Coupling Animo with Fussim

Technical documentation of the coupled model running in the Fortran Simulation Environment version 4.4

C. Rappoldt, M. Heinen, P. Groenendijk

EcoCurves

EcoCurves rapport 9, ISSN 1872-5449

Coupling Animo with Fussim

In opdracht van Alterra BV

Coupling Animo with Fussim

Technical documentation of the coupled model running in the Fortran Simulation Environment version 4.4

C. Rappoldt[1], M. Heinen[2], P. Groenendijk[2]

[1]EcoCurves, Kamperfoelieweg 17, 9753 ER Haren, Nederland

[2]Alterra, P.O. Box 47, 6700 AA Wageningen, The Netherlands

E-mail: kees.rappoldt@ecocurves.nl

EcoCurves rapport 9

EcoCurves, Haren, 2008

REFERAAT

C. Rappoldt, M. Heinen, P. Groenendijk, 2008. *Coupling Animo with Fussim ; Technical documentation of the coupled model running in the Fortran Simulation Environment version 4.4*. EcoCurves rapport 9, EcoCurves, Haren. 34 blz. 3 fig.; 9 ref.

The model Animo for nutriënt management and nutriënt conversion processes in a one-dimensional soil column has been adapted for running under FSE, the Fortran SImulation Environment version 4.4. The FSE4 environment synchronizes the execution of coupled simulation models and facilitates the exchange of information between those models at appropriate moments of their respective update cycles. This allows Animo to be coupled with the model Fussim for two-dimensional transport of soil water. The two-dimensional grid of Fussim corresponds to a number of Animo columns placed side by side.

Trefwoorden: soil processes, nutrients, water, 2D, model coupling, fortran, simulation

This report is available as a PDF file at www.ecocurves.nl.

For questions concerning Animo and Fussim you should contact the owner of these models Alterra (P.O. Box 47; NL–6700 AA Wageningen; The Netherlands). The used tools FSE4, COMMAN, TTUTIL are maintained by and available from EcoCurves.

Kamperfoelieweg 17, 9753 ER Haren (gn), Nederland
Tel.: (050) 5370979; e–mail: kees.rappoldt@ecocurves.nl

[EcoCurves september 2008]

Contents

List of Figures

Woord vooraf

In het kader van een studie aan nutriënten in akkerranden bestond by Alterra de behoefte aan een koppeling tussen de in Fortran geschreven modellen Animo voor bodemprocessen en Fussim voor twee-dimensionaal watertransport. In dit rapport "Coupling Animo with Fussim" wordt de tot stand gebrachte koppeling beknopt gedocumenteerd.

Het uitgangspunt bij het koppelen van Animo en Fussim is dat slechts minimale ingrepen in de code van beide modellen worden gedaan. Zodoende kunnen beide modellen blijvend worden onderhouden door hun auteurs en kan het gekoppelde model ook worden aangepast aan toekomstige verbeteringen in de procesbeschrijving. Het voor de koppeling gebruikte gereedschap is FSE, de "Fortran Simulation Environment" waarvan versie 4 geschikt is voor het koppelen van modellen die sterk van elkaar verschillen in opbouw en integratiemethode.

De aanpak van het probleem is eerst uitvoerig besproken met Piet Groenendijk en Marius Heinen die daarom ook mede-auteur zijn van de documentatie in dit rapport. Tijdens de uitvoering heeft Marius Heinen de verantwoordelijkheid op zich genomen voor het omrekenen van Fussim uitvoer naar Animo invoer. Falentijn Assinck heeft testruns met het gekoppelde model uitgevoerd en tenslotte dank ik Leo Renaud voor zijn snelle reactie op vragen en verzoeken omtrent de Animo code.

Haren, juli 2007
Kees Rappoldt

Chapter 1

Introduction

1.1 Background

The model Animo (Groenendijk *et al.*, 2005) describes the nutrient dynamics for a soil compartment dependent on soil properties, water content and water flux. It makes use of water contents and fluxes calculated before with the 1D model for water transport SWAP. In this way, Animo simulates the nutrients for a soil column.

This is not sufficient, however, for describing large gradients at the edge of a field. If managament along the edge is different it has to be studied in close connection with the transports that will take place between the edge and the ditch and between the edge and the main part of the field. This explains the need for coupling Animo with a 2D model for water transport like Fussim (Heinen, 2001; de Willigen, 1998, 2001).

After an overwiew of the coupling approach, this report briefly describes the structure of the resulting program. The final source files of the coupled Animo have been compared with Animo 4.25. The list of source files used can be found in the Appendix. A listing of all differences between versiuon 4.25 and the coupled version is available as a separate document.

1.2 Approach

At first we considered to couple Animo to Fussim for each soil compartment seperately. That would lead to a real 2D version of Animo in which the 1D input from above and below is replaced by dynamic 2D fluxes calculated by Fussim. However, not just the core of the Animo model, also many procedures for input and output have been designed for a soil column and take into account boundary conditions along top and bottom compartments. Hence, the construction of a 2D Animo requires stripping of the current code and the formulation of a 0D model for a single compartment only, in combination with report generators for a collection of such compartments. This all has to be tested first in a 1D environment and then, finally,

a 2D Animo can be created. Essentially this means rewriting Animo.

We therefore kept intact current Animo and coupled it as a soil column model to Fussim. Copies of Animo ("model instances") are put side by side and together match the grid for which Fussim delivers the hydrology. A disadvantage of this approach is that horizontal transport had to be implemented in a somewhat artificial way. Further, the update order of the soil compartments will not be downstream[1] under all circumstances[2].

A considerable advantage of column wise coupling is that the present functionality of Animo is preserved. The coupling with Fussim may even be switched off and then, for each column, the model remains identical to the standalone version of the model.

The coupling between Animo and Fussim has been realized in a few steps. After each step the Animo model was still working and has been tested by comparing example output with results of the original model.

1 The Animo main has been split up into sections like "Initialize", "Update over a timestep", "New calendar year" and "Terminal". These sections are executed by means of calls from a higher level routine, in this case the FSE 4 integrator "integModelOwn". The model then runs in the FSE 4 simulation shell.

2 The READ statements for reading SWAP input from file are replaced by calls to a new SUBROUTINE Hydrology. This subroutine delivers precisely the same information as the original Read statements. In each call to Hydrology only those variables are requested that were formerly read from file. This can be realized in Fortran 95 by declaring the subroutine parameters OPTIONAL.

3 The array size settings of Animo in PARAM.INC and the constants in OUTBAL.INC are replaced by Fortran 90 modules which are accessed by means of USE statements.

4 The Animo variables in ANIMO.INC are placed in a module "AnimoInstances" and are declared as pointers. These pointers look like ordinary Fortran-77 scalars and arrays and no changes are necessary in the computing code of Animo. In fact, these pointers are associated with an allocated instance of the Animo variables. By means of a call to subroutine "SetModelInstance" the model can switch between the different instances.

5 The columns and rows of Fussim are grouped into columns and layers for Animo. The rules are defined in a small datafile "AnimoGeometry.dat" which is read by a new module FF. Using the geometry conversion in combination with unit conversions, subroutine FromFussim (called by Fussim) prepares, for each column of Animo, a complete list of "SWAP variables" at the end of the last shell timestep taken by Fussim.

