THE DEVELOPMENT OF A COMPUTERIZED MANAGEMENT SYSTEM FOR IRRIGATION SCHEMES

Report to the Water Research Commission

by

Dr N Benadé[1]
Dr J Annandale[2]
Mrs H van Zijl[3]

[1]Previously Energy Laboratory,
Rand Afrikaans University,
currently N Benadé t/a NB Systems

[2]Department of Plant Production and Soil Science
University of Pretoria

[3]Energy Laboratory
Rand Afrikaans University

WRC Report: No: 513/1/97
ISBN No: 1 86845 336 7

Acknowledgements

The authors thank the Water Research Commission for financing this research which made the development of SWB possible.

We would also like to acknowledge the large contribution made by Prof G.S. Campbell from Washington State University for allowing us to use his NEWSWB model as a starting point for this project.

TABLE OF CONTENTS

LIST OF FIGURES

1 OPTIMISATION SYSTEM

1.1 BACKGROUND

Two separate projects, supported by the Water Research Commission were completed at the end of 1992; one at the Rand Afrikaans University on an optimisation system for minimizing distribution losses on irrigation schemes (Benadé, Engelbrecht and Annandale, 1990) and the other at the University of the Orange Free State aimed at maximizing irrigation efficiency on irrigation schemes (Mottram and de Jager, 1994).

The research at the Rand Afrikaans University focused on water delivery and the main tools developed were:

- A computer program, PROCAN, that simulates unsteady flow of water in any canal or river and
- a water administration system, RAUDB/WAS, that handles all water requests by farmers and is currently running on all major irrigation schemes throughout South Africa.

The water administration system never queries the amount of water requested by a farmer and the only limitation on the water delivered is the capacity of the canal system. There is only so much that can be done to minimise water losses in the canal system and discussions with water administrators on the Loskop irrigation scheme identified the need to query the amount of water that is requested by farmers to further optimise the water usage on the irrigation scheme.

We knew that a lot of research had already been done on irrigation scheduling and the main aim of this research project was to identify a suitable scheduling model and integrate it into the water administration system.

Once an irrigation scheduling program is integrated into the optimisation system it would be possible to compare the volume of water requested by the farmers with the actual water needed for all the crops on an irrigation scheme. This will give a complete picture of water utilization on an irrigation scheme and the integrated system will be a useful tool in the hands of water administrators who may decide to initiate scheduling services should it become apparent that water is not managed efficiently.

Irrigation farmers would also benefit from a management tool which integrates a mechanistic scheduling model and a water administration system. Not only will crop water requirements be accurately determined, but water orders (discharge rate and duration) can automatically be calculated and printed.

1.2 AIMS

The aims of this research project were

a) To integrate a mechanistic crop irrigation scheduling program such as that developed by the Department of Agrometeorology at the University of the Orange Free State into a water administration system developed at the Rand Afrikaans University;

b) To collect climate, soil and crop data at Loskop irrigation scheme for testing the irrigation scheduling program;

c) To implement the irrigation scheduling program in the Loskop irrigation scheme as part of an optimisation system to minimize water losses;

d) To generalise the results to an optimisation system that could be used on all major irrigation schemes in South Africa.

1.3 OPTIMISATION SYSTEM COMPONENTS

The optimisation system consists of an unsteady canal flow simulation model PROCAN, (Benadé *et al*, 1990) a water administration system, WAS, and a soil water balance model, SWB.

SWB, the new irrigation scheduling program developed during this research project and described in more detail in Chapter 2, is an upgraded version of NewSWB that was written in Turbo Pascal by Prof. Gaylon Campbell of Washington State University, Pullman, Washington USA. NewSWB is based on the simple Soil water balance model of Campbell and Diaz (1988). The crop growth component of NewSWB is more mechanistic than that found in Campbell and Diaz (1988).

PROCAN and WAS have been implemented (partially or fully) on various irrigation schemes throughout South Africa. The water administration system is used at the Loskop, Sandvet, Hartbeespoort, Pongola, Riet river, Vaalharts and Mooi river irrigation schemes. PROCAN has been successfully calibrated on the first 45 km of the left bank canal of the Loskop irrigation scheme.

A short description of each component of the optimisation system follows.

1.3.1 Unsteady canal flow model (PROCAN)

The aim of the unsteady canal flow model is to simulate unsteady flow in canals and rivers. The original development of PROCAN (then called RAUCAN) was funded by the Water Research Commission (Benadé *et al*, 1990).

PROCAN can be used for:

- calculating lag times;
- calculating evaporation and seepage losses;
- calculating hydrographs;
- evaluating different water distribution scenarios;
- training canal operators;
- designing new canals;
- analyzing existing canals.

Detailed information on PROCAN is provided in thesis written by Dr. Benadé (Benadé, 1994).

1.3.2 Water administration system (WAS or RAUDB)

WAS/RAUDB was originally developed as a data input utility for PROCAN, but it later developed into a fully computerised water administration system that can be used for:

- calculating water balance statements;
- calculating and printing water reports;
- calculating water releases into all canals, taking lag times and water losses into account;
- printing distribution sheets ("leikaarte");
- printing water bailiff instruction sheets ("kanaalwagopdragte");
- printing water releases.

Detailed information on WAS/RAUDB can be obtained from the irrigation schemes that is currently using it, or in Benadé (1994). WAS is implemented at the Loskop irrigation scheme and RAUDB at Hartbeespoort, Vaalharts, Sandvet, Riet river, Mooi river, Groot Marico, Tzaneen and Pongola.

The main aim of this research project was to integrate an irrigation scheduling model into the WAS. The following chapter will describe this in more detail.

1.4 Hardware requirements

SWB requires an IBM compatible computer running PC-DOS or MS-DOS version 3.00 or later. The program requires a hard disk to make full use of its capabilities. A numeric coprocessor is not required but if one is available, it will greatly improve calculation speed. If a graphics adapter is available, the program will use it to display graphics information. The CGA, MCGA, EGA, VGA, Hercules and 3270 PC graphic adapters are supported, but colour will only be used on EGA and VGA adapters. If EMS memory is available, it will be used to load overlay files, resulting in faster overlay swapping. The EMS driver must conform to the Lotus/Intel/Microsoft Extended Memory Specification.

2 IRRIGATION SCHEDULING MODEL

2.1 INTRODUCTION

Acting on advice of the WRC it was originally envisaged that the irrigation scheduling program from the PUTU family of models would be integrated into WAS. PUTU was developed by the Department of Agrometeorology at the University of the Orange Free State. After PUTU was purchased and evaluated the following conclusions were arrived at:

- The different modules of PUTU are written in Basic with a shell in Turbo Pascal (using Turbo Vision) from where the modules are executed. This made PUTU very difficult to integrate into WAS. The irrigation scheduling procedures would have to be rewritten in Turbo Pascal to ensure an elegant and effective integration.

- PUTU input data (climate, soil and crop data) is stored in ASCII files. This would have to be improved by making use of a data base like the Paradox engine that is currently being used in WAS.

After the assessment that PUTU could not be used in its current form, a decision was made to use NewSWB which is written in Turbo Pascal and therefore much easier to integrate into WAS. NewSWB was presented in detail by Prof Campbell (Washington State University) at a WRC supported workshop "Modelling the field water balance" held at Pretoria University in July 1993. NewSWB is similar to PUT-ANYCROP in that they are both mechanistic, generic crop growth models that simulate a supply and demand limited water balance for several soil layers. Their water balances can be driven by Penman-Monteith reference crop evaporation, making them ideal for real-time scheduling using automatic weather station data.

