

RASS¹ 1.1 User's Guide

EPC-Version

Jürg Thöny²

1 INTRODUCTION	1
2 DISTRIBUTION AND INSTALLATION	1
3 RUNNING MDPs WITH RASS	2
3.1 Developing MDPs.....	2
3.2 Transferring MDPs to the RASS host.....	2
3.3 Building executable simulation programs.....	2
3.4 Running executable simulation programs.....	3
4 TUTORIAL	4
4.1 Developing Sensitivity MDP.....	4
4.2 Transferring the MDP Sensitivity to the RASS host.....	5
4.3 Building the executable simulation program Sensitivity.....	6
4.4 Running the executable simulation program Sensitivity.....	7
5 TROUBLE SHOOTING	9
6 LITERATURE	10

¹ RAMSES Simulation Server

² Systems Ecology Group, Institute of Terrestrial Ecology, ETHZ, Grabenstrasse 3, CH-8952 Schlieren, Switzerland

1 Introduction

This user's guide describes how to install and use RASS on Unix workstations using the EPC Modula-2 compiler.

This text assumes that you are familiar with ModelWorks (Fischlin et al. 1994). It is highly recommended to read also (Fischlin 1991) and (Thoeny et al. 1994). You should also be able to work with Unix workstations.

RASS allows to run interactive ModelWorks Model Definition Programs [MDPs] in batch oriented environments like Unix workstations.

The generated results of a batch simulation should be the same as on the interactive RAMSES host, but you have to consider the following aspects:

- 1) The calculated results don't have to be numerical identical. This is due to the different floating point hardware on different hosts. Especially simulations with very small step-sizes may produce detectable differences. It is impossible to say on which host the quality of the results is better.
- 2) RASS ModelWorks is build on top of the Batch-DM (Thoeny et al. 1994). No user interactions are possible. A RASS program will follow the default execution thread; it simulates default answers on every interactive DM component (i.e. pressing the enter key in every modal dialogue).

2 Distribution and installation

RASS consists of the libraries *libMWLib.a*, *libMWLib_x.a* and the shell scripts *MOD2mod*, *DEF2def*, *listdef*, and *RASSMakeMake*. You have to install the libraries in your linker search path and the shell scripts in the command search path of your shell. It is recommended to install the RASS library in */usr/local/lib* and the shell scripts in */usr/local/bin*. Please ask your system administrator to install these files.

If the installation in */usr/local* isn't possible, you can also install RASS local to your home directory. After copying the file *libMWLib.a* to the desired location, you have to run:

```
ranlib PATH_OF_LIB/libMWLib.a PATH_OF_LIB/libMWLib_x.a
```

To allow the linker to find the library, you have to specify the location using the *LD_LIBRARY_PATH* environment variable. Please make sure, that the RASS shell scripts are in the command search path of your shell (consult the documentation of your favoured shell).

The EPC Modula-2 environment has to be installed on the host prior using RASS. This has to be done by your system administrator.

3 Running MDPs with RASS

A simulation program is usually developed in the interactive RAMSES environment. Then the Modula-2 source code and the input files are transferred to the RASS host. There you can build an executable simulation program to finally run your simulation experiments.

3.1 Developing MDPs

Usually a MDP is developed on an interactive RAMSES host, where you should also plan and test your simulation experiment. Beside the default strategy of the Batch-DM (default answers to dialogues), RASS will simulate exactly one user event for you:

If you have an experiment installed in your MDP, RASS will execute it, otherwise RASS will execute one simulation run.

Therefore, you can test your simulation program on the interactive RAMSES host prior transferring it to the RASS host. If you start your simulation respectively experiment and get the desired result by answering all eventual modal dialogues just by pressing the enter key, you can expect to get the same behaviour on the RASS host.

3.2 Transferring MDPs to the RASS host

A developed MDP consists of one or several Modula-2 source files and optional data files. They have to be transferred to the RASS host. Because most hosts have a different definition of text files, it is highly recommended to use FTP (File Transfer Protocol) to transfer files, since it translates them according to the specification of the target host.

The EPC Modula-2 compiler expects source files with the extension `.mod` and `.def`. All modules must have the name `Modulename.mod` or `Modulename.def` for definition modules. The naming is case sensitive. If you transfer files from the RAMSES environment, they are usually named using the extensions `.MOD` and `.DEF`. The two shell scripts `MOD2mod` and `DEF2def` performs this renaming for you. To use these scripts, change to the directory where your source files reside and call both scripts.