6 The FSE 4 simulation shell is configured to start both Fussim and and the instances of Animo. This requires a small controller subroutine "AnimoFussimControl", which is called from the FSE4 simulation shell.

[1] This means in the direction of the water flow.

[2] In the rare case of horizontal water fluxes to the left *and* to the right at the boundary of a the same soil column, some compartments will see input concentrations from one ShellTimeStep ago.

7 A FussimMode is added to the Hydrology subroutine and in this mode Hydrology passes the information from Fussim to Animo instead of the variables read from SWAP files.

The last two steps realize the actual coupling.

Chapter 2

The coupled model

2.1 The FSE environment

The Fortran SImulation Environment (van Kraalingen, 1995) consists of subroutines and functions in a linked library. FSE has been created as an environment for crop simulation but has been extended for coupling models with very different structures or integration methods (Rappoldt & van Kraalingen, 2007). The new version also handles models with multiple instances.

The core routine RunCoupledModels simulates time from ShellStartTime to ShellFinishTime with steps ShellTimeStep. Each step it polls the controller (here: AnimoFussimControl) if there are any models or model instances that have to be started or terminated. Such requests result in a neat series of commands for the models. Running models and instances receive calls from FSE for output and update. An update call should result in some numerical procedure which drives the model to the next ShellTime. RunCoupledModels, however, does not know anything about integration procedures. Internally, models may use sophisticated adaptive timestep algortithms, as long as the new ShellTime is reached. Hence, the ShellTimeStep is just the timestep associated with the coupling.

If one of the models is not able to reach the new time, for instance, since a special event happens at time T, RunCoupledModels orders all models to restore their old status and then reduces its update time to T in a new attempt. Then, at time T event calls are carried out and the simulation continues. In Animo-Fussim the possibiliy of a system restore is not used since Fussim cannot handle it.

FSE 4 makes use of the Communication Manager in linked library COMMAN. COMMAN creates a blackboard on which models can store variables under a name and read variables stored by other models. One way on which models in the FSE environment may exchange information is by means of such PUT and GET calls in special sections of the models which are called by FSE at appropriate moments during the update cycle (see for details Rappoldt & van Kraalingen, 2007). In the Animo-Fussim coupled model this method is not used since Animo requires hydrological information *during* its update procedure. Therefore, instead of information exchange in bewteen update calls, Animo is acquires its hydrology by calling a special subroutine which buffers the information from Fussim.

The user of FSE does not need to change the library when new models are added to the system. Only trivial "Select routines" are required in which calls from FSE are distributed by means of statements like

```
select case (ModelName)
```

```
case ('ANIMO')
   call Animo (Task,ModelInstance,DataOwner)
case ('FUSSIMFSE4')
   call FussimFSE4 (Task,ModelInstance,DataOwner)
case default
   call FatalERR ('ModelSelect','unknown model '//TRIM(ModelName))
end select
```

Note that the variable Task contains the command from the FSE environment or from the integrator. The select routines required for the Animo-Fussim model are included in the file Main.f90.

2.1.1 Calendar

Simulation time starts at ShellStartTime and ends at ShellFinishTime *without a unit of time specified*. By means of some additional input the user may connect the simulated time to a calendar. This requires nothing more than the calendar time corresponding to ShellStartTime and a specification of the unit of time[1]. After doing so, models may do calls to the Calendar putting in the shell time they receive and getting out one of the optional Calendar output variables, including current year, month, etc. The calendar utility is an efficient and a highly precise tool. Its use is recommended[2].

2.2 The Animo Fussim model structure

The coupled model consists of a short main program, an FSE4 version of Animo, Fussim which was already running in the FSE environment, and the linked libraries FSE4, COMMAN, DEFSYS, WEATHER and TTUTIL. Figure 2.1 shows a scheme of the program structure. The function of each part is briefly described in the figure. Here a few additional comments are given. Only the Animo model routine is discussed extensivily.

2.2.1 The Pretty Standard Main

The main program initalizes the GET and PUT calls. These are calls to the Communication Manager in linked library COMMAN. COMMAN creates a blackboard on which models can store variables under a name and read variables stored by other models. One way on which models in the FSE environment may exchange information is by means of such PUT and GET calls in special sections of the models which are called by FSE at appropriate moments during the update cycle.

The main program allows the addition of tools for analyzing model results. Such tools could get their input from files (produced during the model execution), but also from the blackboard utility COMMAN to which models can PUT results as so called "End_Of_Run" values which are not destroyed by FSE on model temination.

[1] The unit of time is supplied as the length of one day in units ShellTime. For example OneDay=1.0 means that the unit of time is a day, Oneday=24.0 means that the unit of time is an hour, and OneDay=86400.0 means that the unit of time is a second.

[2] Calendar does take into account all the rules for leapyears that keep the seasons in place. Leapseconds, however, are not accounted for. These leapseconds correct for the slowly increasing length of the solar day and have been inserted a few times between december-31 24:00:00 and the new year beginning at Jan-1 00:00:00.