2.2 SOIL WATER BALANCE MODEL (SWB)

The program is called SWB, which is an abbreviation for Soil Water Balance. The program is written in Borland Pascal 7.0 and the Paradox Engine 3.1 is used for the underlying data base. The program uses the Penman-Monteith or a modified Priestley Taylor equation for its Potential Evapotranspiration (PET) estimation. It grows a generic canopy and root system, based on dry matter production (related to transpiration corrected for VPD) and the distribution of assimilates (Annandale, van der Westhuizen & Olivier, 1995). Soil water use is calculated with a dimentionless water uptake solution which allows either supply or demand to limit the process. The model is described in more detail in (Annandale *et al*, 1995).

What makes SWB unique is that it is based on Paradox a fully relational data base. This makes it possible to simulate the soil water balance on any number of fields in one run. All necessary data are saved into a few files that simplifies the management of large amounts of data. The data base Paradox can also be used to manipulate data files and even draw graphs of data and results. A utility program TUTILITY.EXE, supplied by Borland, is available to correct corrupted data files if necessary.

The irrigation scheduling model, SWB, can be used on a single farm, a group of farms or all farms on an irrigation scheme. The advantage of calculating the water required by all the crops on an irrigation scheme is that the water requested by the farmers can be compared to the actual crop water requirements. If there is a large difference between the water required and the water requested, steps can be taken to improve irrigation management on a specific irrigation scheme.

SWB is versatile for the following reasons:

- Simulated soil water content can be brought in line with measured water content during the growing season;
- The soil water balance can be calculated for any number of fields sequentially and displayed graphically;
- The program is user friendly and therefore easy to use;
- Maximum and minimum temperatures can be used to estimate PET if Penman-Monteith reference ET is unavailable;

- Password protection, error checking and on line help is provided;
- Water order forms are automatically completed;
- Measured data can easily be compared with simulated values;
- Extensive use is made of graphical output;
- The data files are compatible with the Paradox data base and therefore Paradox can be used to manipulate the data;
- Records of irrigation management are stored;
- Next irrigation date is forecast.

The mechanistic approach is more universally valid than empirical methods:

- Thermal time is used to predict crop development;
- Water use and evaporation are supply and demand limited;
- Soil water conditions influence canopy growth;

The source code of SWB is listed in Appendix C.

2.3 INPUT DATA REQUIREMENTS

2.3.1 Field data

Every field to be scheduled must be entered in the field table. The program provides a link to the type of crop planted in each field as well as a link to the weather station data base. It is possible to use only one crop and one weather station data base for all fields on an irrigation scheme that need to be simulated. A single field, or group of fields, can be simulated in one run. Fields can also be disabled if scheduling is not required.

The following field data is required:

- Field identification;
- Crop identification number;

- Planting date;
- Simulation starting date;
- Field size in ha (used for calculating required water volumes);
- Weather station identification number;
- Permanent wilting point (kPa);
- Field capacity (kPa).

2.3.2 Soil data

Initial data for 11 soil layers are needed. Even if the soil type is identical throughout the soil profile, it must be divided into 11 different layers and entered into the soil data table. This gives a much more realistic simulation of the infiltration process and crop water uptake than do the traditional single layer irrigation scheduling models.

The following information is needed for every field:

- Runoff curve number;
- Soil layer depth to the end of the layer (m);
- Soil layer initial water content at the start of the simulation (mm^{-1});
- Soil layer field capacity (mm^{-1});
- Soil layer permanent wilting point (mm^{-1}).

2.3.3 Crop data

All the different crop types with their corresponding model parameters are listed in the crop type table and can be accessed by the different fields. The parameters in the crop type table are only accessible by means of a password. This is to prevent the user from accidently changing or deleting a crop record. These parameters should only be adjusted by crop scientists that have experience in crop modelling.

The following data is required for each crop:

- Crop identification reference number;
- Crop type description in words;
- Average daily radiation transmission coefficient;
- Dry matter water ratio (corrected for VPD) (Pa);
- Radiation conversion efficiency, (kg MJ^{-1});
- Base temperature (°C) for thermal time calculations;
- Temperature for optimum light-limited growth (°C);
- Cutoff temperature for growing degree days in (°C);
- Emergence growing degree days;
- Growing degree days at the end of vegetative growth;
- Maturity growing degree days;
- Transition period (degree days) from vegetative to reproductive growth;
- Leaf senescence (degree days);
- Maximum rooting depth, (m);
- Fraction of shoot dry matter translocated to heads during grain fill;
- Leaf water potential at maximum transpiration, (J kg^{-1});
- Maximum possible transpiration rate, (mm/day);
- Specific leaf area (m^{-2} kg^{-1});
- Leaf-stem partition parameter;
- Canopy dry matter at emergence (kg m^{-2});
- Fraction of dry matter partitioned to roots;
- Potential grain production as fraction of shoot dry matter at anthesis;
- Root growth rate (m root [$\sqrt{kg\ m^{-2}}$ $]^{-1}$);
- Stress index;
- Percentage allowable depletion at emergence;
- Percentage allowable depletion during the vegetative stage;
- Percentage allowable depletion during the reproductive stage;
- Canopy interception (mm).

2.3.4 Weather data

The following daily weather data are needed for SWB to run successfully:

For each field:

- Irrigation (mm);
- Precipitation (mm).

For each weather station:

- Maximum temperature (°C);
- Minimum temperature (°C);
- Solar radiation ($MJ.m^{-2}\ day^{-1}$);
- Potential evapotranspiration (PET) ($mm\ day^{-1}$);
- Vapour pressure deficit (VPD) (kPa).

It is preferable to enter the measured values for these weather parameters if they are available. Maximum and minimum temperatures must be entered but the other three parameters will be estimated if measured data is unavailable.

The user can enter the data before starting a simulation or start the simulation and let the program prompt the user for the required data.

2.4 MODEL OUTPUT

2.4.1 Text output

Figure 2.1 shows sample text output for a simulation of field 1 on farm A1. The following abbreviations are used in Figure 2.1.

DAP - Days after planting

DOY - Day of year

2.4.2 Graphical output

Figure 2.2 shows the soil water balance graph. Line A is the recommended allowable depletion based on the percentages selected for the various growth stages in the crop data base.

The end of simulation deficit to field capacity is displayed in the top left hand corner. The seasonal components of the soil water balance are summarized below the graph. An indication is also given of crop stress, rooting depth and the estimated time till the next irrigation. Mass balance error should be zero unless simulated water contents were updated with measured values during the season. The simulated soil water content line turns red on a colour screen if the specified crop stress index is not reached.

