treatment (Activated Sludge)
TF	(Trickling Filter) plant

Some treatment plants are a combination of both technologies. In such cases it is recommended to stay on the conservative side and select the type of less efficiency, but still mention the actual type in the name of the discharge site (see below).

Connected river segment (StretchID) The discharge is connected to the first point of the given river segment. Hence, it does make sense to consider discharge sites already when creating the segmentation of the river network, i.e. add new segments (divide segments) for an optimal representation of the actual emission points.

The ID of the river segment is obligatory.

Name The names of discharge sites should help the user in orientation and understanding. Hence, the names should be short, but clearly and contain further important hints where necessary.

GREAT-ER pre-defined format '.stp' For the '.stp' format has to be applied the same technical framework as for the '.riv' format. Each line represents one discharge. The first column is a unique ID for the discharge, followed by the coordinate pair and the attributes. The order of the columns is shown in this example:

```
# ID,X,Y,Pop,DWF(m^3/d),Flow(m^3/d),Type,StretchID,Name
41,-1.390494,53.422081,422677,130000,180000,AS,26842,Blackburn Meadows STW
42,-1.349155,53.369724,95750,21800,24506,AS,26843,Woodhouse Mill STW (Sheffield)
43,-1.326922,53.445114,107203,31250,32410,AS,26841,Aldwarke STW
```

3.3 Examples for preparing raw data

The preparation of the above described data formats and fulfilment of the demands on internal logic can mean much work when performed manually. Hence, it should be the goal to process data which is stored in another structure automatically (or at least semi-automatically) whenever possible to produce the format of the five demanded files. This is especially important if larger (or many) data sets in the same format are to be processed, because a (semi-)automatic routine can be applied multiple times.

But also a single conversion could be made more efficient by applying small software tools and script routines.

The structure in which geographic data are stored are usually quite similar. The conversion into the required formats consists basically of format conversions and unit conversions.

In the following two examples are introduced. The main difference in the original raw data is the representation of the river network. In the first example it consits of single segments similar to the required structure of GREAT-ER. Only segments can be refererred to by their ID's, it is not possible to refer to any point in the river network.

In the second example each river is one line (segment, route). To this line events (e.g. confluence, emission) are associated at certain locations. Any point in the river network can be referred to by the river's ID and a distance (e.g. River Rhine, km 240).

Processing of discharge data The discharge data may require a separate treatment. While the data sets for river networks represent a complete enclosed unit, discharge data may be used in several ways. It does make sense to collect all treatment plants of a region and to select some sites for a certain catchment in that region.

The following scheme illustrates only one possibility on how the raw data processing of discharge data can be performed:

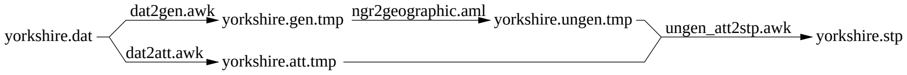


Figure 3.5: *Example for processing sequence for discharge data*

(see `˜/src/data/raw/discharges/makefile`)

Basically, the coordinates are converted to geographic projection and the attributes are re-arranged and in some cases the units are re-calculated.

3.3.1 MicroLowFlows

General information about MicroLowFlows MicroLowFlows is a software for computing flow statistics at ungauged sites. For this, data from gauging stations, precipitation, information about soil types, elevation data for catchment size estimation and artificial influences (abstractions, discharges) are considered. From this geo-referenced water balance the flow statistics for all ungauged river segments are computed.

MicroLowFlows was developed at the Institute of Hydrology (IH) in Wallingford, UK. In the framework of GREAT-ER this tool has been extended by a special consideration of artifical influences. The program runs under MS-DOS.

Except detailed sewage treatment data, the database of MicroLowFlows contains all information about the catchment which are required by GREAT-ER.

The IH has provided small extraction tools for easy retrieval of the desired data. These tools produce simple text files which have then to be further processed to fit the demands of GREAT-ER.

MicroLowFlows export files The extraction tools create simple text files which offer the data in tabular structure. Some demands are already fulfilled in these files. A re-formatting and a mathematical and geographical rework is still necessary. In the following the four extracted text files are described.

An extraction process must be given a start segment. This uniquely determines the catchment.

Coordinates and Attributes This file contains the central data set with the river network. All river segments are included with their coordinates. In MicroLowFlows a segment has a maximum of 9 coordinates. Only due to this fact it was possible to represent the river network as a table.

Those 18 columns follow follow directly the first column with the segment ID. Afterwards follow the real length and the natural flow (mean, 5th percentile). MicroLowFlows contains a river network in a lowered resolution. The field ’real length’ contains the computed length based on the high-resolution digital river network.

Segment names This file describes with two columns the names of each segment. The first column contains the segment ID, the second the name.

Artifical Influence Flows Within the GREAT-ER project MicroLowFlows has been extended by a special consideration of artifical influences. Examples for these artifical influences are abstractions for drinking water, cooling and irrigation and discharges via treatment plants. The corresponding file contains the improved flow (mean, 5th percentile) for all segments.

Catchment boundary The computation of the natural flows in MicroLowFlows is based on digital elevation data in GRID format. Each square of the area is associated with a height. These data lead to the watershed boundaries. This borderline is given in the file as a line-wise sequence of coordinate pairs. During the computation may occur the so-called problem ’figure of 8’. This means, a part of the catchment area is connected to the rest of the catchment via exactly one point (just like the shape of the digit ’8’).

Coversion to pre-defined formats The data conversion covers three parts: The relatively simple routines of unit conversion (in some `awk` scripts) and the change of the projection of all coordinates to the geographic projection (via ARC/INFO).

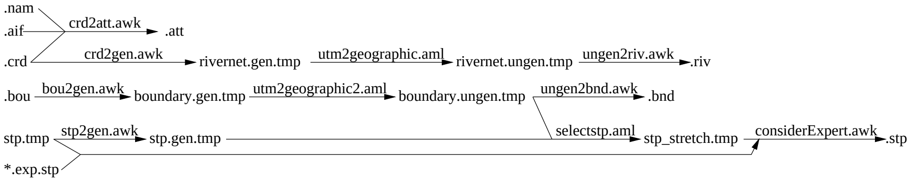


Figure 3.6: *Command sequence for MicroLowFlows data*

(see `~/src/data/raw/mak/makefile.mlf.inc`)

More complicated is the connection of the sewage treatment plant with the river network. For each discharge the closest river segment is computed and the connection is drawn from the discharge site to the start point of the river segment. This may reveal two problems: First, the plant may be connceted wrongly, because due to some reasons the actual pipeline (or carrier canal) is not following the shortest distance to the river. Such errors can only be detected and corrected with local expert knowledge (which is given as a table in the files `*.exp.stp`).

Second, the connection of the plant’s site to the starting point of the closest segment is usually not optimal. A better point on the river would be the one providing the shortest distance to the discharge site. The simulation system of GREAT-ER can only treat segments as a whole and hence only can assume an emission into the starting point of a segment. Hence, for an optimization segments have to be split up.

For this several solutions exist. For GREAT-ER a method was chosen that does not produce entirely new points, but selects the best of the given points of the closest segment. The segment is split up at this point into two subsequent stretches, the downstream stretch is then connected with the discharge site.

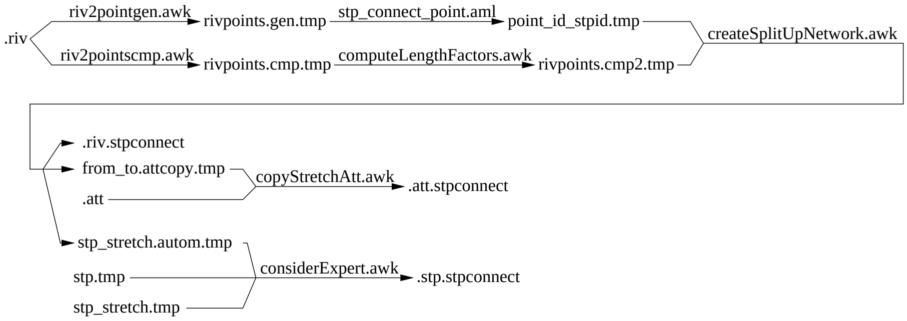


Figure 3.7: *Command sequence for improved discharge connection*

(see `˜/src/data/raw/mak/makefile.stpconnect.inc`)

The downstream part is given a new ID. Both segments get the attributes of the original stretch. One exception of course is the real length. For the computation of the new lengths of the two segments it is assumed that the distances between the single coordinates of the segment is equal. Then factors can be calculated for the lengths.

3.3.2 RhineNet

General information about RhineNet RhineNet is an ARCVIEW based simulation system for the transport and elimination of chemical intermediates in the Rhine and the tributaries Main, Mosel and Lippe. This tool has been developed by Frank Koormann in his Diplomarbeit 1996 at the Institute of Environmental Systems Reseach, University of Osnabrück, Germany.

The river network in RhineNet is available as a so-called 'route'. Discharges are 'events' and associated to a certain point on the route. Analogues all other changes in attributes (e.g. hydrologic data) are stored as events for a route.

Please refer to the ARC/INFO manuals to learn more about routes (part of the segmentation module).

Conversion to pre-defined formats The coverage containing the river network route and events has been processed with the ARC/INFO command `eventarc` to produce the desired segments (arcs). Next, it was necessary to switch the direction of the segments of the tributaries. This had to be done, because in Germany the river kilometers start at the source (with km 0) for those rivers ending in an ocean. Those rivers being a tributary to another river are starting with the kilometer measurement at the point of confluence with km 0. The change of direction has been done manually in ArcEdit, a sub-module of ARC/INFO. Finally the geo-data have exported using the command `ungenerate`.

The actual coordinates of the discharge sites were computed with the ARC/INFO command `eventpoint` based on the route and then also exported with the command `ungenerate`.

The final format adaptions were done manually and were partially supported by the abilities of the text editor VIM.

3.4 Final automatic data processing

The final data processing requires correct source files of the type '.riv', '.stp', '.att', '.bnd' and '.bgd' as described above. The whole processing is managed by a makefile.

This part of the data processing does not consider any special input file format of a simulation system. In the framework of GREAT-ER only those parameters are considered that are the minimal requirement for the simulation software. The simple tabular structure allows to easily extend the data files for adaption to other simulations systems.

The data to be generated can be graded in two groups: geo-data (maps) and the (corresponding) attribute data. The geo-data form the basis for visualization and graphical data processing. These elements (points, lines, polygons) are interconnected with attribute data via unique ID's.

The processing of the background data ('.bgd' file) is not interconnected with the main processing due to its outstanding situation. It is not described in this document (please refer to the corresponding makefiles to learn more about it).

3.4.1 Geo-Data

The establishing of geo-data consists basically of two steps. First, the geographic elements (their ID's and coordinates) are extracted from the files in the pre-defined formats and converted into another format that directly can be handled by ARC/INFO.

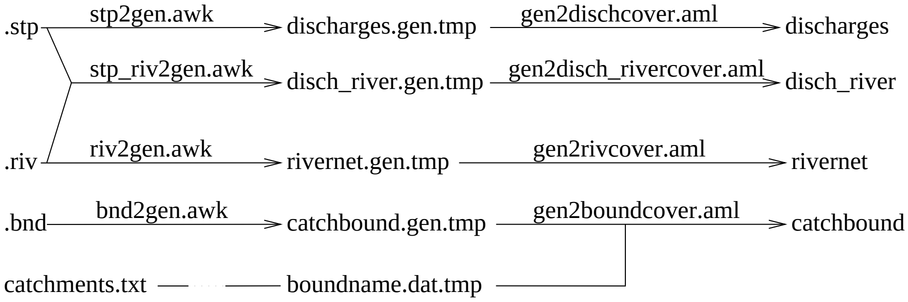


Figure 3.8: *Command sequence for automatic processing of geo-data*

(see `~/src/data/catchments/mak/makefile.final.inc`)

For the two digital maps (coverages) discharge sites ('discharges') and river network ('rivernet') a straight-forward conversion is executed. For the boundary of the catchment ('catchbound') a further attribute is appended: the name of the catchment.

The connection lines between the discharges and corresponding river segments is computed from the data in the '.stp' and '.riv' files and a special coverage is created ('disch_river').

3.4.2 Attribute data

During the processing of the attribute data the three main data files (.stp, .att and .riv, the latter indirectly through the coverage 'rivernet') are joined in a first step to produce in the next step the primary attribute files for river segments ('river.att') and for discharges ('disch.att') and the secondary attribute files for treatment plants ('wwtp.att') and for river classes ('rivclass.att'). The two secondary files are currently static and basically describe generic types of treatment plants and river classes, respectively. The generic information are used when no real data are available.

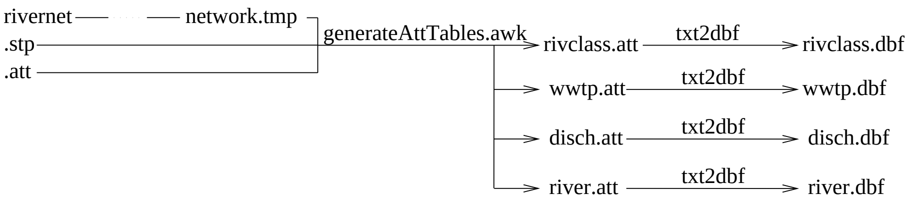


Figure 3.9: *Command sequence for automatic preparation of attribute data*

(see `~/src/data/catchments/mak/makefile.final.inc`)

In a second step of the processing the text based tables are converted into DBase format. The purpose of this is to optimize data management within ArcView.