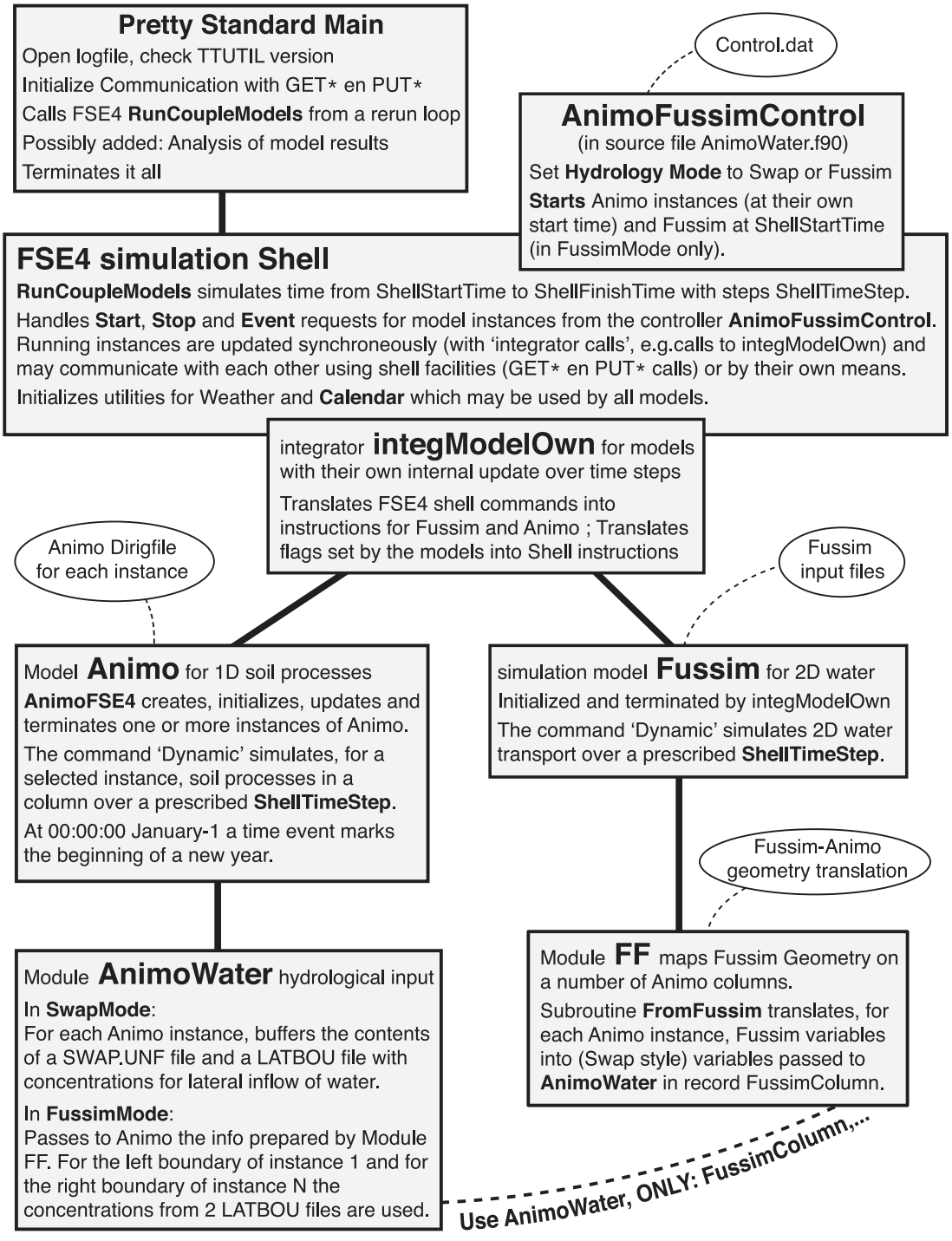

Figure 2.1. Scheme of the coupled model. The rectangular blocks are Fortran-90 subprograms or modules. The FSE4 simulation shell and the integrator integModelOwn are part of the FSE4 object library which has to be linked to the other parts of the program. Other linked libraries are the communication manager COMMAN, the weaher library WEATHER and the I/O utilitiy library TTUTIL. The dashed lines point to some datafiles.

2.2.2 The AnimoFSE4 module

The Fortran-95 module AnimoFSE4 contains a few variables and two subprograms: subroutine Animo which is the actual FSE model, and function AnimoOutXXXX which is used to get the output files of each instance in a seperate folder.

USE of other modules

The model routine Animo makes use of other Fortran-95 modules which are declared by means of USE statements in AnimoFSE4:

NRtype A public domain module from Numerical Recipes (Press *et al.*, 1992). NRtype defines the types for single and double precision floating point variables, integer and logical variables. These types are the same as the ordinary Fortran defaults, but they can be changed. Existing Animo code, however, has not been adapted to the use of NRtype. Without this adaptation the type definitions in NRtype should not be changed.

ABtype Defines string lengths used for input lines, filenames, etc.

logFile A module from the EcoCurves utility library for writing messages to a logfile and to screen. The source code of module logFile is provided. Options for the logging level can be set and screen output can be switched off. The original Animo logfile output to file Message.out has been left untouched, however. Only new messages and messages from the FSE environment are sent to the logFile routines.

Num2String A module from the EcoCurves utility library with functions that write numbers to string. The source code of module Num2String is provided.

FSE4 Function UpdatePointers knows which shell commands are meant for a single model instance and which ones are meant for the entire model.

integModelOwn Communicates Times and flags with the integrator integModelOwn

SimulationUtlities allows th use of the Calendar utility of FSE to find out about the current year and daynumber.

ProgramInfo is used to pass model name, author and version of AnimoFSE4 to FSE.

AnimoParameters contains array sizes used in Animo that were previously contained in PARAM.INC

AnimoInstances contains all Animo variables declared as pointers *and* as elements of a user defined type SimulationData. An allocatable array Kolom of type SimulationData then contains a complete instance status in each of its array elements. By means of a call to SetModelInstance (a subroutine in module AnimoInstances) the pointers are associated with the elements of a complete instance status in Kolom(instance).

AnimoWater in SwapMode AnimoWater reads an unformetted input file with Swap oputput for every Animo instance. The information on this file is passed to Animo via calls to the Hydrology subroutine contained in module AnimoWater. In FussimMode, Hydrology returns information prepared by subroutine FromFussim in module FF. This information is updated by Fussim at every ShellTimeStep.

The model routine Animo

The actual FSE model routine Animo is little more than a large case construct. For case contains the execution of a specific command coming from the calling program (FSE core routine RunCoupledModels or the integrator integModelOwn). RunCoupledModels runs models by calling them with different commands in a meaningful order. Hence, the order of commands like is not contained in the models themselves. Animo has sections for executing the commands:

CreateInstances allocates the number of Animo instances prescribed in input file Control.dat,

DeleteAllInstances de-allocates the Animo instances at the end of a model run,

Initialise a model instance by calling subroutine Input1 with the name of the Dirigfile for that instance in pointer variable Dirigfile. The call to input1 has not been changed, and neither are the other calls in the initial section of Animo,

Prepare Just before update this call is done. Since Animo does not explicitly calculate rates of change this section is almost empty. It just contains a test for reaching a new calendar year. If so, a time event is requested which is initiated by FSE *before* the next update call.

Output , Output is taken care of in Animo by output routines called during the update. No output in this section.

Dynamic This is the update command requesting the model to proceed to the next shell time.

Terminal This terminates the model run.

InstanceEvent Signals the beginning of a new year. Code for ending the previous year and beginning the next year is executed here.

CheckModelEvent This model triggers a periodic model event for future output purposes.

ModelEvent . has no function for the simulation itself but may be used to implement model output (output for all instances). It has been used during development for calling graphical output with an interval of MovieTimeStep days. Although the movie generating routines have been removed from the program, the variable MovieTimeStep is still present on Control.dat and the ModelEvent section may be used to generate an overview of the combined Animo instances. Note that a ModelEvent is executed once for the entire model and not for each instance separately. A model variable XXX has to be referred as Kolom(inst)%XXX for instance number inst between 1 and NumberOfInstances and *not by pointer variable XXX* which points at just one of the values XXX.

GetProgramInfo copies information into character strings of module Program-Info. Used by FSE for logfile output.

There are a few more commands but the corresponding model sections do not contain anything else than a continue statement.

2.2.3 AnimoFussimControl

This is the controller called by FSE subroutine RunCoupledModels during setup, at each shell time step, and after reaching ShellFinishTime. The controller belongs

to the user part of FSE and has been added to the source file AnimoWater.f90. At setup AnimoFussimControl reads HydrologyMode from the input file Control.dat and calls module AnimoWater to actually set the mode to SwapMode or FussimMode.

AnimoFussimControl also reads the Animo input and collects the Start and Stop times (specified as Year numbers and a Day) for all Animo instances. Animo instances with a start time between ShellStart and ShellFinish are started at their own start time. In SWAP mode this results in a number of independent Animo runs, one per instance.