Field A2 Water Balance

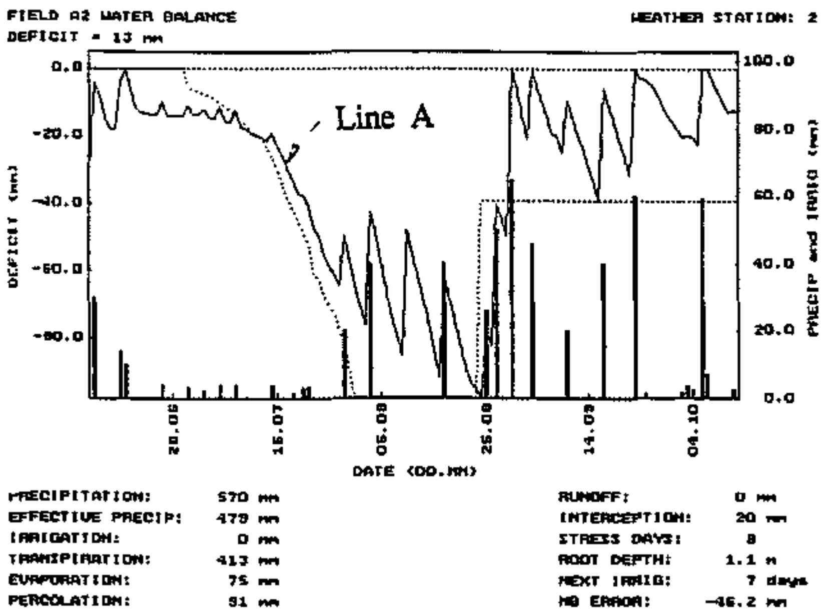


Figure 2.2 Soil water balance

LAI	-	Leaf area index (m^2 leaf m^{-2} soil)
RD	-	Root depth (m)
TDM	-	Total dry matter (t ha^{-1})
HDM	-	Harvestable dry matter (t ha^{-1})
Precip	-	Precipitation (mm)
Irrig	-	Irrigation (mm)
Evap	-	Evaporation (mm)
Trans	-	Transpiration (mm)
Drain	-	Drainage (mm)
Inter	-	Interception (mm)
PET	-	Potential evapotranspiration (mm day^{-1})
SI	-	Stress index
MBError	-	Water mass balance error (mm)

Farm: A1
Field number: 1
Crop ID: 1
Plant date: 28.04.93
Start date: 25.04.93
PWP: -1500
FC: -10
Field size: 10.00 ha
Weather ID: 1
Schedule: Yes
Description: GEDEELTE 1

DATE	DAP	DOY	LAI	RD	TDM	HDM	DEFICIT	Precip	Irrig	Evap	Trans	Drain	Inter	RunOff	PET	SI	MBError
25.04.93	0	115	0.0	0.00	0.000	0.000	30	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0	0.0
26.04.93	0	116	0.0	0.00	0.000	0.000	35	0.0	0.0	4.2	0.0	0.0	0.0	0.0	4.2	0	0.0
27.04.93	0	117	0.0	0.00	0.000	0.000	37	0.0	0.0	1.9	0.0	0.0	0.0	0.0	1.9	0	0.0
28.04.93	0	118	0.0	0.00	0.000	0.000	38	0.0	0.0	1.2	0.0	0.0	0.0	0.0	1.2	0	0.0
.																	
.																	
17.08.93	111	229	0.0	2.02	1.146	0.753	311	0.0	0.0	0.2	0.2	0.0	0.0	0.0	7.2	1	0.0
18.08.93	112	230	0.0	2.02	1.146	0.753	312	0.0	0.0	0.2	0.0	0.0	0.0	0.0	7.3	1	0.0
19.08.93	113	231	0.0	2.02	1.146	0.753	312	0.0	0.0	0.1	0.0	0.0	0.0	0.0	6.4	1	0.0
20.08.93	114	232	0.0	2.02	1.146	0.753	312	0.0	0.0	0.1	0.0	0.0	0.0	0.0	6.0	0	0.0

Cumulative precipitation: 118 mm
Cumulative irrigation: 0 mm
Cumulative evaporation: 108 mm
Cumulative transpiration: 277 mm
Cumulative drainage: 0 mm
Cumulative interception: 15 mm
Cumulative runoff: 0 mm
Cumulative ETD: 384 mm
Number of stress days: 9 days
Effective rain: 118 mm