3.4.3 Core algorithms

Major elements of the data processing consists of simple routines (e.g. format conversions). These will not be described in detail. In this section core problems of the data preparation are discussed.

Creation of a river segment sequence The information about the logic sequence of the single river segments plays an essential role for the simulation model. When applying the efficient tree-walking algorithm, for each segment the upstream segments must be known to be able to follow the river from the most down-stream segment up to the sources.

First, the river segment sequence is created in two steps. This sequence is represented by a two-column table (`fromStretchID,toStretchID`) and describes which segment is downstream of another.

These information can be computed from the digital river network (`rivernet`). This coverage consists of two elements: lines and nodes (see above). When the topology of the coverage is created (ARC/INFO command 'build'), all lines have the two connected nodes as attributes (`fnode`=from_node and `tnode`=to_node). It is then possible to join both two-column tables `fnode,StretchID` and `tnode,StretchID` to produce the desired segment sequence table.

(see `~/src/data/catchments/mak/makefile.final.inc`)

Creation of network topology as attribute table For each segment two columns `up1` and `up2` hold the information about the river network structure. The most usual connection of segments is the 'sequence' and the 'confluence'. The number of two columns is the minimum to describe a river networks topology (a binary tree) when each stretch is allowed to appear only once in the database. For other interconnection types (e.g. bifurcation or confluence of three segments in one point) more columns could be added, but reserved space would only seldomly be used and hence this would not be efficient. Other interconnection types can also be described based on two columns when adding further 'virtual' segments.

(see `~/src/data/catchments/awk/generateAttTables.awk`)

From the above described segment sequence table first an optimal version for further processing is prepared. The table `fromid,toids` as the second column contains a list of the downstream segments. Analogous the table `toid,fromids` contain a list of all upstream segments in the second column.

During the load procedure, the main routine looks up the highest segment ID. This is important to invent further ID's for virtual segments.

When all data are read, it is searched for bifurcation situations. The corresponding segments are associated with their neighbour segments.

Then, the segment table is processed linear and dependend on the number of upstream and downstream segments the type for each segment is decided. The special cases are directly treated.

Table 3.2: *Segment Types*

Type	upstream segments	downstream segments	Description
0	0	0,1,2	Source
1	1	1,2	Normal segment
2	2	0,1	Confluence
-2	1	2	Bifurcation

Confluence of three segments in one point When exactly three segment are directed to the same point, a new (virtual) segment has to be inserted.

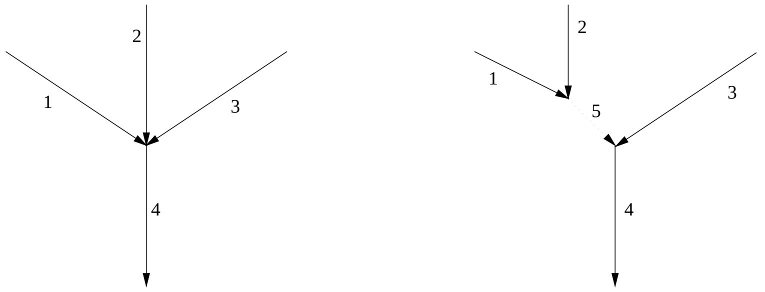


Figure 3.10: *Confluence of three segments*

The virtual segment gets a length of 0. The ID is not appearing in the coverages of the river network and is neutral for the database operation 'join'. The simulation software must of course know how to treat segments with a length of 0: The chemical loads must be transfered (without loss) to the next downstream segment.

Several end-segments If several catchments are joined (e.g. islands) the problem of multiple start segments (eg. the mouths) occurs and needs to be dealt with.

These can be interconnected linear with virtual segments. By convention all ocean-segments are provided with negative ID's.

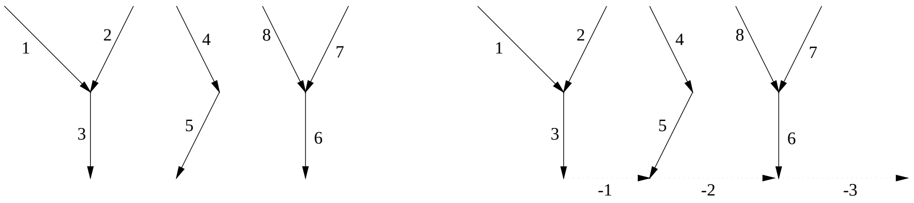


Figure 3.11: *Several end-segments*

Hence the smallest number (in the example -3) resulting from such a situation is the new start segment for the resulting catchment. The ocean segment influence neither visualisation nor simulation.

Bifurcation For bifurcation situations both neighbours are of the type 'Bifurcation'. `up1` will contain the ID of the upstream segment, while `up2` will contain the ID of the neighbour segment.

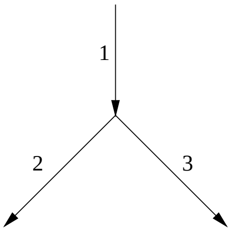
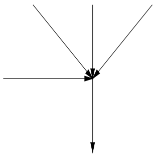
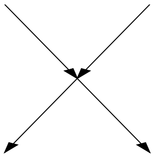
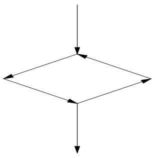


Figure 3.12: *Bifurcations*

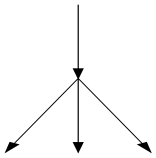
Unconsidered situations

Table 3.3: *Unconsidered situations*

Description	Example
More than three segments are directed to the same point. This situation could be managed similar to the confluence of three segments using virual segments, but from a hydrologic point of view this situation is most unlikely and the chance for wrong base data has a much higher probability.	
In one point both, a confluence and bifurcation takes place. Such a crossing is unlikely. Furthermore, a chemical fate model would need further information to deal with such a situation (what fraction of which upstream segment is going into which downstream segment?).	
The segment sequence form a ring. Creating a ring, a river network is no more a directed, but an indirected graph. The simulation system is not capable to deal with this situation.	

continued on next page

continued from previous page

Description	Example
Seperation of one segment into more than two segments. This situation is unlikely from a hydrologic point of view. An error in the base data is to be supposed.	

4 Data Management within End-User System

Due to the huge quantity of various data used in GREAT-ER, it is necessary to split up the data description into several parts.

The data description will start with the filesystem and explains where to find the different types of data and how they are inter-connected. Next, a hosrt overview on the GIS data is given followed by a list of all GREAT-ER runtime global variables. Subsequent descriptions of the various data structures are given in a tabular form.

4.1 Location of data in the filesystem

The data can be divided into three groups, GIS data, fixed data and user-editable data. The GIS date are always fixed and can not be edited by the user. The following three figures illustrate the location within the GREAT-ER installation filetree.

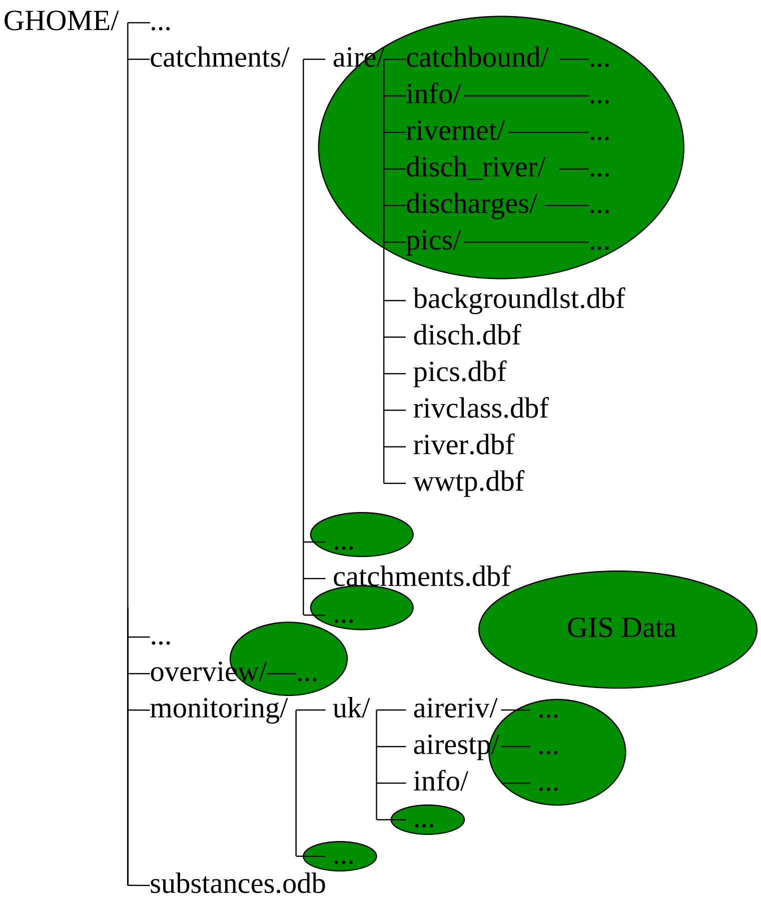


Figure 4.1: *GIS data within the filesystem*

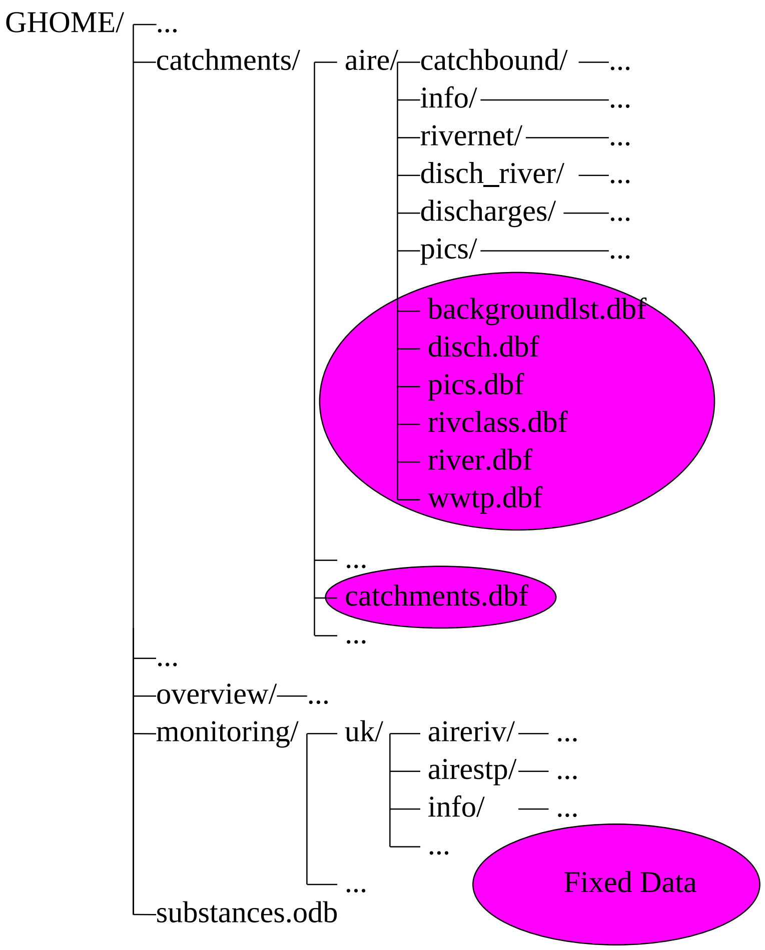


Figure 4.2: *Fixed data within the filesystem*

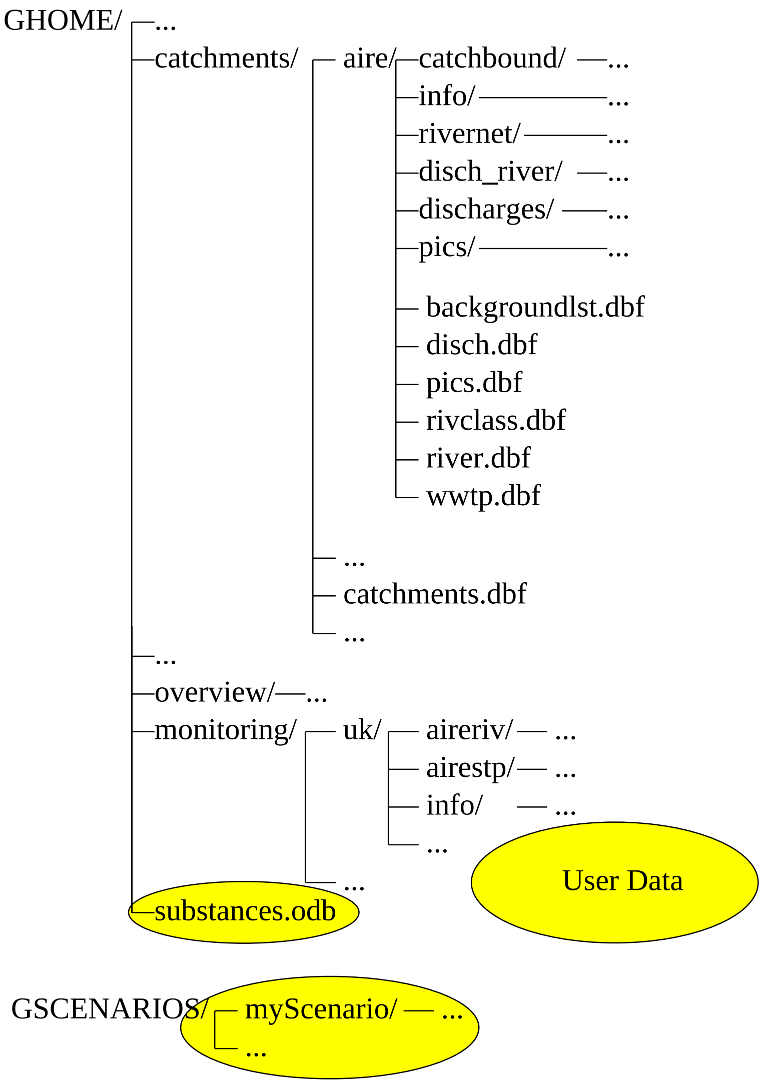


Figure 4.3: *User-editable data within the filesystem*

4.2 Structures of the GIS data

All geographic data are stored as ARC/INFO workspaces. An ARC/INFO workspace is a directory of the filesystem. Each coverage is stored in a subdirectory with the coverage's name. A workspace always contains the directory 'info' where attribute information of the geographic objects are stored.