In FussimMode the controller starts Fussim right from the beginning, at ShellStartTime. It also verifies that all Start times of the Animo instance are the same and starts the whole thing as soon as that time is reached.

After reaching ShellFinishTime, the controller warns for incomplete Animo runs. Animo instances terminated before they reached their own Stop time may have produces incorrect output files, e.g. balances for years that have not been simulated.

The controller is an essential part of the coupling. It may be adapted for additional models like a crop model that starts each year at a certain date and stops at harvest time at its own request. Entire crop rotation sequences are easily implemented in the controller once the crop models are there and interact with the soil.

2.2.4 Module AnimoWater

AnimoWater contains the FSE 4 controller AnimoFussimControl, which organizes the start and stop of the coupled models Animo and Fussim. Besides this controller, AnimoWater is the source of all the hydrological information for Animo, originating either from Swap unformatted input files, or from Fussim. Hence, the Animo model itself "does not know" from where its water contents and fluxes come.

Some more details are provided in the following list with the contents of module AnimoWater

HydrologyMode Saved integer variable set by the controller at either SwapMode or FussimMode during setup. Adding more modes is possible,

AnimoStart, AnimoEnd Datastructure with start and stop times of the Animo instances read during setup from the Animo input files by the controller AnimoFussimControl,

SwapControl Datastructure with control information for reading Swap files in subroutine HydroInit (contained in AnimoWater),

Geometry Datastructure storing the Animo geometry. Values come from Swap files or from Fussim,

SwapColumn Datastructure for storing Swap time step info passed to Animo by Hydrology calls,

ConcData Datastructure used for reading LATBOU input files with lateral concentrations; in SwapMode such a file is read for each instance, in FussimMode concentrations are read from file for the leftmost and rightmost instance only[3],

[3] After completion of the model coupling by EcoCurves, the datastructures dealing with the domain geometry and the lateral boundaryconcentrations have been changed once more in order to deal with non-rectangular domains allowed by Fussim. This is documented partly in Chapter 2.5.

FromFussim Hydrological information from Fussim, converted into Animo units by routine FromFussim in module FF,

AnimoFussimControl The FSE controller (see Chapter 2.2.3),

AnimoHydrologyMode private subroutine for setting the hydrology mode,

HydroInit initializes Swap file reading in SwapMode, does nothing in FussimMode,

HydroNextStep reads the data for the next time step in SwapMode, does nothing in FussimMode,

LatCinit initializes LATBOU file reading, does nothing in FussimMode,

LatCNextStep reads the data for the next time step,

Hydrology communicates hydrology and concentration variables to calling Animo routines,

checks and error messages , private in module AnimoWater. These routines cannot be accessed from outside AnimoWater.

2.2.5 Module FF with subroutine FromFussim

Subroutine FromFussim is the only subroutine in FF which can be called from outside. It is called from Fussim as part of different command sections

Initial Translates the geometry of Fussim rows and layers into the geometry variables of Animo[4]. This requires input from the AnimoGeometry.dat. The name of this file is read from Control.dat (variable Fussim2AnimoTrans). Fussim rows are combined into Animo layers and Fussim columns are combined into Animo columns. Note that the number of horizons in Fussim must be identical with the number of horizons defined in the Animo input files.

Prepare This call takes place just before the updating the Fussim state over a time step. It stores ShellTimeStep begin values.

Dynamic This call takes place at the end of a (small) Fussim time step. Fluxes are integrated here in order to get a total flux at the end of the shell step.

Supply This call takes place from Fussim just before returning to FSE at the end of a shell time step. The results over the shell step just taken are used to update the hydrology values for Animo. Volume averages or boundary totals are calculated for the various variables and the results are stored in the instances of FussimColumn which is in AnimoWater. Also, using the total horizontal water flux between the various columns, the required update order for the Animo columns is set by means of a call to FSE subroutine SetInstanceOrder.

2.3 Source files of the coupled model

Animo source files in which have no changes or just very simple changes were made are in the **folder AnimoRoutines**. In many files, for instance, “include PARAM.INC” has been replaced by “USE AnimoParameters”, and routines that create output files the function AnimoOutXXXX precedes the filename in order

[4] in variable Geometry of module AnimoWater which is USEd in FF.

to put the file in an output folder. New files or files in which larger changes were made are:

main.f90 containing the "Pretty Standard Main" and the FSE select routines (see Chapter 2.1)

AnimoFSE4 containing module AnimoFSE4 (Chapter 2.2.2) and subroutine Stoponerror.

Fussim2.f90 contains the "top level" of the Fussim model. Most of the model is in the linked library F_6100.

AnimoParameters.f90, AnimoBalance.f90 contain array sizes and constants used throughout Animo

AnimoInstances.f90 declares instances of all Animo variables (see Chapter 2.2.2)

AnimoWater.f90 contains module AnimoWater (see Chapter 2.2.4) and the controller AnimoFussimControl.

Input1.f90 now receives the name of a dirigfile as an input argument and calls subroutine Hydrology of module AnimoWater for hydrological input and for lateral concentrations.

ff.f90 contains module ff used by Fussim to prepare the hydrological info for Animo.

ABtype, logFile, DTnowString, Num2String parts of an EcoCurves utility library that are used in the FSE version of Animo or by the linked FSE library.

NRtype Fortran-90 variable KIND convention from Press *et al.* (1992) which is used in new declarations.

2.4 Linked libraries

Linked libraries simplify maintenance and compilation time. The Animo-Fussim coupled model requires:

F_6100 contains most of the Fussim model. Only the top level routine called by FSE is not included in the library,

IMSL The International Mathematical and Statistical Library (Anonymous, 1998) from which matrix inversion routines are used by Fussim's solver,

FSE4 contains the Fortran Simulation Environment, maintained by EcoCurves,

COMMAN contains the Communication Manager that allows models the communicate using PUT en GET calls to and from a blackboard with "published" variables,

Defsys used by COMMAN,

Weather optionally called by FSE4 to access weather files,

TTUTIL utility library (van Kraalingen & Rappoldt, 2000) used for its input and file open routines. The coupled model requires version 4.15, which allows a larger unit number range (up to 999) than version 4.14.

2.5 Geometry mapping

It is obvious that proper computations require good correspondence between both models with respect to soil properties and to exchange of state and rate variables.