3.3 Building executable simulation programs

After you have transferred your source and data files to the RASS host, you can build an executable simulation program. This involves both, the compilation of the sources and the linking with the MW library. The shell script `RASSMakeMake` helps you with this task. Run `RASSMakeMake` in the directory where your MDP resides. It will generate a file named `Makefile`. The Unix tool `make` uses this file to perform the compilation and linking for you. Simply call `make` in the directory where your MDP and the generated Makefile reside. Note, that `RASSMakeMake` assumes, that the first (alphabetically) module without corresponding definition module is the main module.

It is possible that you get some compiler error messages. The reason is usually that Modula-2 is not defined exactly the same on various compiler implementations. Please read the report "Practical considerations on writing portable Modula-2 code" (Thoeny 1994) to get more information about writing portable Modula-2 code.

If you change the source code, you have to call *make* again in order to bring your executable simulation program up do date.

You can use *make* with three optional parameters: *all*, *clean*, and *depend*. *make all* is the same as *make* without parameter. *make clean* will remove all object files (the result of the compilation process). *make depend* will add the inter module dependencies to the *Makefile*. This is useful, if you plan to change your definition modules on the RASS host.

3.4 Running executable simulation programs

The executable simulation program has the same name as the main module of your MDP. It can be started by typing its name.

If you have stored your data files in another location as the directory where you start your executable simulation program, you have to specify the path in the environment variable M2PATH. Consult the documentation of your favoured shell on how to set environment variables. The syntax of M2PATH is:

```
path[:path]
```

where [:path] stands for an optional repetition of :path. The pathes can be given relative to the start directory or to the root directory.

Examples using csh:

```
setenv M2PATH Datafiles
setenv M2PATH Datafiles1:Datafiles2
setenv M2PATH /home/users/goofy/RASS/Models/MyModel/Datafiles
```

If you have used DM library routines which are not yet implemented in the Batch-DM, a file named *RASS.NYI* will be created. *RASS.NYI* contains a list of all calls to those routines. Each line contains one entry. An entry can have two forms:

```
NotYetImplemented : RoutineName in ModuleName
FATAL NotYetImplemented : RoutineName in ModuleName
```

RASS.NYI contains any number of entries of the first form. If an entry of the second form exists, the program was aborted inside the listed routine. Note that the file *RASS.NYI* will only be generated if you have called at least one of the routines, which are not yet implemented. After execution of a simulation program, you should check the existence of *RASS.NYI*. If it exists, examine its content. In order to prevent the examination of an old *RASS.NYI*, it is recommended to remove or rename it prior to the next execution of a RASS program.

The filing of the MW library on RASS behaves the same as on the interactive RAMSES. As a result, you will only get a stash file if at least one monitoriable variable is activated for filing, or if you switch the filing on by calling *SimBase.SetDocumentRunAlwaysMode*.

4 Tutorial

This tutorial guides you through a sample RAMSES to RASS transition. The transfer using FTP is not described here.

The chosen sample model is Sensitivity (File name Sensitivity.MOD) as described in (Fischlin et al. 1994). It demonstrates a parameter sensitivity analysis of a Michaelis-Menten algae growth model. It creates an additional window, where it writes the chosen parameter sets.

At the begin, Sensitivity reads its model parameters from the data file Sensitivity.DAT.

4.1 Developing Sensitivity MDP

The model is already developed and produces the following output when ran from within RAMSES (On a Macintosh, after some window rearrangement):