Figure 2.1 Sample text output

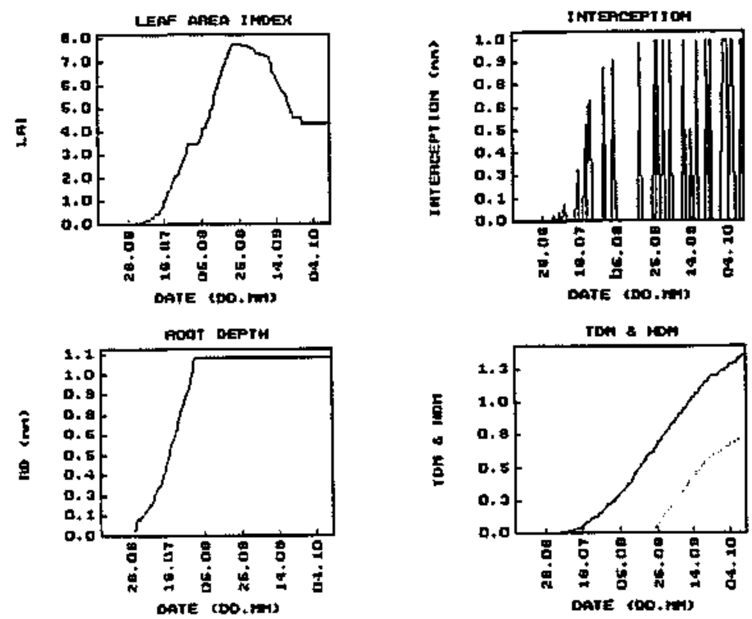


Figure 2.3 Crop growth and canopy water interception

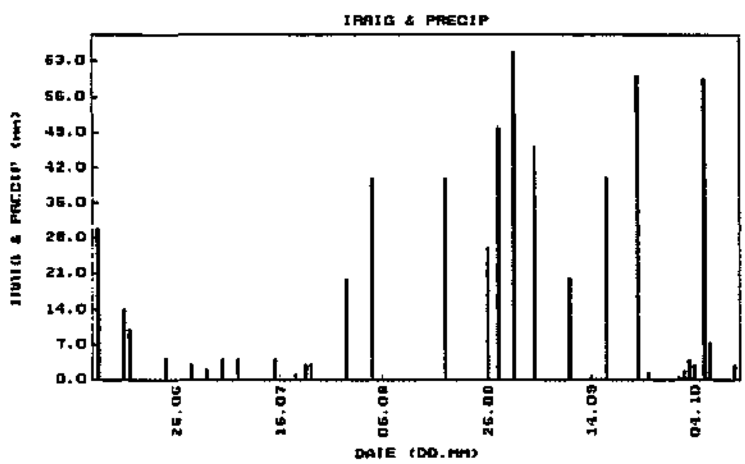


Figure 2.4 Precipitation and irrigation

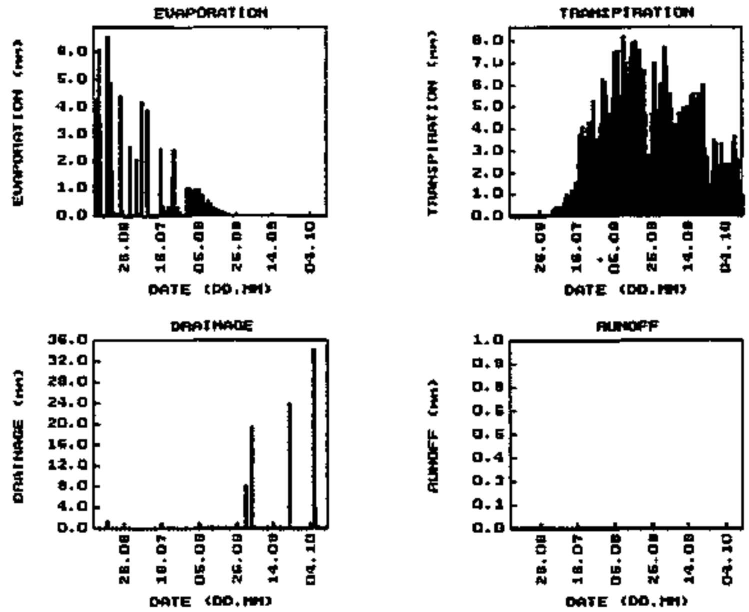


Figure 2.5 Soil water Evaporation, Transpiration, Drainage and Runoff

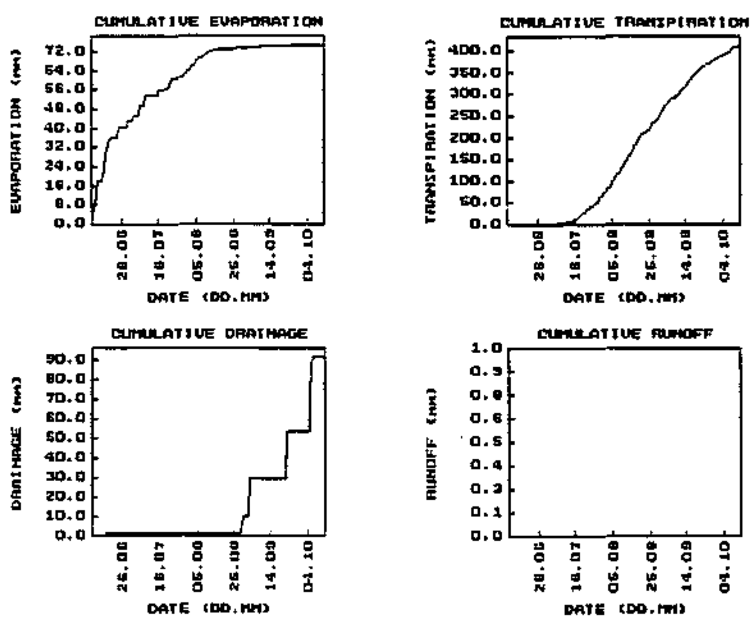


Figure 2.6 Cumulative water losses

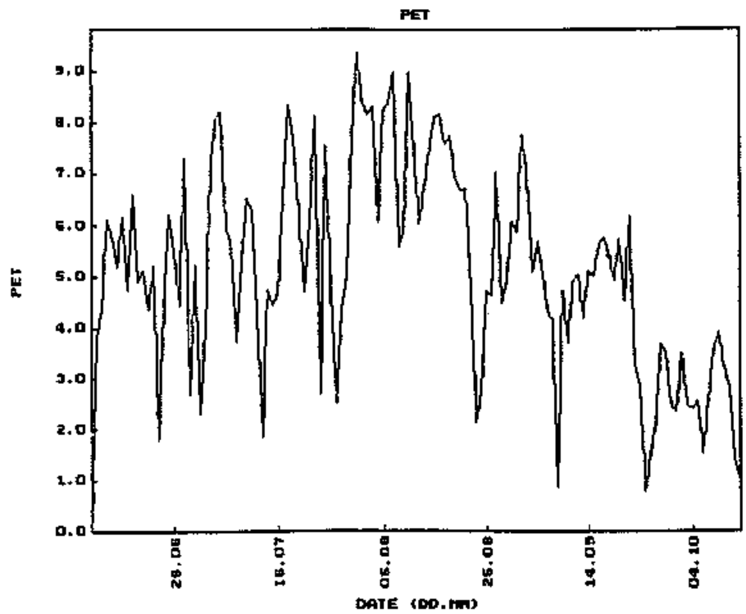


Figure 2.7 Daily Potential evapotranspiration (mm day^{-1})

3 CONCLUSIONS

The successfull completion of any the project depends on how well the aims were met or not.

Once again the aims of this research project were:

a) To integrate a crop irrigation scheduling program such as that developed by the Department of Agrometeorology at the University of the Orange Free State into a water office data base developed at the Rand Afrikaans University;
b) To collect climate, soil and crop data at Loskop irrigation scheme for testing the irrigation scheduling program;
c) To implement the irrigation scheduling program at Loskop irrigation scheme as part of an optimisation system to minimize water losses on irrigation schemes;
d) To generalise the results to an optimisation system that could be used on all major irrigation schemes in South Africa.

The test site was changed from Loskop Irrigation Scheme to Brits because:

- it had to be visited regularly and it was much closer;
- of the availability of suitable personnel to look after the weather stations;
- the availability of suitable crops.

According to the aims of the project it was successful for the following reasons:

- The Soil Water Balance program, SWB, was successfully integrated into the Water Administration System, WAS;
- The Soil Water Balance program runs successfully with climate, soil and crop data that were collected at Brits;
- An irrigation scheduling model is now part of an optimisation system that can be implemented incrementally depending on the requirements of the user;
- Water requested by farmers without the help of an irrigation scheduling program can be compared with volumes calculated by the scheduling program. Comparison

between requested and calculated volumes for a total irrigation scheme are now possible;

- Considerable time can be saved and the possibility of errors reduced as tedious and repetitive calculations are automated;
- Better water loss control that will lead to savings in water will be possible.
- Optimum yields and water savings by farmers are now possible.

3.1 FUTURE DEVELOPMENT

Although the Soil Water Balance program, SWB, was successfully integrated into the Water Administration System, WAS, its usability is hampered by a lack of crop parameters for a range of different crops. Crop parameters are currently only available for peas, maize and tobacco. A follow up research project has fortunately been approved to rectify this shortcoming and to refine the model into an even more useful tool that could be used by irrigators, advisors, researchers and managers.

The follow up project will attempt to determine parameters for the following crops:
Agronomic: Small grains, maize, paprika, ground nuts, tobacco, sugar cane, cotton and dry beans.
Pasture: Rye grass, lucerne, fodder sorghum and kikuyu.
Vegetable: Potatoes, factory and table tomatoes, green peas, sweet potatoes, onions, garlic, cruciferae, carrots and cucurbits.
Fruit: Deciduous, citrus, tropical, subtropical and nuts.

Where possible, existing data from researchers country wide will be used to develop the required parameters. This task would be a lot easier if the WRC could standardize their data archiving requirements for their research projects in a fully relational data base.

REFERENCES

ANNANDALE, JG, VAN DER WESTHUIZEN, AJ & OLIVIER, FC 1995: Die fasilitering van tegnologieoordrag deur verbeterde besproeiingsriglyne vir groente en 'n meganistiese gewasmodeleringsbenadering. Verslag aan die Waternavorsingskommissie, Verslag 476/1/95 Johannesburg: Universiteit van Pretoria.

BENADé, N; ENGELBRECHT, RJ & ANNANDALE, GW 1990: Die optimalisering van die bedryf van besproeiingskanaalstelsels. Verslag aan die Waternavorsingskommissie, Verslag 176/1/90 Johannesburg: Randse Afrikaanse Universiteit.

BENADé, N 1994: Die ontwikkeling van 'n gerekenariseerde bestuurstelsel vir die optimale bedryf van besproeiingskanaalstelsels. Verslag aan die Waternavorsingskommissie, Verslag 367/1/93 Johannesburg: Randse Afrikaanse Universiteit.

BENADé, N 1994: Geoptimeerde lewering van besproeiingswater op aanvraag deur kanaalstelsels. PhD proefskrif Johannesburg: Randse Afrikaanse Universiteit.

CAMPBELL, GS & DIAZ, R 1988: Simplified Soil Water Balance Models to predict crop transpiration. In Drought research priorities for the dry land tropics (BIDINGER, FR & JOHANSEN, C eds) PARANCHERU, AP 502 324, India ICRISAT.

MOTTRAM, R & de JAGER, JM 1994: Research on maximising irrigation project efficiency in different soil-climate-irrigation situations. Report to the WRC, No 226/1/94.