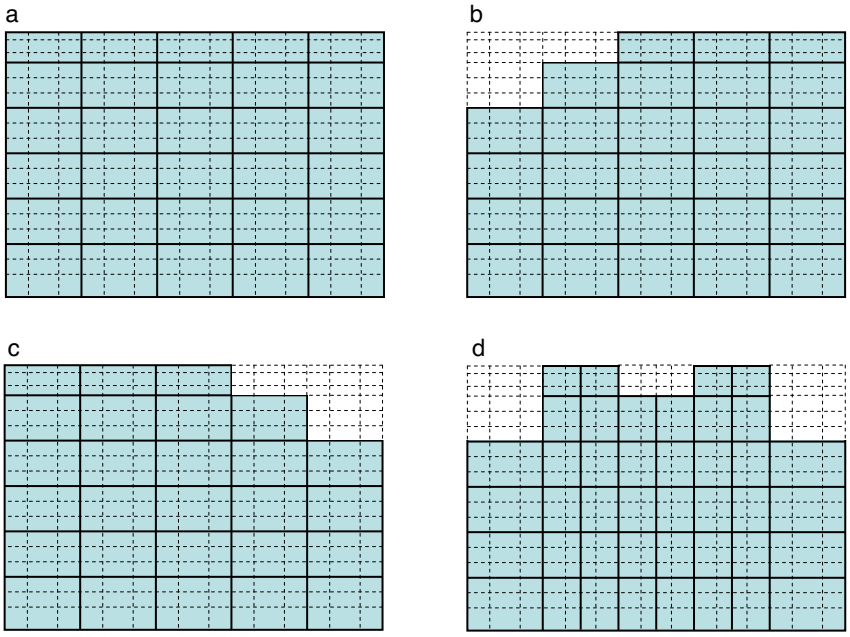

Figure 2.2. Some possible geometry mappings between Animo and Fussim2. The solid lines represent the boundaries of Animo, while the dashed lines represent the boundaries of the Fussim2 grid cells. The colored zones are soil, while the white zones are left out of computation. See text for further explanation.

It is most likely that the grid used by Fussim2 is much more dens than the number of layers and number of instances used by Animo. The most convenient geometry mapping requires that a single Animo layer consists of a whole number of Fussim2 grid cells. Furthermore, Animo uses soil horizons, each with typical soil properties, such as water content at saturation. These should correspond to soil properties used in Fussim2. Soil horizons must be bounded by a whole number of Animo layers.

Figure 2.2 gives some possible mappings between Fussim2 and Animo that are briefly described below.

Case a refers to a simple rectangular soil profile with 5 Animo instances each having 6 layers. Each Animo layer consists of 9 Fussim2 grid cells.

Case b is an example of a soil profile next to a ditch. The ditch wall is represented by a stepwise ditch bank. In total 5 Animo instances are used, The first Animo instance has only 4 layers, the second has 5 layers, and the remaining 3 Animo instances all have 6 layers. Each Animo layer consists of 9 Fussim2 grid cells. Since all Animo instances end at the same depth at the bottom, it can be simply deducted what the neighboring layer is. For example, the first Animo layer of the first instance corresponds to the second Animo layer of the second instance. The shift is simply determined by the difference in number of Animo layers between the neighboring instances.

Case c is a mirrored version of case b, indicating that irregular boundaries can either lie at the left or at the right.

Case d shows that also cases with two ditches here represented by vertical ditch banks can be considered as well as the presence of a shallow trench somewhere in the middle of the field. Note that other than cases a-c, the number of Fussim2 grid cells per Animo layer now is different; Fussim2 now uses 18

columns and 18 layers.

Besides the earlier mentioned correspondence between Animo boundaries and Fussim2 grid boundaries, there is another requirement that all Animo instances end at the same depth, and that all Animo layers coincide at the same depth.

As mentioned before, lateral boundary conditions are supplied by the user in file LatBouLeft.csv and LatBouRight.csv. The exchange code is adapted for irregular boundaries. For example, for case b (Figure 2.2) the left boundary concentration for the first layer is used only for Animo instance 3, and no longer for the first layer of the first Animo instance (which is locate at a greater depth).

For infiltrating ditch water through the bottom of a ditch, we needed to add a functionality to handle the concentration of this infiltrating ditch water. Similar to the supply of left and right boundary concentrations, this can be done through a file named TopBouCxxxx.csv, where xxxx refers to the corresponding Animo instance. Whether or not TopBouC needs to be used by the model is supplied by the use via a separate input variable. In the exchange towards Animo the infiltrating amount of water through the ditch bottom and the corresponding concentration are communicated through the variable FLG (the third drainage component in Animo).

The geometry mapping requires some, but little, input by the user in the data file AnimoGeometry.dat which is described in the following subsection.

2.5.1 The file AnimoGeometry.dat

The file AnimoGeometry.dat contains the following information. The example refers to the special case d of Figure 2.2.

NL The number of Animo layers per instance.

```
NL = 4, 6, 6, 5, 5, 6, 6, 4
```

FRB Supply the bottommost Fussim2 layer (Fussim2 Row Bottom) for each Animo layer for each Animo instance.

```
FRB =
          9 12 15 18
      3 6 9 12 15 18
      3 6 9 12 15 18
        6 9 12 15 18
      3 6 9 12 15 18
      3 6 9 12 15 18
          9 12 15 18
```

FCR Supply the leftmost Fussim2 column number for each Animo instance.

```
FCR = 1, 4, 6, 8, 10, 12, 14, 16
```

TopLayerInDitch Per Animo instance indicate whether (1) or not (0) to use TopBouC.

```
TopLayerInDitch = 1, 6*0, 1
```

TopCfile Names of files per Animo instance containing the top boundary concentration values.

```
TopCfile =
'TopbouC0001.csv', 'TopbouC0002.csv', 'TopbouC0003.csv', 'TopbouC0004.csv'
'TopbouC0005.csv', 'TopbouC0006.csv', 'TopbouC0007.csv', 'TopbouC0008.csv'
```

Chapter 3

Examples

After starting the model it asks for the loglevel. A larger loglevel leads to more messages, coming from lower level subroutines and functions. A large value makes sense only if one wants to inspect the logging of FSE4. A negative value holds the program and asks for a <Return> at every output line. A second question refers to screen output.

The control file Control.dat is looked for at the default directory of the started program. For the output produced by Animo instances folders AnimoOutput0001, AnimoOutput0002, ... should be there.

During the development of the coupled FSE version of Animo the possibility to read the hydrology from SWAP was maintained. At different stages this functionality was tested successfully and the first example below is meant to demonstrate that *nothing* has changed in the output of a Swap-Animo run.

3.1 Testing the Swap mode

Two instances of Animo each run an Animo example provided by Alterra. The output can be compared with the output of the original Animo version and besides the runtime clock messages there are no differences in output.

3.1.1 Control.dat

The control file tells FSE that the Calendar utility is being used by means of

```
SimulationUtilities = 'Calendar'
SimUtilityDataFile  = 'CONTROL.DAT'
```

and that the Calendar input data should be read from the control file, Control.dat. The actual input data connect ShellStartTime with day 350.0 of 1973 and sets the unit of time to a day:

```
! attaching a calendar time to the Shell start time:
StartYear =  1973     ! the start year of the simulation
StartDOY  =   350.0   ! Day Of Year beginning from 1.
OneDay    =     1.0   ! Defines a day as unit of time
```

The settings for the simulation time are

```
ShellStartTime   = 0.0
ShellFinishTime  = 10000.0  ! this spans both Animo runs
! output frequency (d)
ShellOutputInterval = 1.0
ShellSyncInterval   = 0.0   ! no sync time interval used
! coupling frequency (d)
ShellTimeStep       = 1.0
```

and an additional variable read by controller AnimoFussimControl controls the hydrology mode:

```
! Animo hydrology
HydrologyFrom = 'swap'    ! Fussim will NOT be started
```

Then we have the actual model input for Animo:

```
! Animo Dirigfiles for the two instances
Dirigfile = 'CranGrass.ini', 'CranMais.ini'
MovieTimeStep = 10.0  ! time between model events
```