Parameter Sets					
3	1	1:	μ_{max}	= 1.226,	Ks = 13.000, x0 = 1000.00, Run Nr. = 1
3	2	1:	μ_{max}	= 1.226,	Ks = 13.000, x0 = 2500.00, Run Nr. = 2
3	3	1:	μ_{max}	= 1.226,	Ks = 13.000, x0 = 5000.00, Run Nr. = 3
3	1	2:	μ_{max}	= 1.226,	Ks = 845.000, x0 = 1000.00, Run Nr. = 4
3	2	2:	μ_{max}	= 1.226,	Ks = 845.000, x0 = 2500.00, Run Nr. = 5
3	3	2:	μ_{max}	= 1.226,	Ks = 845.000, x0 = 5000.00, Run Nr. = 6
3	1	3:	μ_{max}	= 1.226,	Ks = 1500.00, x0 = 1000.00, Run Nr. = 7
3	2	3:	μ_{max}	= 1.226,	Ks = 1500.00, x0 = 2500.00, Run Nr. = 8
3	3	3:	μ_{max}	= 1.226,	Ks = 1500.00, x0 = 5000.00, Run Nr. = 9
3	1	1:	μ_{max}	= 2.000,	Ks = 13.000, x0 = 1000.00, Run Nr. = 10
3	2	1:	μ_{max}	= 2.000,	Ks = 13.000, x0 = 2500.00, Run Nr. = 11
3	3	1:	μ_{max}	= 2.000,	Ks = 13.000, x0 = 5000.00, Run Nr. = 12
3	1	2:	μ_{max}	= 2.000,	Ks = 845.000, x0 = 1000.00, Run Nr. = 13
3	2	2:	μ_{max}	= 2.000,	Ks = 845.000, x0 = 2500.00, Run Nr. = 14
3	3	2:	μ_{max}	= 2.000,	Ks = 845.000, x0 = 5000.00, Run Nr. = 15
3	1	3:	μ_{max}	= 2.000,	Ks = 1500.00, x0 = 1000.00, Run Nr. = 16
3	2	3:	μ_{max}	= 2.000,	Ks = 1500.00, x0 = 2500.00, Run Nr. = 17
3	3	3:	μ_{max}	= 2.000,	Ks = 1500.00, x0 = 5000.00, Run Nr. = 18
3	1	1:	μ_{max}	= 3.174,	Ks = 13.000, x0 = 1000.00, Run Nr. = 19
3	2	1:	μ_{max}	= 3.174,	Ks = 13.000, x0 = 2500.00, Run Nr. = 20
3	3	1:	μ_{max}	= 3.174,	Ks = 13.000, x0 = 5000.00, Run Nr. = 21
3	1	2:	μ_{max}	= 3.174,	Ks = 845.000, x0 = 1000.00, Run Nr. = 22
3	2	2:	μ_{max}	= 3.174,	Ks = 845.000, x0 = 2500.00, Run Nr. = 23
3	3	2:	μ_{max}	= 3.174,	Ks = 845.000, x0 = 5000.00, Run Nr. = 24
3	1	3:	μ_{max}	= 3.174,	Ks = 1500.00, x0 = 1000.00, Run Nr. = 25
3	2	3:	μ_{max}	= 3.174,	Ks = 1500.00, x0 = 2500.00, Run Nr. = 26
3	3	3:	μ_{max}	= 3.174,	Ks = 1500.00, x0 = 5000.00, Run Nr. = 27

4.2 Transferring the MDP Sensitivity to the RASS host

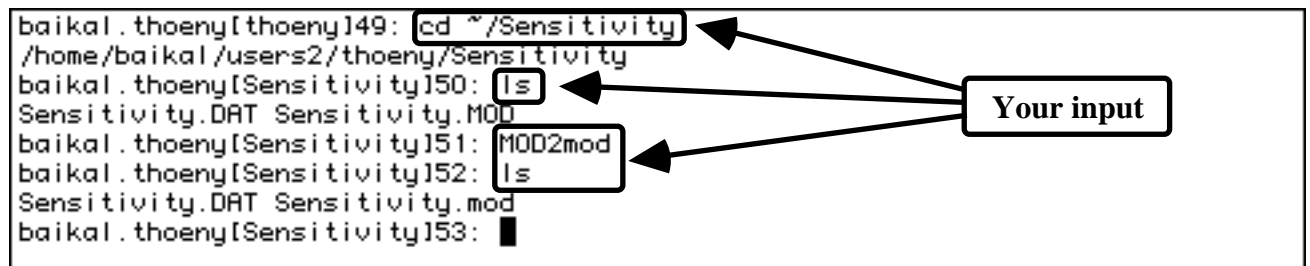
Create a directory named Sensitivity in your home directory on the RASS host:

```
mkdir ~/Sensitivity
```

Transfer Sensitivity.MOD and Sensitivity.DAT from your RAMSES host to the RASS host into this directory using FTP.