APPENDIX A SWB USER'S GUIDE

Soil Water Balance

SWB

User's Guide

1995

Table of contents

List of figures

1 INTRODUCTION

The program, SWB is a soil water balance model developed to improve irrigation scheduling on farms. SWB is part of an optimisation system that is specifically developed for irrigation schemes. The aims of the optimisation system are to

- improve water management
- improve productivity
- improve crop yields
- provide a better service to farmers
- save water
- save money

The optimisation system consists of a simulation model (PROCAN), a water administration model (WAS or RAUDB), a water release calculation model (SigmaQ) and a soil water balance model (SWB).

1.1 Unsteady canal flow model (PROCAN)

The aim of the unsteady canal flow model is to simulate unsteady flow in canals and rivers.

PROCAN can be used for:

- calculating lag times;
- calculating evaporation and seepage losses;
- calculating hydrographs;
- evaluating different water distribution scenarios;
- training canal operators;
- designing new canals;
- analyzing existing canals.

Detailed information on PROCAN is given in the PROCAN user's manual. Dr. N. Benadé at Fischer & Associates can be contacted for any information.

1.2 Water administration system (WAS)

WAS was originally developed as a data input utility for PROCAN, but it later developed into a fully computerised water administration system that can be used for

- calculating water balance statements;
- calculating water reports;
  calculating water releases into all canals, taking lag times and water losses into account;
- printing distribution sheets ("leikaarte");
- printing water bailiff instruction sheets ("kanaalwagopdragte");
- printing water releases.

Detailed information on WAS is provided in the WAS User's manual. Dr. N. Benadé at

Fischer & Associates can be contacted for any information.

1.3 Irrigation scheduling model (SWB)

The irrigation scheduling model, SWB, can be used on a single farm, a group of farms or all farms on an irrigation scheme. The advantage of calculating the water required for all the crops on an irrigation scheme is that the water requested by the farmers can be compared to the actual water required. If there is a big difference between the water required and the water requested, steps can be taken to improve irrigation scheduling on a specific irrigation scheme.

SWB is versatile for the following reasons:

- The soil water content can be updated during the growing season;
- The soil water balance can be calculated for any number of fields sequentially and displayed graphically;
- The program is user friendly and therefore easy to use;
- Maximum and minimum temperatures can be used to estimate PET if Penman-Monteith reference ET is unavailable;
- Password protection, error checking and on line help is available;
- Water order forms are automatically completed;
- Measured data can easily be compared with simulated values;
- Extensive use is made of graphical output;
- The data files are compatible with the Paradox data base and therefore Paradox can be used to manipulate the data;
- Records for irrigation management are stored;
- Next irrigation date is forecast.

1.4 Hardware requirements

SWB requires an IBM compatible computer running PC-DOS or MS-DOS version 3.00 or later. The program requires a hard disk to make full use of its capabilities. A numeric coprocessor is not required but if one is available, it will greatly improve calculation speed. If a graphics adapter is available, the program will use it to display graphics information. The CGA, MCGA, EGA, VGA, Hercules and 3270 PC graphic adapters are supported, but colour will only be used on EGA and VGA adapters. If EMS memory is available, it will be used to load overlay files, resulting in faster overlay swapping. The EMS driver must conform to the Lotus/Intel/Microsoft Extended Memory Specification.

2 USING THE PROGRAM

2.1 Installing the program

The program is distributed on a single floppy disk. Before the program can be used, it must be installed on the computer on which it will be used. Please keep the distribution disk in a safe place after installing the program.

Due to the size of the program and the disk space required for simulations, the program can only effectively be used on hard disk systems. The following procedure can be used to install the program on hard disk systems.

A subdirectory must first be created off the root directory. At the DOS prompt on the selected hard disk, type the commands

```
CD\
MD SWB
CD SWB
```

Insert the SWB distribution disk into drive A and type the command

```
COPY A:*.*
```

The program is now installed for use on a hard disk system.

2.2 Loading the program

The program can be loaded by entering the following command at the DOS prompt on the correct drive and directory

```
SWB
```

The following command line parameter can be used: /BW. Command line parameters are typed after SWB, separated by a space.

/BW forces the program to use a black and white display. This may be necessary if a LCD screen is used or if a colour graphics adapter is used with a composite monitor. The program will also use a black and white display if the command

```
MODE BW80
```

was typed at the system prompt before loading the program. To force the program to use a colour display, type the command

```
MODE CO80
```

at the system prompt before loading the program.

Note: The MODE BW80 and MODE CO80 commands cannot be used on systems with a monochrome or Hercules display adapter.

2.3 Files

SWB is developed using Borland Pascal 7.0 and the Paradox Engine 3.1. The only file that is needed when starting for the first time is SWB.EXE. All the data and index files, that is files with the extension *.DB and *.PX, can be created by SWB.EXE. To manipulate (ie. to add data from one table to another table) the Paradox data files, the standard Paradox database version 4.0 can be used.

The program uses different files. A short description of each file is given.

Irrigation scheduling files

SWB.EXE	SWB executable file.
CROP.DB	Crop parameters Paradox data file.
FIELD.DB	Field data Paradox data file.
MEASURED.DB	Measured data Paradox data file.
PRECIP.DB	Precipitation and irrigation Paradox data file.
RESULT.DB	Simulation results Paradox data file.
SOIL.DB	Soil parameters Paradox data file.
WCORR.DB	Soil water content correction Paradox data file.
WDAY.DB	Daily weather parameters Paradox data file.
WEATHID.DB	Weather station identification data Paradox data file.

Water management files

AANKOOP.DB	Water sales data file.
AANVR.DB	Water requests data file.
BALANS.DB	Water balances data file.
INLYS.DB	Field sizes data file.
LOSS.DB	System losses data file.
METER.DB	Water meters data file.
OORPL.DB	Water transfers data file.

*.CFG	Program configuration files.
*.PX	Paradox primary index files.
*.BGI	Borland graphics device driver files.

TUTILITY.EXE	Utility program that is used to fix corrupted Paradox data files.

Primary index files

For SWB to work satisfactory all *.DB files need a corresponding *.PX file. Data can be transferred from one computer to another by only copying the *.DB files that contain all the data. To recreate the index files on the other computer the **build index** procedure must be executed as described in section 3.

It is also possible to get a corrupted index file due to a power failure or by switching off the computer while you are still in SWB. In such a case the index files must be deleted and rebuilt again. In the unfortunate case where the data was also effected use the **TUTILITY.EXE** program to rectify the problem.

2.4 Main screen display

After loading the program, the title screen will be displayed as shown below. Pressing a key will prompt the user to verify the computers current date. The Screen consists of three parts: the main menu at the top, the status line on the second line and the message line at the bottom of the screen.