The next variables in Control.dat are information that FSE requires to run the coupled models. For each of the models a name is provided, an integrator, the name of a datafile (possibly the primary datafile on which more files are specified) and the number of instances of each model. This example requires two Animo instances and the data for Animo are on Control.dat as well:

```
ModelName        =   1*'FUSSIMFSE4',         'Animo'
Integrator       =   2*'ModelOwn'
ModelDataFile    =   1*'FUSSIM.DAT',         'Control.dat'
IntegSettFile    =   'Control.dat',          'Control.dat'
NrOfInstances    =   1*1,                    2
```

Note that Fussim is mentioned here as well. It will not be started from the controller, howewer, since the hydrology mode was set to 'Swap'. The remaining veriables on Control.dat specify options for model communication and options for output. A new model option

```
UseInstanceOrder =   1*.false., .true.
```

is set .false. for Fussim and .true. for Animo tells FSE that the Animo instances should be updated in a user-defined order. By default the order is $1, 2, 3, \ldots$, but this may be changed with a call to FSE 4 subroutine SetInstanceOrder this may be changed. Module FF makes this call and sets a downstream update order of the Animo columns.

3.1.2 What happens?

In the control file two "Dirigfiles" are mentioned, CranGrass and CranMais, one for each instance. File CranGrass

```
Animo40
GEN="CranGrass\general.inp"
MAT="CranGrass\material.inp"
PLA="CranGrass\plant.inp"
SOI="CranGrass\soil.inp"
```

```
BOU="CranGrass\boundary.inp"
INI="CranGrass\initial.inp"
MAN="CranGrass\management.inp"
SWU="CranGrass\swatre.unf"
WAI="CranGrass\watbal.inp"
WAU="CranGrass\watbal.unf"
CHE="CranGrass\chempar.inp"
INO="initial.out"
CRU="CranGrass\crop_ext.inp"
LAT="CranGrass\latbou.csv"
MES="message.out"
END
```

contains a standard set of filenames which is read by input routine Input1.f90. The name of the SWAP input file "`CranGrass\swatre.unf`" is passed by Input1.dat to subroutine HydroInit of module AnimoWater. This initializes the reading of this file and the transfer of hydrological data to Animo. In the same way also the file with Lateral concentrations LATBOU.CSV is dealt with.

At first, however, the controller AnimoFussimControl reads the start and stop times from the "general input" files mentioned in the two Dirigfiles. The controller writes to the logfile:

```
AnimoFussimControl: Instance      StartTime          FinishTime
AnimoFussimControl: ------------------------------------------------
AnimoFussimControl:     1      01-Jan-1992 00:00   01-Jan-2000 00:00
AnimoFussimControl:     2      01-Jan-1974 00:00   01-Jan-1983 00:00
```

This information is used by the controller to start the instances at their respective start times and to verify at the end of the coupled model run that the Animo finish times have indeed been reached.

3.2 Fussim with a single Animo instance

This example makes use of the Animo input in the CranGrass example. Now the hydrology now comes from Fussim, however. Marius Heinen prepared Fussim input data with 5 soil horizons (the number of horizons in the CranGrass example, verified by Animo) and 44 Fussim layers. The Fussim layers are grouped into 22 Animo layers according to the contents of

```
Fussim2AnimoTrans = 'AnimoGeometry.dat'
```

on the control file. AnimoGeometry.dat contains the grouping of Fussim layers and Fussim columns. In this example

```
NL     = 22
FRB = 2 4 6 8 10 12 14 16 18 20 22 24 26 28 30 32 34 36 38 40 42 44
! for each instance of ANIMO
FCR = 1*1              ! single instance
```

there is a single column, ending at Fussim column FCR ("Fussim Column Right"), and there are 22 Animo layers, ending at the Fussim layers given in variable FRB ("Fussim Row Bottom").

The control input file sets the hydrology mode to "Fussim" and an appropriate range of the simulated time:

```
! attaching a calendar time to the Shell start time:
StartYear =  1991     ! the start year of the simulation
StartDOY  =   350.0   ! Day Of Year beginning from 1.
OneDay    =     1.0   ! Defines a day as unit of time
ShellStartTime   = 0.0
ShellFinishTime  = 3200.0
! output frequency (d)
ShellOutputInterval = 1.0
ShellSyncInterval   = 0.0   ! no sync time interval used
! coupling frequency (d)
ShellTimeStep       = 1.0
! Animo hydrology
HydrologyFrom = 'fussim'   ! Fussim will start at ShellStartTime
```

3.3 Fussim with 5 Animo instances

This example is very similar to the example in Chapter 3.2. The Fussim example sets up drainage at the lower left side of a domain. There is daily rainfall from above, the columns contain different water contents and there is a flux to the lower left corner.

3.3.1 Control and Input

The input of Animo has been organized in a slightly different way in this example. Dirigfiles are given in the control file, like in the previous example. Now this looks like

```
! Animo hydrology
HydrologyFrom = 'fussim'   ! Fussim will start at ShellStartTime
                           ! Animo Instances at their own start/stop times

! Animo Dirigfiles for a single instance
! (the content of the ModelDataFile for the Animo model)
Dirigfile = 'AnimoInput\CranGrass0001.ini', 'AnimoInput\CranGrass0002.ini',
            'AnimoInput\CranGrass0003.ini', 'AnimoInput\CranGrass0004.ini',
            'AnimoInput\CranGrass0005.ini'

! Animo Geometry is derived from the Fussim geometry as described in
Fussim2AnimoTrans = 'AnimoGeometry.dat'
```

The 5 dirigfiles have been created as (edited) copies from the same file and are stored together in folder AnimoInput. File CranGrass0003, for example contains:

```
Animo40
GEN="AnimoInput\CranGrass\general.inp"
MAT="AnimoInput\CranGrass\material.inp"
PLA="AnimoInput\CranGrass\plant.inp"
SOI="AnimoInput\CranGrass\soil.inp"
BOU="AnimoInput\CranGrass\boundary.inp"
INI="AnimoInput\CranGrass\initial.inp"
MAN="AnimoInput\CranGrass\management0003.inp"
SWU="AnimoInput\CranGrass\swatre1.unf"
```

```
WAI="AnimoInput\CranGrass\watbal.inp"
WAU="AnimoInput\CranGrass\watbal.unf"
CHE="AnimoInput\CranGrass\chempar.inp"
INO="initial.out"
CRU="AnimoInput\CranGrass\crop_ext.inp"
LAT="AnimoInput\CranGrass\xxxx.csv"
MES="message.out"
END
```

The Dirigfiles, however, cannot be all identical, since the "MAN file" (management.inp) remains open during the model run. Therefore copies of this file were made. This also creates the possibility to create differences in the nutrrient management among the Animo instances. In this example, the nutrient additions were completely removed from the management file for the instances 1, 2 and 3.

Also the "LAT file" remains open during the model run. In FussimMode, however, only 2 such files are used, the ones for the leftmost and rightmost Animo instance. The above example file for instance 3 contains a dummy name, a non-existing file that would lead to a error message when the program would attempt to read it.