On the RASS host change to the directory Sensitivity, check for the presence of the MDP and data file, and call the shell script *MOD2mod*:

The output on the terminal should look like:



```
baikal.thoeny[thoeny]49: cd ~/Sensitivity
/home/baikal/users2/thoeny/Sensitivity
baikal.thoeny[Sensitivity]50: ls
Sensitivity.DAT Sensitivity.MOD
baikal.thoeny[Sensitivity]51: MOD2mod
baikal.thoeny[Sensitivity]52: ls
Sensitivity.DAT Sensitivity.mod
baikal.thoeny[Sensitivity]53: █
```

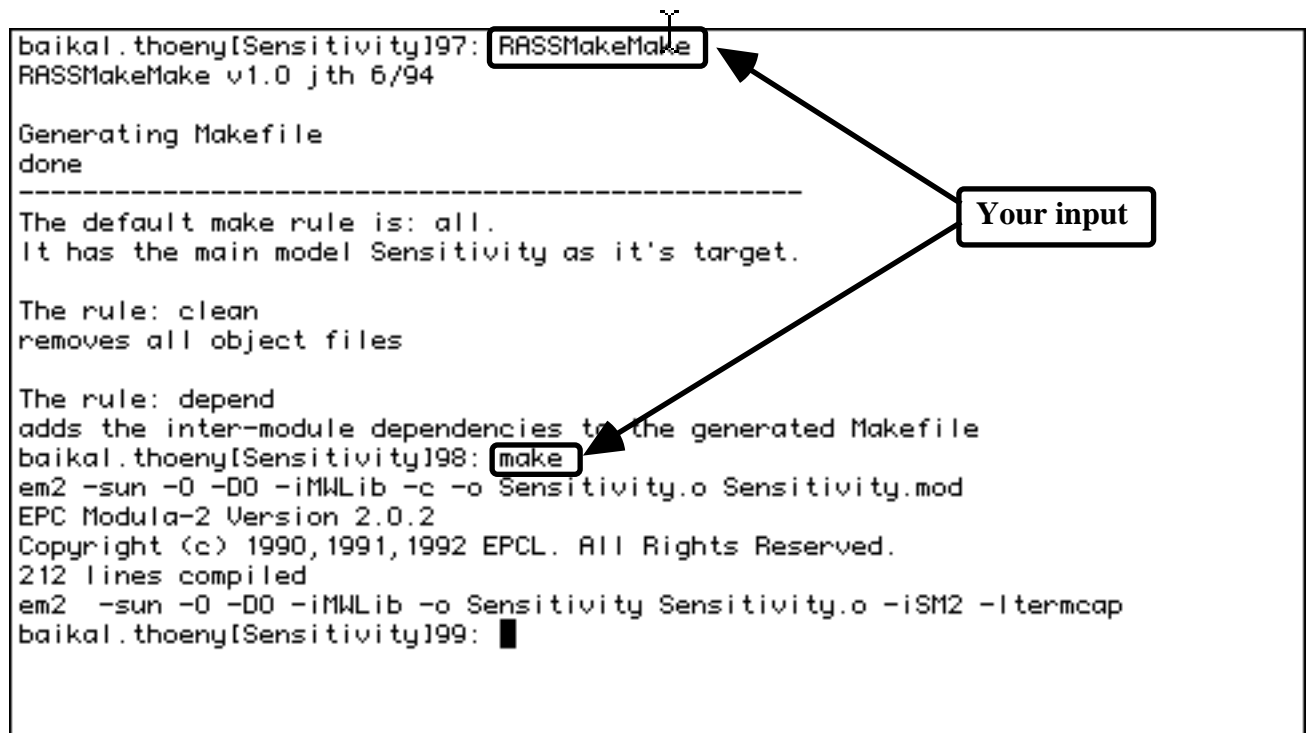
You don't have to call *DEF2def* because the Sensitivity MDP contains no definition modules.

4.3 Building the executable simulation program Sensitivity

The shell script RASSMakeMake generates the Makefile, which you can subsequently use to build Sensitivity. In the directory ~/Sensitivity use the commands:

```
RASSMakeMake
make
```

The output on the terminal should look like:



```
baikal.thoeny[Sensitivity]197: RASSMakeMake
RASSMakeMake v1.0 jth 6/94

Generating Makefile
done
-----
The default make rule is: all.
It has the main model Sensitivity as it's target.

The rule: clean
removes all object files

The rule: depend
adds the inter-module dependencies to the generated Makefile
baikal.thoeny[Sensitivity]198: make
em2 -sun -O -DO -iMWLib -c -o Sensitivity.o Sensitivity.mod
EPC Modula-2 Version 2.0.2
Copyright (c) 1990, 1991, 1992 EPCL. All Rights Reserved.
212 lines compiled
em2 -sun -O -DO -iMWLib -o Sensitivity Sensitivity.o -ISM2 -ltermcap
baikal.thoeny[Sensitivity]199: █
```

The diagram shows a terminal window with a box labeled "Your input" on the right. Two arrows point from this box to the "RASSMakeMake" and "make" commands in the terminal output.