For GREAT-ER, most of the attribute data are stored in separate DBase files. These are joined to the coverages each time in the startup phase of GREAT-ER.

The following table lists up all ARC/INFO workspaces of a GREAT-ER installation:

Table 4.1: ARC/INFO *Workspaces*

Path	Description	Coverages
GHOME/catchments/aire	Specific geographic data for catchment 'aire'. Analog workspaces exist for all other available catchments.	catchbound, rivernet, disch_river, discharges, pics
GHOME/overview	Maps for general (european) overview.	euview
GHOME/overview/de	All overview maps covering Germany. Analog workspaces for other countries are also existing. These workspaces basically contain a subset of data known as the Digital Charts of the World (DCW).	pppoint, pppoly, rdline, rrline (not fixed)
GHOME/overview/de/itter	Further overview information for a catchment. If geographic data do not serve a general purpose, but instead are tuned for a specific catchment they are separated from the country-workspaces.	cities, rhein, biol_wq (not fixed)
GHOME/monitoring/uk	Collection of coverages with monitoring sites and monitoring data.	aireriv, airestp, ... (not fixed)

All GREAT-ER coverages are in geographic projection. The following table lists up the most important coverages and their type:

Table 4.2: *Main Coverages*

Coverage	Type	Description
catchbound	Polygon	Boundary of a catchment. It is used in a scenario view as well as in the main overview window. No particular attribute data.

continued on next page

continued from previous page

Coverage	Type	Description
rivernet	Line(=Arc)	River network of a catchment. Attributes are stored in the file river.dbf.
discharges	Point	Discharge sites of a catchment. Attributes are stored in the file disch.dbf. Discharge sites are independent of the river network.
disch_river	Line(=Arc)	The lines indicate where the discharges are entering the river network. No particular attributes (except the IDs of the correpsonding discharge point and river stretch: DischID and StretchID).
euview	Polygon	Countries, including names as attributes. These data were taken from the ARCVIEW package and contain some additional attribute data.

4.3 Global Variables

Global variables in Avenue are marked by an underline as the first character. In GREAT-ER all global variables are initialized in the script ’initialize.ave’. See section 5.2 on page 79 for a tabular description in which Avenue scripts the variables are used and/or modified.

Table 4.3: *Most Important Global Variables*

Name	Type	Description
_activeScenario	Dictionary(Scenario)	Points to the currently active scenario. This scenario is in the list of the open scenarios (_openScenarios).
_catchments	VTab	This is the pointer to the Catchment Alias Table.
_eventDelay	Number	This number (in seconds) determines the GREAT-ER-internal interval for updating scenarios. An update signal is send to every view window. Those which are representing a scenario will check for changes (e.g. end of simulation).
_distribTypes	Dictionary	Contains a textual explanation for the distribution type codes (’0’ - ’3’). The type are hard-coded in ’initialize.ave’.
_fieldsDic	Dictionary	Contains the contents of the Fieldname Alias Table (DBase file ’fields.dbf’) in a more dynamical storage.
_flow	Number	Only considered in the menu command ’Select Rivernet by Flow’. This number specifies the minmum flow in m^3/s of river stretches to be shown in the view window.
_greaterDLL	DLL	Pointer to the (loaded) Dynamic Link Library of GREAT-ER. This library is stored in the file GHOME/bin/greater.dll.
_GREATER_STATE	String	This text indicates the state of the GREAT-ER package (e.g. ’alpha’ for a version that is only for internal testing, ’beta’ for a larger testing group).
_GREATER_VERSION	String	Contains the version number of GREAT-ER.
_log	LineFile	Pointer to the log-File. In principle, every GREAT-ER-script writes into the log file when it is started and when it is ended. Several other information are written to this file. The file is opened in the script ’initialize.ave’ and will be removed when GREAT-ER is properly ended.

continued on next page

continued from previous page

Name	Type	Description
_no_dos_box	String	Contains a initialization string for the Avenue command 'System.Execute' to avoid the appearance of a DOS-Box.
_openScenarios	List	Unsorted list of the currently opened scenarios. The elements of this list are of the type Dictionary(Scenario). This list is also used to set up the scenario selection in the GREAT-ER main menu.
_paramsDic	Dictionary	Contains the contents of the Paramsname Alias Table (DBase file 'params.dbf') in a more dynamical storage.
_root	String	Contains the contents of the environment variable GHOME as a string. This is the root path of the currently executing GREAT-ER.
_rootSce	String	Contains the contents of the environment variable GSCENARIOS as a string. This is the root path to the currently active scenario directory.
_substances	ODB	Pointer to the currently loaded substance data base.
_wish	String	Contains the path to the Tcl/Tk shell which is used to avoid DOS-Boxes in some cases.

4.4 Catchment Alias Table

A special data file is GHOME/catchments.dbf. It is used to locate the ARC/INFO workspace of a catchment. This is necessary since the names for workspaces cannot be chosen freely. The following table describes the contents of the Dbase file:

Table 4.4: *Catchment Alias Table* ***catchments.dbf***

Field	Type	Description
ID	String[20]	Identification for a catchment in a nice spelling (using also capital letters and non-alphabetical characters). This entry is used for the selection of a catchment in the Dialogs ’New Scenario’ and ’Edit Scenario’ and also displayed at several occasions in the GREAT-ER easy-to-use part.
Dir	String[20]	Specification of the catchments directory. Only the last part of the path must be specified (it is assumed that the directory is under GHOME/catchments).
Country	String[5]	Short code for the country, the catchment is (mainly) associated to. No official codes are used. Currently only four are defined: uk for United Kingdom, de for Germany, it for Italy and be for Belgium. Basically this two-letter code is the name of the sub-directories in GHOME/overview and GHOME/monitoring. It is used when applying some commands of the ’Display’ menu.

4.5 Fieldname Alias Table

This table contains information on several datafields in GREAT-ER. The DBase file
GHOME/misc/fields.dbf
is copied to a Dictionary for better handling in the Avenue scripts.

Table 4.5: *Fieldname Alias Table* ***fields.dbf***

Field	Type	Description
Variable	String[14]	ID (name) of the field.
Unit	String[25]	Unit of the parameter of the corresponding field in text form. This text is used for the easy-to-use mode identification dialogs.
Display	String[75]	Long (readable) description of the parameter of the corresponding field. This text is used for the easy-to-use mode identification dialogs.
Default	Number[18.9]	Default value. This information is used for only a few fields.
gmode_1	Number[2]	Is this field used for GREAT-ER simulation mode 1 (0=no, 1=yes)?
gmode_2	Number[2]	Is this field used for GREAT-ER simulation mode 2 (0=no, 1=yes)?
gmode_3	Number[2]	Is this field used for GREAT-ER simulation mode 3 (0=no, 1=yes)?

4.6 Parametername Alias Table

This table contains information on several parameters in GREAT-ER. The DBase file
GHOME/misc/params.dbf
is copied to a Dictionary for better handling in the Avenue scripts.

Table 4.6: *Paramtername Alias Table* ***params.dbf***

Field	Type	Description
Variable	String[25]	ID (name) of the variable. This name appears as the Key in the Subdictionaries Substance, ModelMode or CatchmentParamters.
Unit	String[25]	Unit of the parameter in text form. This text is used for the easy-to-use mode identification dialogs.
Display	String[75]	Long (readable) description of the parameter. This text is used for the easy-to-use mode identification dialogs.
Default	Number[18.9]	Default value for the parameter. This number is offered in the dialogs when choosing the 'Default' Button.
DefDistri	Number[2]	Code of the distribution type of the default value.
VarType	String[10]	Semicolon separated list of distribution type codes allowed for this parameter. This information is forwarded to the parameter dialogs to let them avoid senseless distribution selections.
gmode_1	Number[2]	Is this parameter used for GREAT-ER simulation mode 1 (0=no, 1=yes)?
gmode_2	Number[2]	Is this parameter used for GREAT-ER simulation mode 2 (0=no, 1=yes)?
gmode_3	Number[2]	Is this parameter used for GREAT-ER simulation mode 3 (0=no, 1=yes)?
WRange	String[30]	Interval description for a logic range of the parameter value. Refer to the params.dbf contents for examples on the syntax of the range strings. The range strings are only processed in the corresponding dialogs (not in Avenue).
ERange	String[30]	Interval description for a physical range of the parameter value. Refer to the params.dbf contents for examples on the syntax of the range strings. The range strings are only processed in the corresponding dialogs (not in Avenue).

4.7 Parameter Dictionary

During execution of GREAT-ER and for storing in Object Data Base (ODB) format, the ARCVIEW class *Dictionary* is often used for typical collections (please refer to the ARCVIEW manuals for detailed description of this class). The most important one is the *Parameter Dictionary* (in this document referred as *Dictionary(P)*):

Table 4.7: *Subdictionary* ***Parameter***

Key	Type	Description
value	String	The actual value of the paramter. This can either be a simple floating point number or (in the case of an attached distribution) more than one number separated by semicolons. In the case of parameters with a different type (e.g. Boolean) this entry contains a certain coding (e.g. ’0’=false, ’1’=true). The coding is explained in the description of the corresponding parameter.
WRangeExceeded	String	Holds information about exceedance of the warning range. ’0’ indicates that the value is inside the warning range. ’1’ indicates that the value is outside the warning range (values can not be outside the error range).
DistriTyp	String	Describes the distribution type of the parameter: ’0’ = not distributed, ’1’ = normal distribution, ’2’ = log-normal distribution, ’3’ = uniform distribution.
required	String	Is either ’0’ (= not required) or ’1’ (= required). Whether a paramter is required for a simulation with the current model selections is determined via the routine ’f_IsNotRequired.ave’. The information about requirement is basically used by the dialogs which present not required paramters in a different color than black.
Comment	String	A comment text for the corresponding parameter. By default filled with the text ’none’.

Those dictionaries referred as *Dictionary(S)* represent simplified paramters which consists only of the ’value’ entry.

4.8 Substance Database

The default substance database is located in the file GHOME/substances.odb and is an *Object Data Base* (ODB) as used by ARCVIEW. This file is loaded during the start-up phase of GREAT-ER.

The substance database contains a set of substances each being a set of parameters collected in a Dictionary. Dictionaries under ARCVIEW (Avenue) are also known as *collections* in object-oriented terms. A collection is an object that can contain other objects.

For a detailed description of the data types ODB and Dictionary, please refer to the User's Manual of ARCVIEW. For more information about the parameters please refer to the documentation of the simulation software.