Note that each Dirigfile also specifies the names of two output files. These names are the same for all instances. Nevertheless each instance creates its own INO and MES file, since each instance writes to its own output folder.

3.3.2 Result

Figure 3.1 shows some results of the simulation with 5 Animo instances. Management additions are made only on the two rightmost columns which is clearly visible in the plots. A large part of the Nitrate disappears from the profile during the simulation.

The plots have been constructed by output calls to a graphical library temporarily inserted in the ModelEvent section.

3.3.3 Update order

The Animo instances are placed in positive direction, so the rightmost instance has number 5. Since water flow is to the left a downstream update order must be 5,4,3,2,1 as soon as the horizontal flux emerges. In the logfile we indeed see the following update commands

```
RunCoupledModels: SimulateInstances
   ModelOwn: [19.000000] Simulate (ModelOwn,1) <=> (FUSSIMFSE4,1)
      FUSSIMfse4: Dynamic
      FromFUSSIM: Supply
   ModelOwn: [19.000000] Simulate (ModelOwn,6) <=> (ANIMO,5)
      Animo: Dynamic for Instance 5
   ModelOwn: [19.000000] Simulate (ModelOwn,5) <=> (ANIMO,4)
      Animo: Dynamic for Instance 4
   ModelOwn: [19.000000] Simulate (ModelOwn,4) <=> (ANIMO,3)
      Animo: Dynamic for Instance 3
   ModelOwn: [19.000000] Simulate (ModelOwn,3) <=> (ANIMO,2)
      Animo: Dynamic for Instance 2
   ModelOwn: [19.000000] Simulate (ModelOwn,2) <=> (ANIMO,1)
```

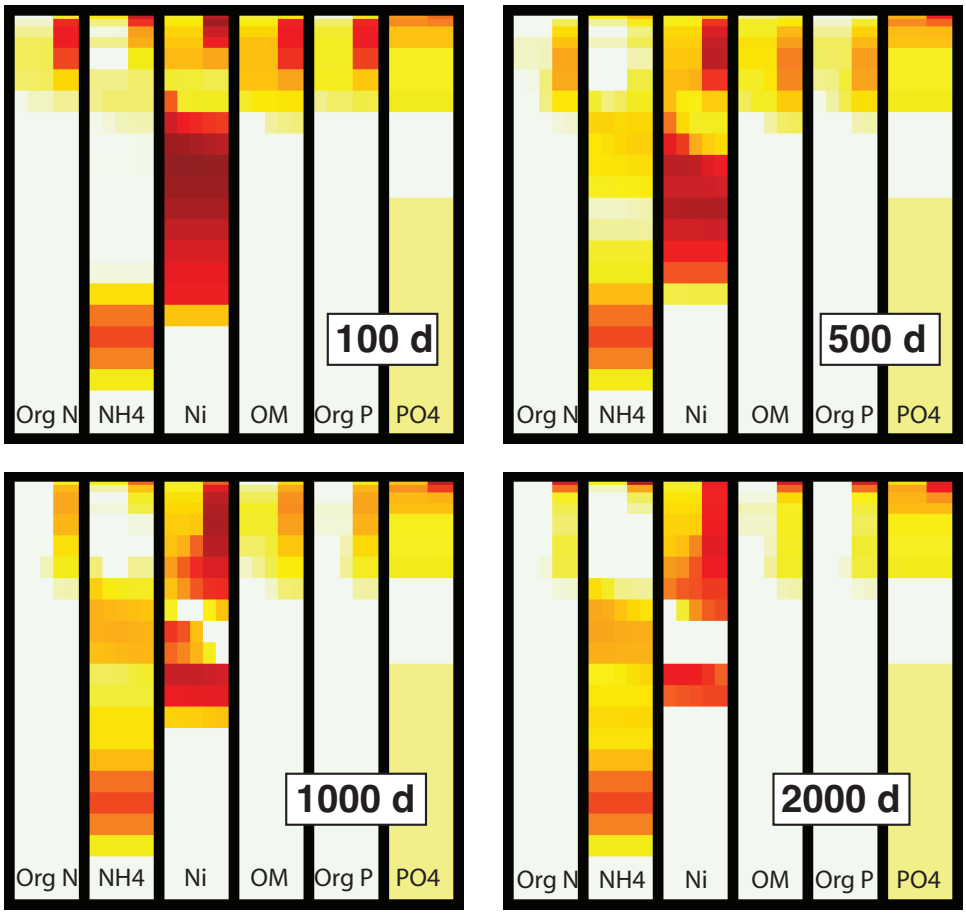

Figure 3.1. Example calculations with Fussim and 5 instances Animo. The four figures show results for day 100, 500, 1000 and 2000, respectively. Each figure consists of 6 bars that show the soil concentrations in 5 columns and 22 rows for Organic N, ammonium N, Nitrate N, organic matter, organic P and phosphate P, respectively. The color scales are logarithmic with 2 decades (a factor 100) between white (lowest concentration during the simulation) and brown (highest concentration).

```
      Animo: Dynamic for Instance 1
RunCoupledModels: === 05-Jan-1992 00:00 ==== ShellTime = 20.000000
```

showing that Fussim is updated first (it comes first in Control.dat), followed by the 5 instances of Animo in reverse order. Note that the string `(ModelOwn,6) <=> (ANIMO,5)` means that instance 5 of Animo was connected with status block 6 of integrator ModelOwn[1].

[1] In a status block, the integrator stores control information about a model instance. From the above logfile output it may be inferred that status block 1 was connected to Fussim and that blocks 2 to 6 deal with the five Animo instances.

References

Anonymous, 1998. IMSL, Fortran 90 Subroutines and Functions. Technical report, Visual Numerics, www.vni.com.

de Willigen, M. H. A. P., 1998. A two-dimensional simulation model for water flow, solute transport, and root uptake of water and nutrients in partly unsaturated porous media. Technical report, DLO Research Institute for Agrobiology and Soil Fertility & C.T. de Wit Graduate School for Production Ecology, Wageningen, the Netherlands. QASA 20.

de Willigen, M. H. A. P., 2001. Fussim2 version 5; New features and updated user's guide. Technical report, Alterra, Wageningen, the Netherlands. Alterra–rapport 363.

Groenendijk, P., Renaud, L. V., Roelsma, J., 2005. Prediction of Nitrogen and Phosphorus leaching to groundwater and surface waters. Technical report, Alterra, Wageningen, the Netherlands. Alterra–rapport 983.

Heinen, M., 2001. FUSSIM2: brief description of the simulation model and application to fertigation scenarios. Agronomie 21, 285–296.

Press, W. H., Flannery, B. P., Teukolsky, S. A., Vetterling, W. T., 1992. Numerical Recipes, the art of scientific computing, second edition. Cambridge University Press, New York.