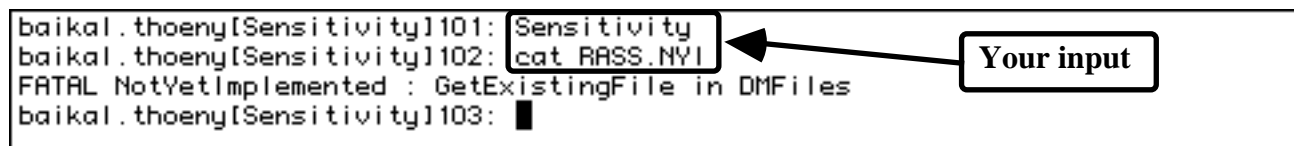
4.4 Running the executable simulation program Sensitivity

Now you can run Sensitivity which will be finished very fast. Then analyse the RASS.NYI:

```

baikal.thoeny[Sensitivity]101: Sensitivity
baikal.thoeny[Sensitivity]102: cat RASS.NYI
FATAL NotYetImplemented : GetExistingFile in DMFiles
baikal.thoeny[Sensitivity]103: █

```



Whoops! This transition went wrong. The best thing at this point is, to return to your RAMSES host and look at the MDP. You will find GetExistingFile twice; first in the import list,

```

FROM DMFiles IMPORT GetExistingFile, TextFile, GetReal, SkipGap,
  ReadChars, Close, Response;

```

and secondly inside the procedure ReadAndSetParameters.

```

GetExistingFile(parFile, 'Open parameter file "Sensitivity.DAT"');

```

As you can see, the call to GetExistingFile has no parameter for a default file name. Therefore, RASS was unable to continue and terminated your program. In order to prevent this, we replace GetExistingFile with DMFiles.Lookup in the import list,

```

FROM DMFiles IMPORT Lookup, TextFile, GetReal, SkipGap,
  ReadChars, Close, Response;

```

and inside the procedure ReadAndSetParameters.

```

Lookup(parFile, "Sensitivity.DAT", FALSE);

```

Now we can test our changes on RAMSES and transfer the Sensitivity.MOD again to the RASS host. There we have to rerun the shell script *MOD2mod*. We don't have to rerun RASSMakeMake, since the structure of the MDP hasn't changed.

Rerun make, which builds us finally the new executable simulation program of Sensitivity.

An other run of Sensitivity gives us the following output, which happens to be exactly the same as on RAMSES:

```

baikal.thoeny[Sensitivity]124: Sensitivity
3 1 1:  $\mu_{max}$  = 1.226, Ks = 13.000, x0 = 1000.00, Run Nr. = 1
3 2 1:  $\mu_{max}$  = 1.226, Ks = 13.000, x0 = 2500.00, Run Nr. = 2
3 3 1:  $\mu_{max}$  = 1.226, Ks = 13.000, x0 = 5000.00, Run Nr. = 3
3 1 2:  $\mu_{max}$  = 1.226, Ks = 845.000, x0 = 1000.00, Run Nr. = 4
3 2 2:  $\mu_{max}$  = 1.226, Ks = 845.000, x0 = 2500.00, Run Nr. = 5
3 3 2:  $\mu_{max}$  = 1.226, Ks = 845.000, x0 = 5000.00, Run Nr. = 6
3 1 3:  $\mu_{max}$  = 1.226, Ks = 1500.00, x0 = 1000.00, Run Nr. = 7
3 2 3:  $\mu_{max}$  = 1.226, Ks = 1500.00, x0 = 2500.00, Run Nr. = 8
3 3 3:  $\mu_{max}$  = 1.226, Ks = 1500.00, x0 = 5000.00, Run Nr. = 9
3 1 1:  $\mu_{max}$  = 2.000, Ks = 13.000, x0 = 1000.00, Run Nr. = 10
3 2 1:  $\mu_{max}$  = 2.000, Ks = 13.000, x0 = 2500.00, Run Nr. = 11
3 3 1:  $\mu_{max}$  = 2.000, Ks = 13.000, x0 = 5000.00, Run Nr. = 12
3 1 2:  $\mu_{max}$  = 2.000, Ks = 845.000, x0 = 1000.00, Run Nr. = 13
3 2 2:  $\mu_{max}$  = 2.000, Ks = 845.000, x0 = 2500.00, Run Nr. = 14
3 3 2:  $\mu_{max}$  = 2.000, Ks = 845.000, x0 = 5000.00, Run Nr. = 15
3 1 3:  $\mu_{max}$  = 2.000, Ks = 1500.00, x0 = 1000.00, Run Nr. = 16
3 2 3:  $\mu_{max}$  = 2.000, Ks = 1500.00, x0 = 2500.00, Run Nr. = 17
3 3 3:  $\mu_{max}$  = 2.000, Ks = 1500.00, x0 = 5000.00, Run Nr. = 18
3 1 1:  $\mu_{max}$  = 3.174, Ks = 13.000, x0 = 1000.00, Run Nr. = 19
3 2 1:  $\mu_{max}$  = 3.174, Ks = 13.000, x0 = 2500.00, Run Nr. = 20
3 3 1:  $\mu_{max}$  = 3.174, Ks = 13.000, x0 = 5000.00, Run Nr. = 21
3 1 2:  $\mu_{max}$  = 3.174, Ks = 845.000, x0 = 1000.00, Run Nr. = 22
3 2 2:  $\mu_{max}$  = 3.174, Ks = 845.000, x0 = 2500.00, Run Nr. = 23
3 3 2:  $\mu_{max}$  = 3.174, Ks = 845.000, x0 = 5000.00, Run Nr. = 24
3 1 3:  $\mu_{max}$  = 3.174, Ks = 1500.00, x0 = 1000.00, Run Nr. = 25
3 2 3:  $\mu_{max}$  = 3.174, Ks = 1500.00, x0 = 2500.00, Run Nr. = 26
3 3 3:  $\mu_{max}$  = 3.174, Ks = 1500.00, x0 = 5000.00, Run Nr. = 27
baikal.thoeny[Sensitivity]125:

```

RASS.NYI contains now:

NotYetImplemented : CreateWindow in DMWindows

Since the batch-DM was designed for non interactive usage, it is unable to create a window, but it redirects the text written to a window to the standard output file.

5 Trouble shooting

Symptom	Possible solutions
The linker doesn't find libMWLib.a or libMWLib_x.a	<p>The libraries are not in the linker search path. Eventually ask the local system administrator to put it into an accessible place.</p> <p>If the libraries are not in the linker search path (i.e. local to your home directory), use the environment variable <code>LD_LIBRARY_PATH</code> to tell the linker where it can find the library.</p>
The linker complains about an "out of date" symbol table of libMWLib.a.	Use ranlib to bring the symbol table up to date.
The executable simulation program doesn't find my data files.	<p>The file names are case sensitive on Unix systems. You may have to rename some files.</p> <p>If your data files aren't in the start-up directory, specify the search path with the environment variable <code>M2PATH</code>.</p>
A MDP file consists only of one line on my Unix workstation.	You should transfer all your MDPs using FTP in text (ASCII) mode.
I'm using Fetch as the FTP Client on my Macintosh computer, and it doesn't allow me to transfer my RAMSES MDPs in text mode.	<p>RAMSES on Macintosh computers sets the type of the MDPs to 'MoTx'. Fetch currently only allows the transfer of files with the type 'TEXT' in text mode. Use an other FTP client (e.g. XferIt sfalken@apple.com)</p> <p>It is also possible to set the preferences of the RAMSES shell (Modeling session, programming session, MiniShell) to "export mode". Thereafter the RAMSES shell will set the type of the touched MDPs to 'TEXT'.</p>

6 Literature

Fischlin, A. "Interactive modeling and simulation of environmental systems on workstations." In Dynamic Systems in Medicine, Biology, and Ecology, ed. D.P.F. Möller and O. Richter. 131-145. 275. Berlin: Springer, 1991.

Fischlin, A., D. Gyalistras, O. Roth, M. Ulrich, J. Thoeny, T. Nemecek, H. Bugmann, and F. Thommen. ModelWorks 2.2: An Interactive Simulation Environment for Personal Computers and Workstations. Systems Ecology, ETHZ, 1994. 14.

Thoeny, J. Practical considerations on portable Modula 2 code. Systems Ecology Group, ETHZ, 1994. Internal Report # ??

Thoeny, J., A. Fischlin, and D. Gyalistras. "RASS: Towards bridging the gap between interactive and off-line simulations." In CISS - First Joint Conference of International Simulation Societies in Zuerich, Switzerland, edited by W. Karplus and R. Rimane J. Halin, The Society for Computer Simulation International, P.O. Box 17900, San Diego, Cal. 92177, USA, 99-103, Year.