The different objects stored in the substance dictionary can be accessed via a *key*. The keys, their type and a short description is given in the following table:

Table 4.8: *Subdictionary **Substance***

Key	Type	Description
`SubstanceID`	Dictionary(S)	The substance name.
`casNo`	Dictionary(S)	The Chemical Abstract Service number of the substance.
`ID`	String	The combination of `SubstanceID` and `casNo`: Identification of the substance used in the window title.
`einecsNo`	Dictionary(S)	The European Inventory of Existing Commercial Chemical Substances number of the substance.
`synonym`	Dictionary(S)	substance synonyms
`M`	Dictionary(P)	Molar mass
`pKa`	Dictionary(P)	acid / base dissociation constant
`protolysis`	Dictionary(P)	acid (1) or base (-1) dissociation factor (neutral = 0)
`K_OW`	Dictionary(P)	octanol / water partitioning coefficient
`ws`	Dictionary(P)	water solubility
`vp`	Dictionary(P)	vapor pressure
`H`	Dictionary(P)	Henry coefficient
`DefaultConsumption`	Dictionary(P)	Consumption used by default in a simulation if no other values are defined.
`K_DO`	Dictionary(P)	biodegradation - affinity constant for oxygen
`Q10`	Dictionary(P)	temperature correction factor for biodegradation
`k_std_biodeg`	Dictionary(P)	standard double first-order biodegradation rate
`a_anaerobic`	Dictionary(P)	anaerobic biodegradation correction factor
`a_anoxic`	Dictionary(P)	anoxic biodegradation correction factor

continued on next page

continued from previous page

Key	Type	Description
`a_sorbed`	Dictionary(P)	sorbed biodegradation correction factor
`k_aerobic`	Dictionary(P)	??? not used any more: standard aerobic biodegradation rate
`useMonod`	Dictionary(P)	Switch for consideration of monod kinetics (0 = no, 1 = yes)
`k_max`	Dictionary(P)	Monod: max. biodegradation rate
`Y`	Dictionary(P)	Monod: biomass yield
`K_s`	Dictionary(P)	Monod: affinity constant
`b`	Dictionary(P)	Monod: biomass decay rate
`k_instream`	Dictionary(P)	default instream removal rate
`k_vol`	Dictionary(P)	default volatilisation rate
`k_deg`	Dictionary(P)	default degradation rate
`k_hydr_a`	Dictionary(P)	acid hydrolysis rate
`k_hydr_b`	Dictionary(P)	base hydroloysis rate
`k_hydr_n`	Dictionary(P)	neutral hydrolysis rate
`k_photo`	Dictionary(P)	near-surface photolysis rate
`a_river`	Dictionary(P)	correction factor for biodegradation in rivers
`K_AW`	Dictionary(P)	dimensionless Henry's Law Constant
`Kd_river`	Dictionary(P)	river - solids / liquid partitioning
`Kd_ML`	Dictionary(P)	mixed liquor - solids / liquid partitioning
`Kd_sewage`	Dictionary(P)	sewage - solids / liquid partitioning
`R_primary`	Dictionary(P)	chemical removal in primary settler
`R_activated_sludge`	Dictionary(P)	chemical removal in activated sludge
`R_trickling_filter`	Dictionary(P)	chemical removal in trickling filter
`a_AS`	Dictionary(P)	correction for biodegradation in activated sludge
`k_WWTP_nonbio`	Dictionary(P)	rate for non-biological degradation in WWTPs
`R_sewer`	Dictionary(P)	chemical removal in sewer
`modified`	Boolean	modified flag for storage control

4.9 Scenario Data

All data related to a scenario are stored in one subdirectory of `GSCENARIOS`. In a scenario's directory the attribute data for the river network, discharges, etc. are stored as DBase files. The DBase files can not directly be modified in the easy-to-use mode. Indirect changes are performed e.g. via the 'Edit Discharge-site Data' routine.

All model parameters and the selected substance data are stored in the ODB file
`GSCENARIOS/myScenario/scenario.odb`.
This file contains one Dictionary which holds all scenario data:

Table 4.9: *Subdictionary **Scenario***

Key	Type	Description
Title	String	Name of the scenario.
modified	Boolean	Indicates whether currently opened scenario has been modified since the last save-operation ('0' = no, '1' = yes).
modifiedSinceLastSim	Boolean	Indicates whether the any parameter has changed since the last simulation ('0' = no, '1' = yes).
Comment	String	Text for commenting upon the scenario.
CatchmentID	String	ID of the selected catchment. The ID appears in the Catchment Alias Table.
CatchmentParameters	Dictionary(Catchment Parameters)	Collection of parameters describing the catchment.
ModelMode	Dictionary(Model Mode)	Collection of selections and switsches for the simulation execution.
Substance	Dictionary(Substance)	Contains a full substance dataset. When selecting a substance from a substance database, a full copy is attached here.
Results	Dictionary(Results)	Collection of results data and pointers to results.

continued on next page

continued from previous page

Key	Type	Description
StartStretch	Dictionary(StartStretch)	Some data for the handling of a start stretch (only existing when a subcatchment is selected.
Tmp	Dictionary(Tmp)	Various temporay information. Before a scenario is saved, these data are deleted.

Table 4.10: *Subdictionary* ***CatchmentParameter***

Key	Type	Description
`temp_water`	Dictionary(P)	water temperature
`temp_sewage`	Dictionary(P)	sewage temperature
`temp_air`	Dictionary(P)	air temperature
`wind_speed`	Dictionary(P)	wind speed
`h_air`	Dictionary(P)	air mixing height
`SS_sewage`	Dictionary(P)	suspended solids in sewage
`f_oc_sewage`	Dictionary(P)	sewage solids OC fraction
`ro_SS_sewage`	Dictionary(P)	sewage solids density
`PHI_cap_BOD`	Dictionary(P)	per capita BOD flux
`PHI_cap_N`	Dictionary(P)	per capita N flux
`R_prim_BOD`	Dictionary(P)	BOD removal efficiency in primary settler
`R_prim_N`	Dictionary(P)	N removal efficiency in primary settler
`r_end`	Dictionary(P)	activated sludge endogenous respiration rate
`f_oc_ML`	Dictionary(P)	ML solids OC fraction
`ro_SS_ML`	Dictionary(P)	ML solids density
`PSI`	Dictionary(P)	empirical constant for surface aeration
`kGa_kLa`	Dictionary(P)	kGa / kLa
`ML_salinity`	Dictionary(P)	salinity of activated sludge ML
`CS_Q_DWF`	Dictionary(P)	combined sewer flow as fraction of DWF
`K_air`	Dictionary(P)	mass tranfer in air
`K_water`	Dictionary(P)	mass tranfer in water
`k_sorb_prim`	Dictionary(P)	1st order (de)sorption rate - primary settler
`k_sorb_ML`	Dictionary(P)	1st order (de)sorption rate - ML
`k_sorb_sec`	Dictionary(P)	1st order (de)sorption rate - secondary settler
`a_prim`	Dictionary(P)	Primary Settler: Effect of Q/DWF on solids removal
`a_sec`	Dictionary(P)	Secondary Settler: Effect of Q/DWF on solids removal

continued on next page

continued from previous page

Key	Type	Description
`temp_corr`	Dictionary(P)	correlation of temperature to flow (both aire and water)
`wind_corr`	Dictionary(P)	correlation of wind speed to flow
`SS_corr`	Dictionary(P)	correlation of SS to flow
`DO_corr`	Dictionary(P)	correlation of DO to flow
`BOD_corr`	Dictionary(P)	correlation of BOD to flow
`k_SS_corr`	Dictionary(P)	correlation off SS sedimentation rate to flow

Table 4.11: *Subdictionary* ***ModelMode***

Key	Type	Description
`mc_shots`	String	Number of Monte Carlo shots to perform
`mode_sewer`	String	Model mode (1, 2 or 3) sewer model
`impose_sewer`	String	flag (0=no, 1=yes): impose selected model mode for all sewer
`mode_AS`	String	Model mode (1, 2 or 3) activated sludge model
`impose_AS`	String	flag (0=no, 1=yes): impose selected model mode for all AS STP
`mode_TF`	String	Model mode (1, 2 or 3) trickling filter model
`impose_TF`	String	flag (0=no, 1=yes): impose selected model mode for all TF STP
`mode_river`	String	Model mode (1, 2 or 3) river model
`impose_river`	String	flag (0=no, 1=yes): impose selected model mode for all stretches
`biodegradation_model`	String	flag (0=off, 1=on) biodegradation model
`hydrolysis_model`	String	flag (0=off, 1=on) hydrolysis model
`photodegradation_model`	String	flag (0=off, 1=on) photodegradation model
`sedimentation_model`	String	flag (0=off, 1=on) sedimentation model
`volatilization_model`	String	flag (0=off, 1=on) volatilization model
`BGConcentration`	Number	background concentration

Table 4.12: *Subdictionary* ***Results***

Key	Type	Description
`FileName`	String	name of model results file
`CsimTheme`	FTheme	basic theme generated after simulation
`CsimvTab`	VTab	table with simulation results

continued on next page

continued from previous page

Key	Type	Description
`RiverFactor`	Number	unit conversion factor assuming original results in [mg/L]
`RiverUnit`	String	new Csim Unit as text
`Classtheme`	FTheme	theme generated by Csim Classes River
`Csim`*XX*	FTheme	Theme of *XX* Csim percentile
`CombineCsimFlow`	FTheme	Theme resulting from the menu command 'Combine Csim/Flow'
`FileNameSTP`	String	name of model results file for intermediate results of influent and effluent
`CsimSTPinfluentTheme`	FTheme	theme showing influent results generated via menu command
`CsimSTPeffluentTheme`	FTheme	theme showing effluent results generated via menu command
`CsimSTPvtab`	VTab	table with simulation results of influents and effluents
`STPFactor`	Number	unit conversion factor assuming original results in [mg/L]
`STPUnit`	String	new Csim Unit as text
`PECinitial`	String	text containing the results of the last PECintial computation.
`PECcatchment`	String	text containing the results of the last PECcatchment computation.

Table 4.13: *Subdictionary* ***Tmp***

Key	Type	Description
chart	Chart	chart of concentration profile. Created when performing corresponding menu command.
dir	FileName	contains the path for the corresponding scenario.
dischTab	Table	ARCVIEW table containing the data of the DBase file *disch.dbf*
marketTab	Table	ARCVIEW table containing the data of the DBase file *market.dbf*. This data set does not initially exist. It is created when additional market information are specified via the menu command 'Edit Market Data'.
odb	ODB	the Object Data Base of the corresponding scenario.
pictureTheme	FTheme	theme containing a point coverage which shows the locations of available photographs.

continued on next page

continued from previous page		
Key	Type	Description
profile	VTab	table containing the results of the routine ’Concentration profile’.
rivclassTab	Table	ARCVIEW table containing the data of the DBase file *rivclass.dbf*
riverTab	Table	ARCVIEW table containing the data of the DBase file *river.dbf*
view	View	the ARCVIEW view where all themes for the corresponding scenario are displayed.
wwtpTab	Table	ARCVIEW table containing the data of the DBase file *wwtp.dbf*

Table 4.14: *Subdictionary* ***StartStretch***

Key	Type	Description
markerTheme	FTheme	theme which contains just a clear marker for the new start segment of the river network.
stretchID	String	the ID of the new start stretch.

5 Reference Tables

The reference tables display the dependencies between various scripts or programmes and the access to shared variables.

This chapter contains the detailed Cross-Reference Tables for the Avenue Scripts and reference tables of global variables used in these scripts. The Tcl/Tk Scripts and the C-based filters do not have cross-references and hence only the Avenue Scripts are listed which execute/call those other programs.

5.1 Avenue Scripts

The Cross-Reference Tables for Avenue-Scripts contain five columns:

- **Calls**: All Avenue Scripts called by the current script by the requests `Run` or `SetApply`.
- **Called by**: All Avenue Scripts calling the current script by the requests `Run` or `SetApply`.
- **Executes**: All external scripts (e.g. Tcl/Tk) or programmes started by the `System`-requests `Execute` or `ExecuteSynchronous`.
- **Linked to Controls**: The menu items or tools the current script is linked to.
- **Global Variables**: All global variables used and modified by the current script

Table 5.1: *Cross-Reference Table* ***ActivateView.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables

Table 5.2: *Cross-Reference Table* ***addCatchView.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
f_UpdateCatchViewTitle.ave View.ZoomToThemes ZoomIn.ave	EditScenario.ave NewScenario.ave OpenScenario.ave SelectCatchment.ave			_catchments _root

Table 5.3: *Cross-Reference Table* ***addDCW.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
	ShowBackground.ave			_activeScenario _catchments _log

Table 5.4: *Cross-Reference Table* ***alphaConcProfile.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
ZoomIn.ave	alphaConcProfileTool.ave			_activeScenario _log

Table 5.5: *Cross-Reference Table* ***alphaConcProfileTool.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
alphaConcProfile.ave				_activeScenario _log

Table 5.6: *Cross-Reference Table* ***alphaSelPopAndFlow.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
				_activeScenario _log

Table 5.7: *Cross-Reference Table* ***BackgroundConc.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
	view_update.ave			_activeScenario _log

Table 5.8: *Cross-Reference Table* ***BaselineConcentration.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
			Background Concentration	_activeScenario _log

Table 5.9: *Cross-Reference Table* ***browseVariables.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables

Table 5.10: *Cross-Reference Table* ***CalcCsimx.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
f_PercentileFromLogNorm.ave	Csim_Classes_River.ave		Calculate River Csim X	_log_activeScenario

Table 5.11: *Cross-Reference Table* ***ChangeSubstanceDB.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
			Change Database	_log _substances

Table 5.12: *Cross-Reference Table* ***CloseScenario.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
	EditScenario.ave		Close Scenario	_activeScenario _log

Table 5.13: *Cross-Reference Table* ***CloseSubstance.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
f_RemoveMarketTab.ave f_RemoveWWTPElimTab.ave f_UpdateCatchViewTitle.ave			Close Substance	_activeScenario