Rappoldt, C., van Kraalingen, D. W. G., 2007. FSE 4.1, user guide and technical documentation. EcoCurves rapport in prep., available from kees.rappoldt@ecocurves.nl.

van Kraalingen, D. W. G., 1995. The FSE system for crop simulation, version 2.1. Technical report, DLO Research Inistitute for Agrobiology and Soil fertility; The C.T.de Wit graduate school for Production Ecology, Wageningen, the Netherlands.

van Kraalingen, D. W. G., Rappoldt, C., 2000. Reference manual of the fortran utility library ttutil v. 4. Technical report, Plant Research International (Report 5), Wageningen, the Netherlands. Updated PDF file available from kees.rappoldt@wur.nl.

Appendices

Appendix A

Animo source files

The files most affected by the changes are the following:

Param.inc replaced by a Fortran-90 module in AnimoParameters.f90.

animo.inc replaced by the Animo instance variables declared in module AnimoInstances.f90. The Fortran-90 module also contains a long list of pointers which is used in the Animo model in order to keep as much of the original code as possible.

ANIMO.for replaced by the FSE 4 simulation model ANIMOfse4. Calls to lower level "process subroutines" have been left unchanged. The integration loop of the interation has been adapted to the FSE4 requirements.

Outbal.inc replaced by declarations in Fortran-90 module AnimoBalance.

input1.for geometry and water related input has been adapted. Call to subroutines in the new ANIMOWater.f90 replace READ statements for "water input" in the original model.

Outsel.for adapted to the use of Animo instances.

A full Animo source file comparison has been made using catenated source files. For the original Animo 4.25 the following list has been used.

```
   43 lines from Param.inc
  704 lines from animo.inc
 1091 lines from ANIMO.for
   24 lines from Outbal.inc
 2633 lines from input1.for
  958 lines from Outsel.for
  648 lines from Addit.for
  285 lines from Aeration_original.for
  123 lines from AERATION_SONICG.FOR
   49 lines from CORRECTION.FOR
   63 lines from Denitr.for
   48 lines from FRACPNT.FOR
  547 lines from Function.for
  524 lines from Grassprd.for
   61 lines from Grassup3.for
  213 lines from grass_init.for
  984 lines from Hydro_aggregated.for
  218 lines from Hydro_detailed.for
```

```
  974 lines from Inicalc.for
  283 lines from Init.for
   90 lines from init_yr.for
  158 lines from Input_addit.for
  425 lines from Input_cropext.for
  599 lines from Input_Echo.for
  173 lines from Input_hydro.for
  358 lines from MAPOHYDRO.FOR
   61 lines from MAPOINICALC.FOR
   64 lines from MAPOINIT.FOR
  202 lines from MAPOINPU.FOR
 3661 lines from MAPOOUTBAL.FOR
 1289 lines from MAPOTRANSPORT.FOR
  655 lines from Miner.for
   81 lines from MODFLUX.FOR
 1352 lines from Outbal_calc.for
  164 lines from Outbal_Init.for
 2001 lines from Outbal_write.for
  789 lines from Output2.for
  150 lines from Output_Init.for
  373 lines from OXYDEM.FOR
  221 lines from Rates.for
  197 lines from RESPIRATE.FOR
  117 lines from root_extern.for
  110 lines from root_grass.for
  154 lines from root_plant.for
   46 lines from Temper.for
  189 lines from Transca.for
  422 lines from Transgen.for
 2412 lines from Transorp.for
  242 lines from TRANSPORT.FOR
  342 lines from Transsub.for
  167 lines from UBoundconc.for
   66 lines from Upintg_Extern.for
   73 lines from Upintg_Grass.for
   96 lines from Upintg_Plant.for
   80 lines from Uptpar_Extern.for
  242 lines from Uptpar_Grass.for
  321 lines from Uptpar_Plant.for

Total length:28615 lines
```

For the coupled Animo the first few names are different. The rest of the list below looks pretty much the same.

```
   29 lines from AnimoParameters.f90
 2161 lines from AnimoInstances.f90
 1371 lines from ANIMOfse4.f90
   67 lines from AnimoBalance.f90
 2793 lines from input1.f90
 1187 lines from ANIMOWater.f90
  964 lines from AnimoRoutines/Outsel.for
  652 lines from AnimoRoutines/Addit.for
  291 lines from AnimoRoutines/Aeration_original.for
```

```
  127 lines from AnimoRoutines/AERATION_SONICG.FOR
   53 lines from AnimoRoutines/CORRECTION.FOR
   67 lines from AnimoRoutines/Denitr.for
   52 lines from AnimoRoutines/FRACPNT.FOR
  547 lines from AnimoRoutines/Function.for
  528 lines from AnimoRoutines/Grassprd.for
   65 lines from AnimoRoutines/Grassup3.for
  218 lines from AnimoRoutines/grass_init.for
  996 lines from AnimoRoutines/Hydro_aggregated.for
  222 lines from AnimoRoutines/Hydro_detailed.for
  990 lines from AnimoRoutines/Inicalc.for
  287 lines from AnimoRoutines/Init.for
   95 lines from AnimoRoutines/init_yr.for
  162 lines from AnimoRoutines/Input_addit.for
  429 lines from AnimoRoutines/Input_cropext.for
  606 lines from AnimoRoutines/Input_Echo.for
  209 lines from AnimoRoutines/Input_hydro.for
  366 lines from AnimoRoutines/MAPOHYDRO.FOR
   65 lines from AnimoRoutines/MAPOINICALC.FOR
   68 lines from AnimoRoutines/MAPOINIT.FOR
  206 lines from AnimoRoutines/MAPOINPU.FOR
 3666 lines from AnimoRoutines/MAPOOUTBAL.FOR
 1304 lines from AnimoRoutines/MAPOTRANSPORT.FOR
  661 lines from AnimoRoutines/Miner.for
   85 lines from AnimoRoutines/MODFLUX.FOR
 1363 lines from AnimoRoutines/Outbal_calc.for
  173 lines from AnimoRoutines/Outbal_Init.for
 2012 lines from AnimoRoutines/Outbal_write.for
  837 lines from AnimoRoutines/Output2.for
  156 lines from AnimoRoutines/Output_Init.for
  377 lines from AnimoRoutines/OXYDEM.FOR
  229 lines from AnimoRoutines/Rates.for
  201 lines from AnimoRoutines/RESPIRATE.FOR
  121 lines from AnimoRoutines/root_extern.for
  114 lines from AnimoRoutines/root_grass.for
  157 lines from AnimoRoutines/root_plant.for
   50 lines from AnimoRoutines/Temper.for
  198 lines from AnimoRoutines/Transca.for
  426 lines from AnimoRoutines/Transgen.for
 2433 lines from AnimoRoutines/Transorp.for
  247 lines from AnimoRoutines/TRANSPORT.FOR
  342 lines from AnimoRoutines/Transsub.for
  171 lines from AnimoRoutines/UBoundconc.for
   70 lines from AnimoRoutines/Upintg_Extern.for
   77 lines from AnimoRoutines/Upintg_Grass.for
  100 lines from AnimoRoutines/Upintg_Plant.for
   84 lines from AnimoRoutines/Uptpar_Extern.for
  246 lines from AnimoRoutines/Uptpar_Grass.for
  329 lines from AnimoRoutines/Uptpar_Plant.for

 Total length:32102 lines
```

The result of the file comparison is available as a separate document FileDiffs-Animo425FSE.pdf.