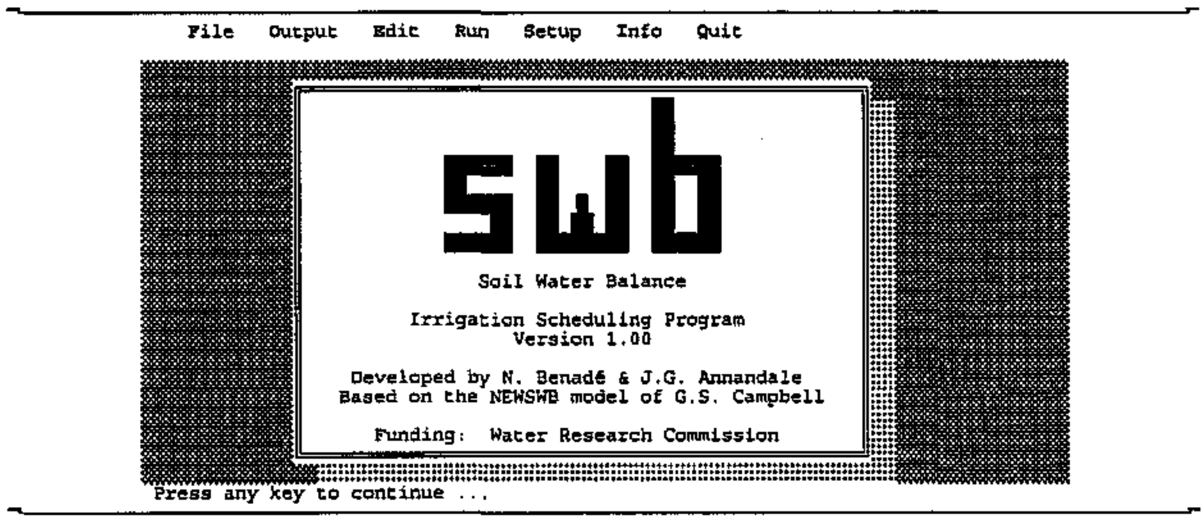


Figure 2.1 Introductory screen

Main menu
The main menu is the key to the operation of the program, as all functions are selected from the main menu or from one of its sub-menus.

Status line
The status line displays the current date or information about the current Paradox data file.

Field 15 of 100: Indicates that there is 100 records in the Field table and that you are at record number 15.

Message line
The message line displays messages to the user. During normal operations the line displays the keys that can be pressed with their associated functions. During execution of some procedures, the message line displays a description of the function.

2.5 Using the environment

An environment consisting of a menu system and dialogue boxes is implemented in the program. The aim is to make the program as easy as possible to operate.

Menu system
A menu system consisting of horizontal, pull-down and pop-up menus is used. Pull-down menus are accessed from horizontal menus, while pop-up menus are in turn accessed from pull-down menus. The following basic actions are used to navigate through the menu system:

- Use the highlighted letter to choose a menu item or use the arrow keys to move to the item and press the *Enter* key.
- Press the *Esc* key to leave a menu.
- Use the *Right* and *Left* arrow keys to move from one pull-down menu to another.

Dialogue boxes
The program uses dialogue boxes to communicate with the user. There are three kinds of dialogue boxes: input boxes, verify boxes and error boxes.

Input boxes
The user must supply input in the box on the screen. There are three types of input: integer values, floating point values and text strings. Integer values must be entered in the usual format: 123 or -13. Floating point values can be entered in normal or scientific notation: 25.15 or 6.7E5. Text strings can be entered directly. Pressing the *Esc* key at any time will abort the input operation.

Verify boxes
The user is asked to verify a certain operation. Pressing the Y key will signal yes and pressing the N key will signal no. Pressing the *Esc* key will abort the operation.

Error boxes
If an error occurs, an error message is displayed in an error box. The user must then respond to the instructions given to return from the error situation. Appendix A contains a list of all possible error messages with possible solutions.

3 FILE MANAGEMENT

```
File
┌─────────────────┐
│ Directory       │
│ Build database  │
│ Empty database  │
└─────────────────┘
```

The file pull down menu offers choices regarding displaying directories, building and emptying a database.

3.1 Directory

Displays the directory and files specified.

3.2 Build database

Creates non existing Paradox *.DB and *.PX files.

3.3 Empty database

Deletes all data from *.DB files. If you want to delete specific data files you can delete the corresponding *.DB and *.PX files and build the database again. Empty *.DB and *.PX files will then be created in the places of the ones deleted.

Note: If you want to delete a massive amount of data, say for example the historical records of a specific weather station, it is better to use the Paradox database it self rather than deleting records one by one using the SWB program. This option is only for people who own Paradox version 4.0 and knows how to use it.

4 EDITING

4.1 IRRIGATION SCHEDULING

```
Edit
┌──────────────────────┐
│ Weather data       ► │
│ Field data           │
│ Soil               ► │
│ Crop parameters      │
│ Measured data        │
└──────────────────────┘
```

The Edit Schedule menu allows the user to edit and display weather, field, soil, crop and measured data. All input and output is done from this menu.

The keys that are used for editing any table in SWB are the following:

Insert: Inserts a new record. Save the new record by pressing the F2 key.

Delete: Deletes a record. The password to delete records from certain tables is **Delete** with a capital letter D.

F2: Saves a newly inserted record, a record that has been changed or initiate the find procedure.

F3: Changes a record. The password to change records in certain tables is **Change** with a capital letter C. Save the changed record by pressing the F2 key.

F4: Finds a record. Initiate the find by pressing the F2 key.

F7: Outputs the data in table format to the screen or printer. Tables that cannot be printed or viewed in table format are the Crop, Soil, Water content update and Measured data table.

PgUp: Moves one record forward.
PgDn: Moves one record backward.

4.1.1 Weather data input

```
Weather data

Irrig & Precip
Weather data
Stations
```

This menu allows the user to edit daily irrigation, precipitation and weather data. Irrigation, precipitation or weather data can only be inserted for weather stations that exist in the weather station data file. To insert a new weather station refer to section 4.1.1.3.

4.1.1.1 Irrig & Precip

For a simulation to run successfully it needs the daily irrigation and precipitation, even if it is zero. The user can enter the data before starting a simulation or start the simulation and let the program prompt the user to enter the missing data. The second option is recommended.

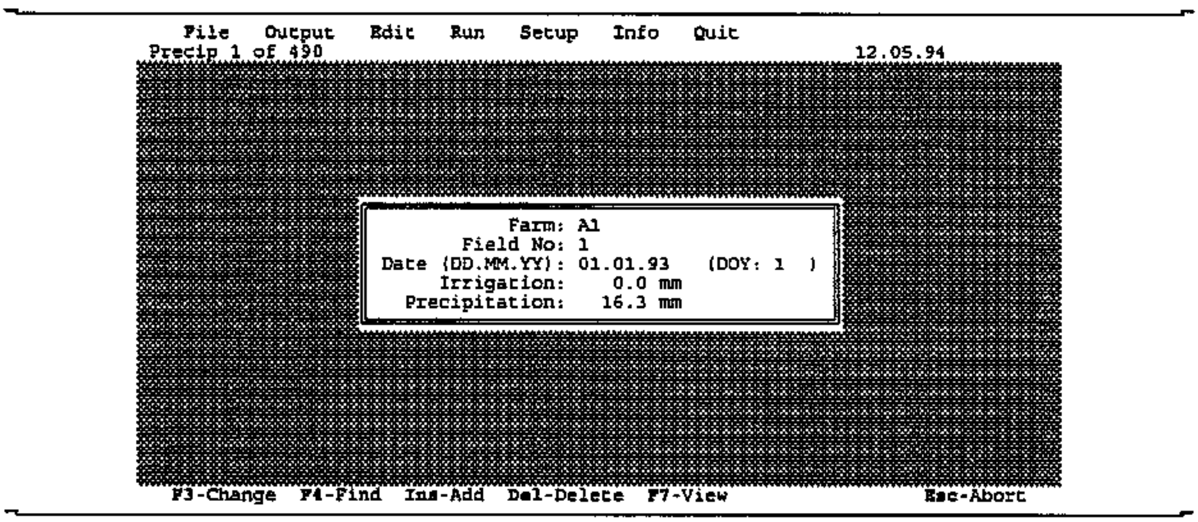


Figure 4.1 Daily irrigation and precipitation data

Farm: Farm identification.
Field No: Field number on the specific farm.
Date (DD.MM.YY): Date of irrigation and/or precipitation.
Irrigation: Irrigation in mm.
Precipitation: Precipitation in mm.