Table 5.14: *Cross-Reference Table* ***CombineCsimFlow.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
f_MultiClassAnalysis.ave		-	Combine Csim/Flow	_activeScenario _log

Table 5.15: *Cross-Reference Table* ***ConcProfile.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
ZoomIn.ave	alphaConcProfileTool.ave ConcProfileTool.ave		Export Profile	_activeScenario

Table 5.16: *Cross-Reference Table* ***ConcProfileTool.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
ConcProfile.ave			Concentration Profile	_activeScenario

Table 5.17: *Cross-Reference Table* ***Csim_Classes_River.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
CalcCsimx.ave		classify txt2dbf	Csim Classes River	_activeScenario _greaterDLL _log _root

Table 5.18: *Cross-Reference Table* ***DeleteScenario.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
f_SelectScenario.ave			Delete Scenario	_log _no_dos_box _openScenarios

Table 5.19: *Cross-Reference Table* ***DeleteSubstance.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
f_ODB_Remove.ave			Delete Substance	_openScenarios _substances

Table 5.20: *Cross-Reference Table* ***Dont_Close.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
		-		_log

Table 5.21: *Cross-Reference Table* ***Easy-to-useMode.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
ViewIdentify.ave	GREAT-ER.ave		Easy-to-use Mode	_GREATER_STATE

Table 5.22: *Cross-Reference Table* ***easy.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
ViewIdentify.ave				_GREATER_STATE

Table 5.23: *Cross-Reference Table* ***EditCatchmentData.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
f_IsNotRequired.ave f_UpdateCatchViewTitle.ave	NewScenario.ave		Edit Model Parameters	_activeScenario _greaterDLL _log

Table 5.24: *Cross-Reference Table* ***EditDischargeData.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
EditDischargeDataTool.ave			Edit Discharge-site Data	_activeScenario _log _root

Table 5.25: *Cross-Reference Table* ***EditDischargeDataTool.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
f_MakeWWTPElimTab.ave f_RemoveWWTPElimTab.ave f_UpdateCatchViewTitle.ave	EditDischargeData.ave			_activeScenario _greaterDLL _log

Table 5.26: *Cross-Reference Table* ***EditEnvironParams.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
f_UpdateCatchViewTitle.ave				_activeScenario _greaterDLL _log

Table 5.27: *Cross-Reference Table* ***EditMarketData.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
EditMarketDataTool.ave			Edit Market Data	_activeScenario _log _root

Table 5.28: *Cross-Reference Table* ***EditMarketDataTool.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
f_MakeMarketTab.ave f_UpdateCatchViewTitle.ave	EditMarketData.ave			_activeScenario _greaterDLL _log

Table 5.29: *Cross-Reference Table* ***EditModelMode.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
f_UpdateCatchViewTitle.ave	NewScenario.ave		Select Model	_activeScenario _greaterDLL _log

Table 5.30: *Cross-Reference Table* ***EditRiverParams.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
f_UpdateCatchViewTitle.ave				_activeScenario _greaterDLL _log

Table 5.31: *Cross-Reference Table* ***EditScenario.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
addCatchView.ave CloseScenario.ave f_CopyDefaultTabs.ave f_DeleteTabs.ave f_RemoveResults.ave f_SaveScenario.ave f_ScenarioDlg.ave f_UpdateCatchViewTitle.ave f_UpdateMenuGreater.ave OpenScenario.ave			Edit Scenario	_activeScenario _log _rootSce

Table 5.32: *Cross-Reference Table* ***EditSewerParams.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
f_UpdateCatchViewTitle.ave				_activeScenario _greaterDLL _log

Table 5.33: *Cross-Reference Table* ***EditSubstance.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
f_CloneSubstance.ave f_RemoveMarketTab.ave f_RemoveWWTPElimTab.ave f_SubstanceDlg.ave f_UpdateCatchViewTitle.ave			Edit Substance	_activeScenario _log

Table 5.34: *Cross-Reference Table* ***EditWWTPParams.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
f_UpdateCatchViewTitle.ave				_activeScenario _greaterDLL _log

Table 5.35: *Cross-Reference Table* ***EnableIfActiveScenario.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
				_openScenarios

Table 5.36: *Cross-Reference Table* ***EnableIfChart.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
				_activeScenario

Table 5.37: *Cross-Reference Table* ***EnableIfCsim.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
				_activeScenario

Table 5.38: *Cross-Reference Table* ***EnableIfIsActiveScenario.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
				_openScenarios

Table 5.39: *Cross-Reference Table* ***EnableIfModelSelected.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
				_openScenarios

Table 5.40: *Cross-Reference Table* ***EnableIfPictures.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
				_activeScenario

Table 5.41: *Cross-Reference Table* ***EnableIfSceHasCatchment.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
				_activeScenario

Table 5.42: *Cross-Reference Table* ***EnableIfSceHasSubstance.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
				_activeScenario

Table 5.43: *Cross-Reference Table* ***EnableIfSubCatchment.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
				_activeScenario

Table 5.44: *Cross-Reference Table* ***EnableIfSubDBAndActiveSce.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
				_openScenarios _substances

Table 5.45: *Cross-Reference Table* ***EnableIfSubstanceDB.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
				_substances

Table 5.46: *Cross-Reference Table* ***ExitGREAT-ER.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
f_SaveScenario.ave			Exit	_activeScenario _log _openScenarios

Table 5.47: *Cross-Reference Table* ***ExpertMode.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
View.Identify			Expert Mode	

Table 5.48: *Cross-Reference Table* ***Export.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
			Export	_greaterDLL _log _root _rootSce

Table 5.49: *Cross-Reference Table* ***ExportConcProfile.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
		dbf2txt	Export Profile	_activeScenario _log _no_dos_box _root

Table 5.50: *Cross-Reference Table* ***Flow.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
			Combine Csim/Flow Select Rivernet by Flow	_flow _log

Table 5.51: *Cross-Reference Table* ***f_CloneModelMode.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
	f_CloneScenario.ave			

Table 5.52: *Cross-Reference Table* ***f_CloneParamList.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
	f_CloneScenario.ave			_log

Table 5.53: *Cross-Reference Table* ***f_CloneResults.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
	f_CloneScenario.ave			

Table 5.54: *Cross-Reference Table* ***f_CloneScenario.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
f_CloneModelMode.ave f_CloneParamList.ave f_CloneResults.ave f_CloneStartStretch.ave f_CloneSubstance.ave f_CloneTmp.ave	f_SaveScenario.ave			_log

Table 5.55: *Cross-Reference Table* ***f_CloneStartStretch.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
	f_CloneScenario.ave			

Table 5.56: *Cross-Reference Table* ***f_CloneSubstance.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
	EditSubstance.ave f_CloneScenario.ave f_ScenarioDlg.ave OpenSubstance.ave PickSubstance.ave SaveSubstance.ave			

Table 5.57: *Cross-Reference Table* ***f_CloneTmp.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
	f_CloneScenario.ave			

Table 5.58: *Cross-Reference Table* ***f_CopyDefaultTabs.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
	EditScenario.ave NewScenario.ave SelectCatchment.ave			_catchments _log _root

Table 5.59: *Cross-Reference Table* ***f_DeleteTabs.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
f_RemoveTabs.ave	EditScenario.ave SelectCatchment.ave view_close.ave			_log

Table 5.60: *Cross-Reference Table* ***f_DetermineUnit.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
	view_update.ave			_log

Table 5.61: *Cross-Reference Table* ***f_IsNotRequired.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
	EditCatchmentData.ave f_SubstanceDlg.ave SimulDataCheck.ave	-		_activeScenario _log _paramsDic

Table 5.62: *Cross-Reference Table* ***f_LogNorm2Norm.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
	f_PercentileFromLogNorm.ave			

Table 5.63: *Cross-Reference Table* ***f_MakeMarketTab.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
f_RemoveMarketTab.ave	EditMarketDataTool.ave			_activeScenario _log

Table 5.64: *Cross-Reference Table* ***f_MakeWWTPElimTab.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
f_RemoveWWTPElimTab.ave	EditDischargeDataTool.ave			_activeScenario _log

Table 5.65: *Cross-Reference Table* ***f_MultiClassAnalysis.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
	CombineCsimFlow.ave	-		_log

Table 5.66: *Cross-Reference Table **f_Norm2LogNorm.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables

Table 5.67: *Cross-Reference Table **f_ODB_Remove.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
	DeleteSubstance.ave SaveSubstance.ave			_log

Table 5.68: *Cross-Reference Table **f_PercentileFromLogNorm.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
f_LogNorm2Norm.ave f_PercentileFromStdNorm.ave f_StdNorm2Norm.ave	CalcCsimx.ave			

Table 5.69: *Cross-Reference Table **f_PercentileFromStdNorm.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
	f_PercentileFromLogNorm.ave			

Table 5.70: *Cross-Reference Table **f_RemoveMarketTab.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
	CloseSubstance.ave EditSubstance.ave f_MakeMarketTab.ave LoadSubstance.ave			_activeScenario _log

Table 5.71: *Cross-Reference Table* ***f_RemoveResults.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
	EditScenario.ave SelectSubCatchmentTool.ave StartSimulation.ave UnselectSubCatchment.ave	-		_activeScenario _log

Table 5.72: *Cross-Reference Table* ***f_RemoveTabs.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
	f_DeleteTabs.ave			_log

Table 5.73: *Cross-Reference Table* ***f_RemoveWWTPElimTab.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
	CloseSubstance.ave EditDischargeDataTool.ave EditSubstance.ave f_MakeWWTPElimTab.ave LoadSubstance.ave			_activeScenario _log

Table 5.74: *Cross-Reference Table* ***f_SaveScenario.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
f_CloneScenario.ave	EditScenario.ave ExitGREAT-ER.ave SaveScenario.ave SaveScenarioAs.ave view_close.ave			_greaterDLL _log

Table 5.75: *Cross-Reference Table* ***f_ScenarioDlg.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
f_CloneSubstance.ave	EditScenario.ave NewScenario.ave			_catchments _greaterDLL _log _substances

Table 5.76: *Cross-Reference Table* ***f_ScenarioInfo.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
				_log

Table 5.77: *Cross-Reference Table* ***f_SelectAll.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
	PECcatchment.ave	-		_log

Table 5.78: *Cross-Reference Table* ***f_SelectLoaded.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
	PECcatchment.ave	-		_log

Table 5.79: *Cross-Reference Table* ***f_SelectMostDownUnloaded.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
	PECcatchment.ave	-		_log

Table 5.80: *Cross-Reference Table* ***f_SelectScenario.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
	DeleteScenario.ave OpenScenario.ave			_greaterDLL _log _rootSce

Table 5.81: *Cross-Reference Table* ***f_SelectUnloaded.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
		-		_activeScenario _log

Table 5.82: *Cross-Reference Table* ***f_StdNorm2Norm.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
	f_PercentileFromLogNorm.ave			

Table 5.83: *Cross-Reference Table* ***f_SubstanceDlg.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
f_IsNotRequired.ave	EditSubstance.ave NewSubstance.ave	wish substance.tcl		_greaterDLL _log

Table 5.84: *Cross-Reference Table* ***f_UpdateCatchViewTitle.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
	addCatchView.ave CloseSubstance.ave EditCatchmentData.ave EditDischargeDataTool.ave EditEnvironParams.ave EditMarketDataTool.ave EditModelMode.ave EditRiverParams.ave EditScenario.ave EditSewerParams.ave EditSubstance.ave EditWWTPParams.ave LoadSubstance.ave NewSubstance.ave SaveScenario.ave SaveSubstance.ave StopSimulation.ave view_update.ave			_log

Table 5.85: *Cross-Reference Table* ***f_UpdateMenuGreater.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
	EditScenario.ave NewScenario.ave OpenScenario.ave view_close.ave view_update.ave			_activeScenario _openScenarios

Table 5.86: *Cross-Reference Table* ***f_WriteSimulInput.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
	StartSimulation.ave	-		_activeScenario _log _no_dos_box

Table 5.87: *Cross-Reference Table* ***GREAT-ER.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
Easy-to-useMode.ave initialize.ave initMenu.ave initPopUp.ave ZoomIn.ave	StartGREAT-ER.ave		Exit	_catchments _GREATER_STATE _log _root

Table 5.88: *Cross-Reference Table* ***Help_Info.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
			About GREAT-ER	_GREATER_STATE _GREATER_VERSION

Table 5.89: *Cross-Reference Table* ***Help_Model.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
			About Model	_greaterDLL _log

Table 5.90: *Cross-Reference Table* ***Help_UI.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
			Contents	_greaterDLL _log

Table 5.91: *Cross-Reference Table* ***Identify.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
	Easy-to-useMode.ave easy.ave		Identify	

Table 5.92: *Cross-Reference Table* ***Import.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
			Import	_greaterDLL _log _root _rootSce

Table 5.93: *Cross-Reference Table* ***imsConnect.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
	imsTest.ave			_log _WebServerURL

Table 5.94: *Cross-Reference Table* ***imsGetMapPoint.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
	imsMapClick.ave imsTest.ave			_log

Table 5.95: *Cross-Reference Table* ***imsMainPage.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
	imsTest.ave			_log