4.1.1.2 Weather data

For a simulation to run successfully it needs daily weather data. If the PET and VPD are zero the program uses the maximum, minimum temperature and solar radiation to calculate an estimation. On the other hand, if PET and VPD has non zero values the program will use it and ignore the rest. The user can enter the data before starting a simulation or start the simulation and let the program prompt the user for the missing data. The second option is recommended.

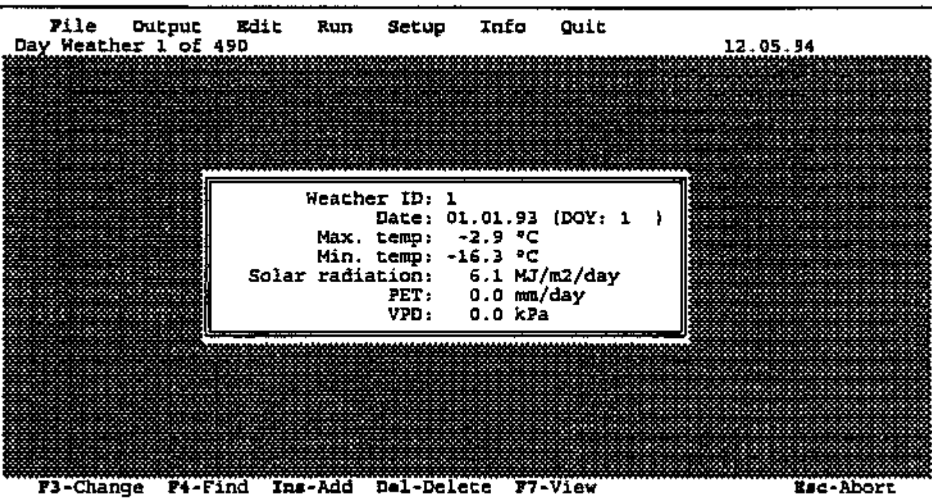


Figure 4.2 Daily weather data

Weather ID:	Weather station identification as specified in the weather station table.
Date:	Date for weather data.
Max. temp:	Maximum temperature in degrees Celsius.
Min. temp:	Minimum temperature in degrees Celsius.
Solar radiation:	Solar radiation in MJ m^{-2} day^{-1}.
PET:	Potential evapotranspiration in mm day^{-1}
VPD:	Vapour pressure deficit in kPa.

4.1.1.3 Stations

Daily weather data for a specific weather station can only be entered in the weather data table if the weather station is listed in the weather station table. The data needed to enter a weather station is described below.

A weather station is linked to a specific field via the Weather ID of the specific weather station. This link is described in section 4.1.2.

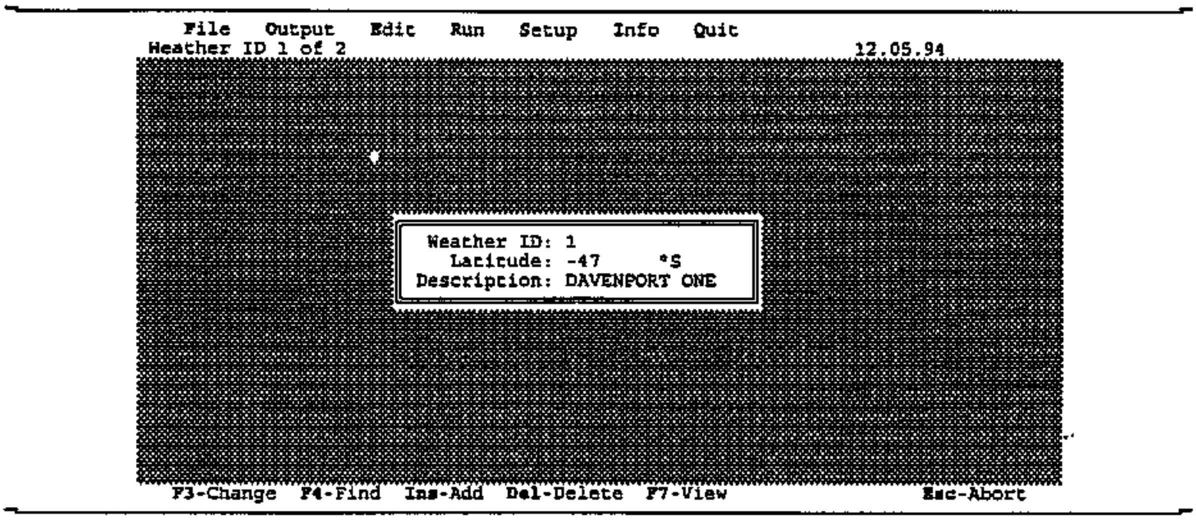


Figure 4.3 Weather stations table

Weather ID:	Weather station identification.
Latitude:	Latitude in degrees south.
Description:	A 15 character description of the weather station.

4.1.2 Field data input

Every field that the user wants to simulate must be entered in the field table. There must be a link to the type of crop planted on the specific field as well as a link to the weather station that measures the weather parameters. A field which **scheduled flag** (see the input form in Figure 4.4) is set to **No**, will be ignored when simulations for a **group** of fields or simulations for **all** fields is done.

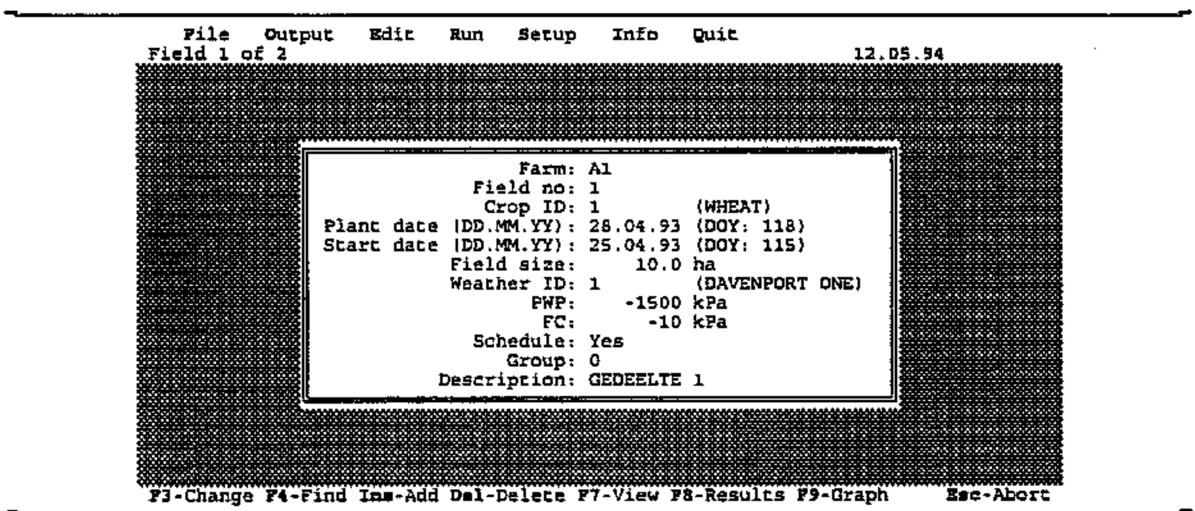


Figure 4.4 Field data

Farm:	Farm identification.
Field no:	Field number on the specific farm.