Table 5.96: *Cross-Reference Table* ***imsMakeHTML.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
imsMapClick.ave	imsTest.ave			_FieldList _log _RecordList

Table 5.97: *Cross-Reference Table* ***imsMakeMap.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
imsMapClick.ave	imsTest.ave			_log

Table 5.98: *Cross-Reference Table* ***imsMapClick.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
imsGetMapPoint.ave	imsMakeHTML.ave imsMakeMap.ave imsTest.ave			

Table 5.99: *Cross-Reference Table* ***imsParse.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
	imsTest.ave			_log

Table 5.100: *Cross-Reference Table* ***imsTest.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
imsConnect.ave ReloadScript.ave				_log

Table 5.101: *Cross-Reference Table* ***initialize.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
	GREAT-ER.ave			_activeScenario _catchments _distribTypes _eventDelay _fieldsDic _flow _greaterDLL _GREATER_STATE _GREATER_VERSION _log _no_dos_box _openScenarios _paramsDic _root _rootSce _substances _wish

Table 5.102: *Cross-Reference Table* ***initMenu.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
	GREAT-ER.ave			

Table 5.103: *Cross-Reference Table* ***initPopUp.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
	GREAT-ER.ave			_log

Table 5.104: *Cross-Reference Table* ***LoadSubstance.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
f_RemoveMarketTab.ave f_RemoveWWTPElimTab.ave f_UpdateCatchViewTitle.ave SaveSubstance.ave				_activeScenario _log _substances

Table 5.105: *Cross-Reference Table* ***NewScenario.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
addCatchView.ave EditCatchmentData.ave EditModelMode.ave f_CopyDefaultTabs.ave f_ScenarioDlg.ave f_UpdateMenuGreater.ave			New Scenario	_activeScenario _log _openScenarios _rootSce

Table 5.106: *Cross-Reference Table* ***NewSubstance.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
f_SubstanceDlg.ave f_UpdateCatchViewTitle.ave			New Substance	_activeScenario _log

Table 5.107: *Cross-Reference Table* ***OpenActiveScenario.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
			n Scenario Title	_activeScenario

Table 5.108: *Cross-Reference Table* ***OpenScenario.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
addCatchView.ave f_SelectScenario.ave f_UpdateMenuGreater.ave SelectSubCatchmentMarker.ave	EditScenario.ave		Open Scenario	_activeScenario _log _openScenarios

Table 5.109: *Cross-Reference Table* ***OpenSubstance.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
f_CloneSubstance.ave			Open Substance	_activeScenario _substances

Table 5.110: *Cross-Reference Table* ***PECcatchment.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
f_SelectAll.ave f_SelectLoaded.ave f_SelectMostDownUnloaded.ave			PECcatchment	_activeScenario _log

Table 5.111: *Cross-Reference Table* ***PECinitial.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
			PECinitial	_activeScenario _log

Table 5.112: *Cross-Reference Table* ***PickSubstance.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
f_CloneSubstance.ave			Pick Substance	_activeScenario _log _openScenarios

Table 5.113: *Cross-Reference Table* ***Picture.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
PictureTool.ave			Show Site Pictures	_activeScenario _log

Table 5.114: *Cross-Reference Table* ***PictureTool.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
	Picture.ave			_activeScenario _log

Table 5.115: *Cross-Reference Table* ***ReloadAllScripts.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
ReloadScript.ave				

Table 5.116: *Cross-Reference Table* ***ReloadAnyScript.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
ReloadScript.ave				

Table 5.117: *Cross-Reference Table* ***ReloadScript.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
	imsTest.ave ReloadAllScripts.ave ReloadAnyScript.ave StartGREAT-ER.ave			

Table 5.118: *Cross-Reference Table* ***RemoveBackgroundData.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
removeDCW.ave			Remove Background Data	_activeScenario _catchments _log

Table 5.119: *Cross-Reference Table* ***removeDCW.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
	RemoveBackgroundData.ave			_activeScenario _catchments _log

Table 5.120: *Cross-Reference Table* ***RemoveThemes.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
			Remove Themes	_log

Table 5.121: *Cross-Reference Table* ***Report.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
		-	Report	_activeScenario _distribTypes _fieldsDic _GREATER_STATE _GREATER_VERSION _log _paramsDic

Table 5.122: *Cross-Reference Table* ***RiverFlows.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
			Show River Flows	_activeScenario _log

Table 5.123: *Cross-Reference Table* ***SaveScenario.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
f_SaveScenario.ave f_UpdateCatchViewTitle.ave	EditScenario.ave ExitGREAT-ER.ave SaveScenario.ave SaveScenarioAs.ave view_close.ave		Save Scenario	_activeScenario

Table 5.124: *Cross-Reference Table* ***SaveScenarioAs.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
f_SaveScenario.ave			Save Scenario as	_activeScenario _log

Table 5.125: *Cross-Reference Table* ***SaveSubstance.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
f_CloneSubstance.ave f_ODB_Remove.ave f_UpdateCatchViewTitle.ave	LoadSubstance.ave		Save Substance	_activeScenario _substances

Table 5.126: *Cross-Reference Table* ***SelectBackgroundData.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
ShowBackground.ave			Add Background Data	_activeScenario _catchments _log

Table 5.127: *Cross-Reference Table* ***SelectCatchment.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
addCatchView.ave f_CopyDefaultTabs.ave f_DeleteTabs.ave	SelectCatchmentByName.ave			_activeScenario _catchments _log _root

Table 5.128: *Cross-Reference Table* ***SelectCatchmentByName.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
SelectCatchment.ave			Select Catchment	_catchments _log

Table 5.129: *Cross-Reference Table* ***SelectSubCatchmentFromView.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
SelectSubCatchmentMarker.ave ZoomIn.ave	SelectSubCatchmentTool.ave			_activeScenario _log

Table 5.130: *Cross-Reference Table* ***SelectSubCatchmentMarker.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
	OpenScenario.ave SelectSubCatchmentFromView.ave			_activeScenario _log

Table 5.131: *Cross-Reference Table* ***SelectSubCatchmentTool.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
f_RemoveResults.ave SelectSubCatchmentFromViewave			Select Subcatchment	_activeScenario _log

Table 5.132: *Cross-Reference Table* ***sendEvents.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
				_eventDelay _log

Table 5.133: *Cross-Reference Table* ***ShowBackground.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
addDCW.ave	SelectBackgroundData.ave			_activeScenario _catchments _log

Table 5.134: *Cross-Reference Table* ***ShowSTPEfflCsim.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
			Discharge Effluent Csim	_activeScenario _log

Table 5.135: *Cross-Reference Table* ***ShowSTPInflCsim.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
			Discharge Influent Csim	_activeScenario _log

Table 5.136: *Cross-Reference Table* ***SimulDataCheck.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
f_IsNotRequired.ave				_activeScenario _catchments _log _paramsDic

Table 5.137: *Cross-Reference Table* ***StartGREAT-ER.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
GREAT-ER.ave ReloadScript.ave				

Table 5.138: *Cross-Reference Table* ***StartSimulation.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
f_RemoveResults.ave f_WriteSimulInput.ave		simul gawk simSTP2db4.awk gawk simul2db4.awk txt2dbf	Start Simulation	_no_dos_box _activeScenario _catchments _greaterDLL _log

Table 5.139: *Cross-Reference Table* ***StopSimulation.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
f_UpdateCatchViewTitle.ave			Stop Simulation	_activeScenario _log

Table 5.140: *Cross-Reference Table* ***UnselectSubCatchment.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
f_RemoveResults.ave		-	Unselect Subcatchment	

Table 5.141: *Cross-Reference Table* ***ViewIdentify.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
	Easy-to-useMode.ave easy.ave			_greaterDLL _log _unitsTab

Table 5.142: *Cross-Reference Table* ***ViewLogfile.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
				_log

Table 5.143: *Cross-Reference Table* ***view_close.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
f_DeleteTabs.ave f_SaveScenario.ave f_UpdateMenuGreater.ave ZoomIn.ave				_activeScenario _log _openScenarios

Table 5.144: *Cross-Reference Table* ***view_update.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
BackgroundConc.ave f_DetermineUnit.ave f_UpdateCatchViewTitle.ave f_UpdateMenuGreater.ave				_activeScenario _catchments _log _root

Table 5.145: *Cross-Reference Table* ***ZoomCatchmentExtent.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
			Full Catchment Extent	_log

Table 5.146: *Cross-Reference Table* ***ZoomIn.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
	addCatchView.ave alphaConcProfile.ave ConcProfile.ave GREAT-ER.ave SelectSubCatchmentFromView.ave view_close.ave		Zoom In	

Table 5.147: *Cross-Reference Table* ***ZoomOut.ave***

Calls	Called by	Executes	Linked to Controls	Global Variables
			Zoom Out	

5.2 Avenue Global Variables

The following tables list all Avenue Scripts the named global variable is used in. See section 4.3 on page 33 for a detailed description of the most important global variables.

Table 5.148: *Reference Table Global Variable* `_FieldList`

`_FieldList` used in
imsMakeHTML.ave

Table 5.149: *Reference Table Global Variable* `_GREATER_STATE`

`_GREATER_STATE` used in	
Easy-to-useMode.ave	GREAT-ER.ave
Help_Info.ave	Report.ave
easy.ave	initialize.ave

Table 5.150: *Reference Table Global Variable* `_GREATER_VERSION`

`_GREATER_VERSION` used in	
Help_Info.ave	Report.ave
initialize.ave	

Table 5.151: *Reference Table Global Variable* `_RecordList`

`_RecordList` used in
imsMakeHTML.ave

Table 5.152: *Reference Table Global Variable* `_WebServerURL`

`_WebServerURL` used in
imsConnect.ave

Table 5.153: *Reference Table Global Variable* `_activeScenario`

`_activeScenario` used in	
BackgroundConc.ave	BaselineConcentration.ave
CalcCsimx.ave	CloseScenario.ave
CloseSubstance.ave	CombineCsimFlow.ave
ConcProfile.ave	ConcProfileTool.ave
Csim_Classes_River.ave	EditCatchmentData.ave
EditDischargeData.ave	EditDischargeDataTool.ave
EditEnvironParams.ave	EditMarketData.ave
EditMarketDataTool.ave	EditModelMode.ave
EditRiverParams.ave	EditScenario.ave
EditSewerParams.ave	EditSubstance.ave
EditWWTPParams.ave	EnableIfChart.ave
EnableIfCsim.ave	EnableIfPictures.ave
EnableIfSceHasCatchment.ave	EnableIfSceHasSubstance.ave
EnableIfSubCatchment.ave	ExitGREAT-ER.ave
ExportConcProfile.ave	LoadSubstance.ave
NewScenario.ave	NewSubstance.ave
OpenActiveScenario.ave	OpenScenario.ave
OpenSubstance.ave	PEC_RegionaLRiver.ave
PEC_initial.ave	PECcatchment.ave
PECinitial.ave	PickSubstance.ave
Picture.ave	PictureTool.ave
RemoveBackgroundData.ave	Report.ave
RiverFlows.ave	SaveScenario.ave
SaveScenarioAs.ave	SaveSubstance.ave
SelectBackgroundData.ave	SelectCatchment.ave
SelectSubCatchmentFromView.ave	SelectSubCatchmentMarker.ave
SelectSubCatchmentTool.ave	ShowBackground.ave
ShowSTPEfflCsim.ave	ShowSTPInflCsim.ave
SimulDataCheck.ave	StartSimulation.ave
StopSimulation.ave	addDCW.ave
alphaConcProfile.ave	alphaConcProfileTool.ave
alphaSelPopAndFlow.ave	f_IsNotRequired.ave
f_MakeMarketTab.ave	f_MakeWWTPElimTab.ave
f_RemoveMarketTab.ave	f_RemoveResults.ave
f_RemoveWWTPElimTab.ave	f_SelectUnloaded.ave
f_UpdateMenuGreater.ave	f_WriteSimulInput.ave
initialize.ave	removeDCW.ave
view_close.ave	view_update.ave

Table 5.154: *Reference Table Global Variable* `_catchments`

`_catchments` used in	
GREAT-ER.ave	RemoveBackgroundData.ave
SelectBackgroundData.ave	SelectCatchment.ave
SelectCatchmentByName.ave	ShowBackground.ave
SimulDataCheck.ave	StartSimulation.ave
addCatchView.ave	addDCW.ave
f_CopyDefaultTabs.ave	f_ScenarioDlg.ave
initialize.ave	removeDCW.ave
view_update.ave	

Table 5.155: *Reference Table Global Variable* `_distribTypes`

`_distribTypes` used in	
Report.ave	initialize.ave

Table 5.156: *Reference Table Global Variable* `_eventDelay`

`_eventDelay` used in	
initialize.ave	sendEvents.ave

Table 5.157: *Reference Table Global Variable* `_fieldsDic`

`_fieldsDic` used in	
Report.ave	initialize.ave

Table 5.158: *Reference Table Global Variable* `_flow`

`_flow` used in	
Flow.ave	initialize.ave

Table 5.159: *Reference Table Global Variable* `_greaterDLL`

`_greaterDLL` used in	
Csim_Classes_River.ave	EditCatchmentData.ave
EditDischargeDataTool.ave	EditEnvironParams.ave
EditMarketDataTool.ave	EditModelMode.ave
EditRiverParams.ave	EditSewerParams.ave
EditWWTPParams.ave	Export.ave
Help_Model.ave	Help_UI.ave
Import.ave	StartSimulation.ave
ViewIdentify.ave	f_SaveScenario.ave
f_ScenarioDlg.ave	f_SelectScenario.ave
f_SubstanceDlg.ave	initialize.ave

Table 5.160: *Reference Table Global Variable* `_log`

`_log` used in	
BackgroundConc.ave	BaselineConcentration.ave
CalcCsimx.ave	ChangeSubstanceDB.ave
CloseScenario.ave	CombineCsimFlow.ave
Csim_Classes_River.ave	DeleteScenario.ave
Dont_Close.ave	EditCatchmentData.ave
EditDischargeData.ave	EditDischargeDataTool.ave
EditEnvironParams.ave	EditMarketData.ave
EditMarketDataTool.ave	EditModelMode.ave
EditRiverParams.ave	EditScenario.ave
EditSewerParams.ave	EditSubstance.ave
EditWWTPParams.ave	ExitGREAT-ER.ave
Export.ave	ExportConcProfile.ave
Flow.ave	GREAT-ER.ave
Help_Model.ave	Help_UI.ave
Import.ave	LoadSubstance.ave
NewScenario.ave	NewSubstance.ave
OpenScenario.ave	PEC_Regional_River.ave
PEC_initial.ave	PECcatchment.ave
PECinitial.ave	PickSubstance.ave
Picture.ave	PictureTool.ave
RemoveBackgroundData.ave	RemoveThemes.ave
Report.ave	RiverFlows.ave
SaveScenarioAs.ave	SelectBackgroundData.ave
SelectCatchment.ave	SelectCatchmentByName.ave
SelectSubCatchmentFromView.ave	SelectSubCatchmentMarker.ave
SelectSubCatchmentTool.ave	ShowBackground.ave
ShowSTPEfflCsim.ave	ShowSTPInflCsim.ave
SimulDataCheck.ave	StartSimulation.ave
StopSimulation.ave	ViewIdentify.ave
ViewLogfile.ave	ZoomCatchmentExtent.ave
addDCW.ave	alphaConcProfile.ave
alphaConcProfileTool.ave	alphaSelPopAndFlow.ave
f_CloneParamList.ave	f_CloneScenario.ave
f_CopyDefaultTabs.ave	f_DeleteTabs.ave
f_DetermineUnit.ave	f_IsNotRequired.ave
f_MakeMarketTab.ave	f_MakeWWTPElimTab.ave
f_MultiClassAnalysis.ave	f_ODB_Remove.ave
f_RemoveMarketTab.ave	f_RemoveResults.ave
f_RemoveTabs.ave	f_RemoveWWTPElimTab.ave
f_SaveScenario.ave	f_ScenarioDlg.ave
f_ScenarioInfo.ave	f_SelectAll.ave
f_SelectLoaded.ave	f_SelectMostDownUnloaded.ave
f_SelectScenario.ave	f_SelectUnloaded.ave
f_SubstanceDlg.ave	f_UpdateCatchViewTitle.ave
f_WriteSimulInput.ave	imsConnect.ave
imsGetMapPoint.ave	imsMainPage.ave
imsMakeHTML.ave	imsMakeMap.ave
imsParse.ave	imsTest.ave
initPopUp.ave	initialize.ave
removeDCW.ave	sendEvents.ave
view_close.ave	view_update.ave

Table 5.161: *Reference Table Global Variable* `_no_dos_box`

`_no_dos_box` used in	
DeleteScenario.ave	ExportConcProfile.ave
StartSimulation.ave	f_WriteSimulInput.ave
initialize.ave	

Table 5.162: *Reference Table Global Variable* `_openScenarios`

`_openScenarios` used in	
DeleteScenario.ave	DeleteSubstance.ave
EnableIfActiveScenario.ave	EnableIfIsActiveScenario.ave
EnableIfModelSelected.ave	EnableIfSubDBAndActiveSce.ave
ExitGREAT-ER.ave	NewScenario.ave
OpenScenario.ave	PickSubstance.ave
f_UpdateMenuGreater.ave	initialize.ave
view_close.ave	

Table 5.163: *Reference Table Global Variable* `_paramsDic`

`_paramsDic` used in	
Report.ave	SimulDataCheck.ave
f_IsNotRequired.ave	initialize.ave

Table 5.164: *Reference Table Global Variable* `_root`

`_root` used in	
Csim_Classes_River.ave	EditDischargeData.ave
EditMarketData.ave	EditScenario.ave
Export.ave	ExportConcProfile.ave
GREAT-ER.ave	Import.ave
NewScenario.ave	SelectCatchment.ave
addCatchView.ave	f_CopyDefaultTabs.ave
f_SelectScenario.ave	initialize.ave
view_update.ave	

Table 5.165: *Reference Table Global Variable* `_rootSce`

`_rootSce` used in	
EditScenario.ave	Export.ave
Import.ave	NewScenario.ave
f_SelectScenario.ave	initialize.ave

Table 5.166: *Reference Table Global Variable* `_substances`

`_substances` used in	
ChangeSubstanceDB.ave	DeleteSubstance.ave
EnableIfSubDBAndActiveSce.ave	EnableIfSubstanceDB.ave
LoadSubstance.ave	OpenSubstance.ave
SaveSubstance.ave	f_ScenarioDlg.ave
initialize.ave	

Table 5.167: *Reference Table Global Variable* `_unitsTab`

`_unitsTab` used in
ViewIdentify.ave

Table 5.168: *Reference Table Global Variable* `_wish`

`_wish` used in
initialize.ave

5.3 Scripts / Programs

The following tables list all Avenue Scripts the named script / programme is called by. Please note that for the execution of the scripts the referring interpreter must be started. These are WISH for Tcl/Tk-Scripts and GAWK for awk-Scripts

Table 5.169: *Reference Table Script / Programme* `wgnuplot`

`wgnuplot` called in
PEC_initial.ave

Table 5.170: *Reference Table Script / Programme* `classify`

`classify` called in
Csim_Classes_River.ave

Table 5.171: *Reference Table Script / Programme* `dbf2txt`

`dbf2txt` called in
ExportConcProfile.ave

Table 5.172: *Reference Table Script / Programme* `simSTP2db4.awk`

`simSTP2db4.awk` called in
StartSimulation.ave

Table 5.173: *Reference Table Script / Programme* `simul2db4.awk`

`simul2db4.awk` called in
StartSimulation.ave

Table 5.174: *Reference Table Script / Programme* `simul`

`simul` called in
StartSimulation.ave

Table 5.175: *Reference Table Script / Programme* `txt2dbf`

`txt2dbf` called in	
Csim_Classes_River.ave	StartSimulation.ave

6 GREAT-ER Messages

6.1 Conventions

We have set up certain typographic conventions to accentuate the meaning of words mentioned in this manual. The following table describes the main conventions:

Style	Meaning
Serif	Usual text
Italic	Emphasis
SMALL CAPITALS	Products / Software
Sans Serif	Menu items
Slanted	Input
`Typewriter`	Software output, messages
`Typewriter slanted`	File names, directories
<**PATH**>	System variables, e.g. *PATH*

6.2 Hints

Hints are usually only informational. Although something has failed there is no real error in the sense of loss of data or inavailable functions. Following the hints and the additional remarks given in this section the problems can be solved.

1. `No substance database found - creating new.`
   For some reason the default substance database could not be found. Therefore a new file entitled *`substance.odb`* was created to generate a new default database. You may try to install manually or to copying another substance DB to the default name *`substance.odb`*.

2. `Sorry, it is not allowed to close this window.`
   You have tried to close the view of Europe (the GREAT-ER basic view) by the clicking on the close button (”X”) on the window title bar. The resulting process would be similar to the menu item Exit but is not stable and may cause loss of data. For this reason the action is prevented. Please use the Exit item to close GREAT-ER.

3. `Sorry there is already an open scenario with this title.`
   It is not allowed to have two open scenarios with the same name. The scenario name is the key to identifying a scenario and must be unique. Please close the currently open scenario with the name in question if you want to open the new one.

4. `It is necessary to first save the scenario.`
   For reasons of stability and data safety it is only allowed to change catchment data for an unmodified scenario, i.e. a scenario directly saved before changing the catchment data.

5. `No substance specified for active scenario`
   No substance found for active scenario. As market data and discharge-site data dialogs offer users the opportunity to overwrite general substance settings by site-specific values, these are only applicable if a substance has already been selected.

6. `A simulation for this scenario seems to be still in progess.`
   This is a typical hint: currently not available, the function can be executed some time later because it is not possible to run more than one simulation at a time for a scenario.

7. `This selection tool has to be applied to the active scenario view`
   You have clicked on the view with the European overview while the subcatchment selection tool was active. The message appears, because it is not possible to select a stretch from the overview.

6.3 Warnings

Warnings are given if something has gone wrong. These are different to errors in the sense that they do not cause a loss of data and may be resolved with the use of the technical documentation.

1. `Environment variable <GHOME> not set.`
   Although this environment variable needed by GREAT-ER (specifying the GREAT-ER home directory) is set by the set-up program, it is now missing (maybe deleted manually). A following dialog enables you to specify the home directory for the current GREAT-ER session but it is recommended to set the variable permanently with the system settings.

2. `Environment variable <GSCENARIOS> not set.`
   Although this environment variable needed by GREAT-ER (specifying the GREAT-ER sceanrio directory) is set by the set-up program, it is now missing (maybe deleted manually). A following dialog enables you to specify the home directory for the current GREAT-ER session but it is recommended to set the variable permanently with the system settings.

3. `No unit information available. Unit is set to [mg/L]. Please check your system / scenario parameters!`
   For some reason the automatic unit conversion has failed. The simulator output is [mg/L] so it is assumed that the loaded data is in this unit. Please check this assumption. This message also appears when no release of the substance has been specified (e.g. a catchment with only industrial discharges and no site-specific release).

4. `GREAT-ER menu not found.`
   The GREAT-ER menu structure seems to be corrupt. This can happen by modifying the structure in the Expert Mode. A rough solution is to leave GREAT-ER without saving anything. If you want to save your scenario data you have to restore the menu structure: See the ARCVIEW manual for customising a project. Check the menus for the View Doc. Rename the one which contains the usual GREAT-ER commands back to ”`&GREAT-ER`”.

5. `Exit item not found.`
   The GREAT-ER menu structure seems to be corrupted. This could happen by modifying the structure in the Expert Mode. A rough solution is to leave GREAT-ER. You may save your scenarios before leaving. An alternative is to restore the menu structure: See the ArcView manual for customising a project. Look for the GREAT-ER menu for the View Doc. Scan through the menu items and find the item with the click script ExitGREAT-ER.ave Rename it to ”`E&xit`”.

6. `Not able to fully delete the scenario. The scenario may now be invalid.`
   `Possible reason is access violation.`
   Because of some file access limitations GREAT-ER was not able to delete the scenario data completely. Probably one of the files in the directory is still opended by another program. Please close the file and remove remaining scenario data by deleting the scenario directory manually.

7. `Creation of export file xxx failed.`
   Because of some access limitiations the automatic generation of the export file is not possible. You might try to change the access restrictions and export again or to export the scenario manually: as all scenario data are stored in the related directory under the GREAT-ER scenario directory the export file is a zip archive of this subdirectory. You can create this archive with you favorite zip tool.

8. `Importing of file xxx failed.`
   Because of some access limitiations the automatic unpacking of the export file is not possible. You might try to change the access restrictions and import again or to import the scenario manually: as all scenario data are stored in the related directory under the GREAT-ER scenario directory the export file is a zip archive of this subdirectory. You can unpack this archive with you favorite unzip tool to the scenario directory.

9. `Neither CAS nor Name specified.`
   The last edits have been lost because the data set has no CAS or name, which are the minimum requirements to identify a substance data set.

10. `Not able to open database - keeping previous.`
    Two reasons are possible here: Either, due to access limitations, the database is not accessible or the database has somehow been deleted, which should not be the case.

11. `River network theme missing. Function not available.`
    The river network theme can not be found, which means it is no longer a theme of the active view or it has been renamed. The easy-to-use way to fix both is to save and close the scenario and reopen it. When using the expert mode the problem can be solved by adding the river network theme to the active view or by renaming it to ”River Net” in the Theme/Properties window.

12. `Discharge sites theme missing. Function not available.`
    The discharge site theme can not found, which means it is no longer a theme of the active view or it was renamed. The easy-to-use mode way to fix both is to save and close the scenario and reopen it. When using the expert mode the problem can be solved by adding the rivernet theme to the active view or by renaming it again to ”River Net” in the Theme/Properties window.

13. `Discharge table is not editable.`
    This is due to access restrictions. To enable the discharge table to become editable the access rights need to be changed.

14. `xxx : can not be opened for writing.`

This is due to access restrictions and can be solved by changing them. This ”warning” involves no loss of data.

15. `The table of simulation results is not editable.`
    `Calculation of background concentration xxx aborted.`
    This is due to access restrictions and can be solved by changing them.

16. `Csim theme feature table has errors.`
    One of the fields QMean, Reallength, VMean or PECintMean is missing in the theme's feature table, probably due to the fields having been renamed manually. This edit need to be undone. A second possibility may be that the field is hidden. This can be fixed by unhiding the field within the Expert mode.

17. `The table of simulation results is not editable.`
    `Calculation of xxx-percentile aborted.`
    This is due to access restrictions and can be solved by changing them.

18. `The theme's legend is not as required for PEC's. Profile plotting aborted.`
    The current legend classification is based on more than one field. This is not possible for profile plotting. Switch to the one field which should be used for plotting the profile.

19. `Background data table not found.`
    The table that contains information about additional background data can not be found. This indicates that the catchment installation routine did not work correctly or the table has been deleted or renamed manually. The file *`backgroundlst.dbf`* needs to be in the directory `%GHOME%\catchments\<CATCHMENT>`.

20. `Background data list could not be created.`
    This is an internal AVENUE error which should not normally occur. Nothing has to be done and no data can get lost. Just try to reload the background data.

21. `DCW coverage can not be opened.`
    One or more of the coverages pppoly, pppoint, rdline, rrline can not be opened. Check whether the coverages are in the directory `%GHOME%\overview\<COUNTRY>`. This may be the case if a scenario with a catchment from a new country has been included. In this case the DCW data need to be included in the directory `%GHOME%\overview\<COUNTRY>`.

22. `Your scenario data is incomplete: no model parameters.` You have to edit the model parameters to specify all necessary ones, This warning is caused by cancelling the model parameter dialog during the creation of a new scenario.

23. `Picture of xxx (yyy) not found.`

    The image file of the picture can not be found. It may be deleted or renamed. Go into the expert mode and check the database by loading the attribute table of the Site Pictures theme (via Themes/Table...) and then looking whether the image file name corresponds to a file in the directory `%GHOME%\graphics\sitepics`. If there are some inconsistencies, eliminate them.

24. `Not able to save scenario in xxx - directory already exists.`

    As saving is not possible to a directory of another scenario, go back to the ”Save As” dialog and specify a nonexisting directory. It will be created automatically.

25. `Not able to create directory`

    This is due to access restrictions and can be solved by changing them.

6.4 Errors

Errors indicate that something fundamental has gone wrong. An error may cause serious consequences, especially loss of data. But usually an error message indicates that something previous has failed or was deleted. Hence a GREAT-ER error message is caused by an earlier error.

1. `Banner file xxx/graphics/greater.gif not found!`
   This is usually caused by an impossible setting for the system variable %**GHOME**%. Check if %**GHOME**% is pointing to your GREAT-ER installation (by using the system settings); if not, set it correctly.

2. `The scenario directory named by GSCENARIOS (xxx) does not exist!`
   `Please create the directory and restart GREAT-ER.`
   Create the directory pointed to by %**GSCENARIOS**% (as indicated in the system settings) and restart GREAT-ER.

3. `Cannot get file xxx!`
   Without this file the execution of GREAT-ER is not possible. The problem may in some cases be solved by changing access restrictions. Otherwise the file can be read from the CD by extracting it from the file *`greater.zip`* on the CD into the directory `%GHOME%\ave`. This message also arises when the set-up has been performed manually using an old version of Zip that cuts the filenames to the old 8.3 convention. In this case, use the set-up program (recommended) or a newer version of Zip.

4. `Cannot get script load procedure! This makes it impossible to launch GREAT-ER.`
   This indicates that the script *`ReloadScript.ave`* is not loadable. The problem may in some cases be solved by changing access restrictions. Otherwise *`ReloadScript.ave`* can be read from the CD by extracting it from the file *`greater.zip`* on the CD into the directory `%GHOME%\ave`. This message also arises when the set-up has been performed manually using an old version of Zip that cuts the filenames to the old 8.3 convention. In this case, use the set-up program (recommended) or a newer version of Zip.

5. `Cannot compile load procedure! This makes it impossible to launch GREAT-ER.`
   This indicates that the script has errors, i.e. it has somehow been modified. It needs to be reinstalled by either reinstalling the whole GREAT-ER software or by installing, i.e. extracting, only the scripts that contain errors from the file *`greater.zip`* on the CD into the directory `%GHOME%\ave`.

6. `Error occurred opening script file xxx!`
   The script could not be opened. The problem may in some cases be solved by changing access restrictions. Otherwise *`ReloadScript.ave`* can be read from the CD by extracting it from the file *`greater.zip`* on the CD into the directory `%GHOME%\ave`. This message also arises when the set-up has been performed manually using an old version of ZIP that cuts the filenames to the old 8.3 convention. In this case, use the set-up program (recommended) or a newer version of ZIP.

7. `Error occurred compiling script xxx!`
   This indicates that the script has errors, i.e. it has somehow been modified. It needs to be reinstalled by either reinstalling the whole GREAT-ER software or by installing, i.e. extracting, only the scripts that contain errors from the file *`greater.zip`* on the CD into the directory `%GHOME%\ave`.

8. `No catchment database found!`
   The essential file *`catchments.dbf`* is missing. You have to reinstall GREAT-ER or only the file by extracting it from the file *`greater.zip`*, stored on the CD, to the directory `%GHOME%\catchments`.

9. `Internal Error: StretchID field missing!`
   The field to identify river stretches in the simulation result table is missing. Either the field name has been changed manually (it has to be 'stretchid') or the data is seriously corrupt. You should start a new simulation using the same parameters.

10. `Catchment CCC can not be opened!`
    The basic catchment data or parts of it are missing. There are three possibilities for this:

    (a) You imported a scenario for a catchment you do not have the data for. Check your installation and contact the party from which you obtained the scenario to update your catchment database.

    (b) Your catchmant database (under your GREAT-ER installation in the directory 'catchments' and its subdirectories) has the wrong read access restrictions. Change these settings by yourself if possible or ask your system administrator.

    (c) Your catchment database is seriously corrupt. Try to restore the catchment data by reinstalling GREAT-ER over your existing installation.

    The essential basic catchment coverages are: catchbound, disch_river, rivernet, discharges.

11. `Not able to get information about scenario!`
    The communication between this dialog and the GREAT-ER core application has failed. ARCVIEW has run into an unstable mode. It is recommended to leave GREAT-ER immediately without saving and to restart it.

12. `Substance not found! Database seems to be corrupt!`
    Although the substance you have selected is listed in the database there seems to be no data for it. This suggests that the substance's data set, or more generally the substance database itself, is damaged.

13. `The result theme data structure seems to be corrupt!`

The result theme data structure seems to be corrupted. You should try to recreate the scenario and restart the same simulation. If the problem should occur again, one of the needed idfields are missing ...

14. `Can not open 'scenario.odb'!`
    The basic scenario settings are not accessible. This may be caused by the wrong access restrictions, which can be fixed by changing them to read/write permission, or by a loss of data. In the latter case you need to reconstruct the scenario and save it again.

15. `Wrong number of items in 'scenario.odb'!`
    The basic scenario settings seem to be damaged. This may imply a loss of data. You need to reconstruct the scenario and save it again, preferably using a different name.

16. `No view for active scenario!`
    This problem should not normally occur. It indicates that the system has become unstable. You should therefore leave GREAT-ER without saving.

17. `Active scenario not found!`
    This problem should not normally occur. It indicates that the system has become unstable. You should therefore leave GREAT-ER without saving. Simulation results are stored under your scenario directory in the files 'pec.dbf' and 'pecstp.dbf'.

18. `Scenario data seems to be corrupt!`
    The scenario data structure seems to be seriously corrupt. Saving is no longer possible. Simulation results which you might want to keep are stored under your scenario directory in the files 'pec.dbf' and 'pecstp.dbf'. Exit the scenario without saving.

19. `No catchment database found!`
    The essential file *`catchments.dbf`* is missing. You have to reinstall GREAT-ER or only the file by extracting it from the file *`greater.zip`*, stored on the CD, to the directory `%GHOME%\catchments`.

20. `xxx table fields missing!  Function not available.`
    The table seems to be corrupt or has been edited manually. You have to reinstall it by extracting it (a *.dbf file) from the file *`greater.zip`*, stored on the CD, to the appropriate location in your installation (see Table/Properties for the correct location).

21. `QueryString for xxx table has errors!`
    A wrong query string suggests impossible field names and therefore a corrupt table. You have to reinstall the table by extracting it (a *.dbf file) from the file *`greater.zip`*, stored on the CD, to the appropriate location in your installation (see Table/Properties for the correct location).

22. `More than one entry for WWTP! WWTP data table seems to be corrupt.`
    The query to obtain the wwtp data from the database resulted in more than one data record for one wwtp. It may be fixed by manually deleting one of the entries, i.e. by editing the table WWTP data in the Expert mode.

23. `Can not find field fff in xxx table! Simulation aborted.`
    The table seems to be corrupt or has been edited manually. You have to reinstall it by extracting it (a *.dbf file) from the file *`greater.zip`*, stored on the CD, to the appropriate location in your installation (see Table/Properties for the correct location).

24. `Required parameter ppp not found in xxx dataset! Simulation aborted.`
    The dataset seems to be corrupt or has been edited manually. You have to reinstall it by extracting it from the file *`greater.zip`*, stored on the CD, to the appropriate location in your installation.

25. `Simulation results: the Csim xxx field was not found!`
    This error should never occur. It suggests that the simulation table has an incorrect structure. If this happens, just delete these simulation results, save the scenario and restart GREAT-ER to repeat your simulation.

26. `Csim theme feature table not found!`
    This error should never occur. It suggests that the simulation table has an incorrect structure. If this happens, just delete these simulation results, save the scenario and restart GREAT-ER to repeat your simulation.

27. `Can not find table with STP Csims!`
    The scenario data structure seems to be corrupt. This might have been caused by a loss of data or by renaming the result tables. You can check this by looking to see whether the file *`CsimSTP.dbf`* is in the scenario's directory. The easiest way to fix this is to repeat the simulation.

28. `Discharge record not found!`
    The database seems to be corrupt or has been changed manually. You have to reinstall it by extracting it from the file *`greater.zip`*, stored on the CD, to the appropriate location in your installation.

29. `Background data table seems to be corrupt!`
    One of the essential fields ”Name”, ”MapName”, ”Type”, ”Legend” has not been found. This means the installation is either corrupt or was changed manually. A reinstallation from the CD is necessary.

30. `Necessary script for data set xxx not found!`
    The background data table indicates a special script to be used for displaying the selected data. The script is not loadable, i.e the selected background data can not be displayed. The necessary script is given in the file *`backgroundlst.dbf`* in the catchment-specific subdirectory of the *`overview`* directory. The table can be added in the Expert mode to look (and then search for) the script.

31. `Error occurred compiling display script xxx!`
    The background data table indicates a special script to be used for displaying the selected data. The script seems to have some errors hence compiling (which is previous to executing) has failed. This makes it impossible to visualise the selected background data automatically. The file must be edited by an AVENUE programmer to fix the error.

32. `Background data information not complete!`
    The background information table *`backgroundlst.dbf`* is not complete. Refer to the technical manual for the required structure, and then look to see whether this is the same for the table which is stored in the catchment-specific subdirectory of the *`overview`* directory.

33. `Unsupported background data format!`
    The background data is of a type not considered in the current implementation. The data can therefore not be displayed. Supported types are point, arc, polygon, shape, grid and image.

34. `Cannot make xxx theme!`
    The background theme could not be loaded. Check whether the name and the path given in the background information table *`backgroundlst.dbf`* are valid. If not, change them (in the Expert-mode) or, more easily, make a complete reinstallation. First, check whether you have read access to the data.

35. `Background data not loadable!`
    ODB: either access problems, or a file is missing or corrupt.
    Parts of the background data you selected are missing or unreadable. First check if data is available and change access restrictions if necessary. If the data is missing, try to reinstall GREAT-ER over your existing installation to restore the data. If the data is not part of the original GREAT-ER package, contact the party you received the data from.

36. `xxx table not editable!  Analysis aborted.`
    This is usually due to access restrictions. You require write access in the directory named by your <**TEMP**> system variable.

37. `Picture theme missing!  Function not available.`
    The picture theme is not in the scenario dictionary structure. This implies a corrupt scenario. Create a new scenario using the same catchment. If the error persists, you need to reinstall GREAT-ER over your old installation.

38. `Picture theme seems to be corrupt!  Function not available.`
    Essential fields are missing, due to corrupt or manually changed data. If the catchment was part of the CD, reinstall GREAT-ER over your old installation. If not, contact the catchment's data provider to enquire how to proceed.

39. `Not able to save scenario.odb`

    This is caused by a missing write access in your scenario directory, i.e. a subdirectory under the directory named by your <**GSCENARIOS**> system variable.

40. `Not able to write xxx.  Scenario may be corrupt now!`

    This is caused by a missing write access. It may have caused a loss of data.

41. `No subcatchment selected in the active scenario.`

    Although the flags report a subcatchment for the active scenario the required information can not be found in the data structures. This suggests the scenario data are damaged. Try to recreate the scenario using a new name.