Crop ID:	Crop identification number as specified in the crop table described in section 4.1.4.
Plant date:	Crop plant date in DD/MM/YY.
Start date:	Simulation starting date in DD/MM/YY.
Field size:	Field size in ha. Is used for calculating water volumes.
Weather ID:	Weather station identification number as specified in the weather station table described in section 4.1.1.3.
PWP:	Permanent wilting point in kPa.
FC:	Field capacity in kPa.
Schedule:	Flag set Yes/No to schedule or not.
Group:	A group number to simulate a group of fields.
Description:	Description of the specific field.

To view a graphical representation of the simulation results of a specific field, first find the field by using the PgUp, PgDn or F4 keys and second press the F9 key to display a list, as shown below, of graphs that can be displayed. The last option on the list will specify if the output will be send to the screen or printer.

The size of the graphic output to the printer can be controlled with the **Setup** menu.

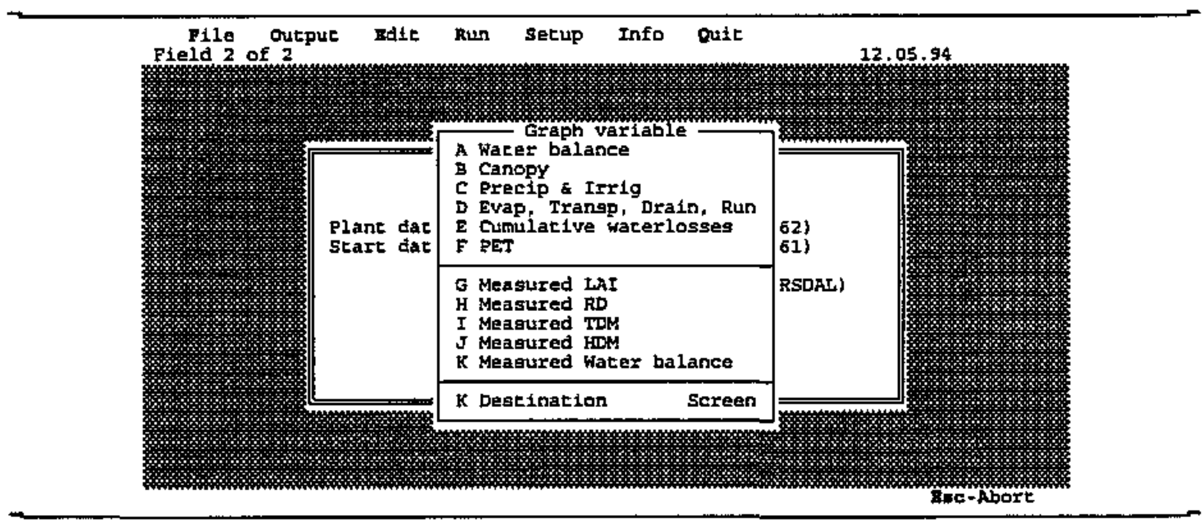


Figure 4.5 Graphic output list

4.1.3 Soil data input

```
Soil
Initial data
Water content
```

This option allows the user to edit the soil parameters and to update the soil water content of a specific field.

4.1.3.1 Initial data

Initial data for 11 soil layers is needed. Even if the soil type is identical throughout the soil profile, it must be divided into 11 different layers and entered into the soil data table as shown in Figure 4.6.

File Output Edit Run Setup Info Quit

Soil 1 of 2 12.05.94

Farm: A1
Field No: 1
Runoff No: 1000

Layer	Depth from (m)	Depth to (m)	Initial WC (m/m)	FC (m/m)	PWP (m/m)	Bulk dens (g/cm3)
1	0.000	0.100	0.20	0.26	0.09	1.10
2	0.100	0.190	0.20	0.26	0.09	1.20
3	0.190	0.290	0.20	0.26	0.09	1.20
4	0.290	0.390	0.21	0.26	0.09	1.30
5	0.390	0.490	0.22	0.26	0.09	1.50
6	0.490	0.690	0.25	0.26	0.09	1.40
7	0.690	0.890	0.25	0.26	0.09	1.40
8	0.890	1.290	0.28	0.28	0.09	1.40
9	1.290	1.690	0.28	0.28	0.09	1.40
10	1.690	2.090	0.28	0.28	0.09	1.40
11	2.090	2.490	0.28	0.28	0.09	1.40

F3-Change F4-Find Ins-Add Del-Delete Esc-Abort

Figure 4.6 Initial soil data

Farm:	Farm identification.
Field No:	Field number on the specific farm. A farm and field number that is not listed in the field table, will not be excepted.
Runoff No:	Runoff curve number.
Depth to:	Soil layer depth to the end of the layer in m.
Initial WC:	Initial soil water content at the start of the simulation in m/m.
FC:	Field capacity in m/m.
PWP:	Permanent wilting point in (m/m).
Bulk dens:	Bulk density in g/cm^3

4.1.3.2 Water content

If the water content of the soil profile is measured during the growing season, it can be updated by entering the correct water content into the water content correction table as shown in Figure 4.7.

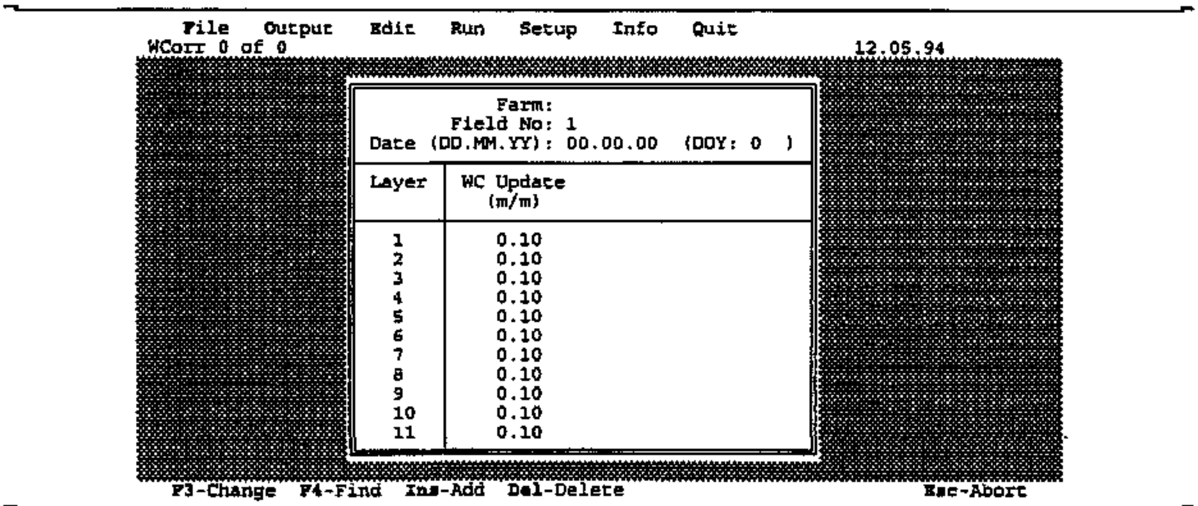


Figure 4.7 Water content correction data

Farm:	Farm identification.
Field No:	Field number on the specific farm. A farm and field number that is not listed in the field table, will not be excepted.
Date:	Date of update, DD/MM/YY.
WC Update:	Update of water content of layers 1 to 11, m/m.

4.1.4 Crop parameters

All the different crop types and their corresponding parameters are listed in the crop type table. The parameters in the crop type table are only accessible by means of a password. A password is used to protect the user from accidently changing or deleting a crop record. The password to change crop type parameters is the word **Change** with a capital letter C. The password to delete a crop type record is the word **Delete** with a capital letter D.

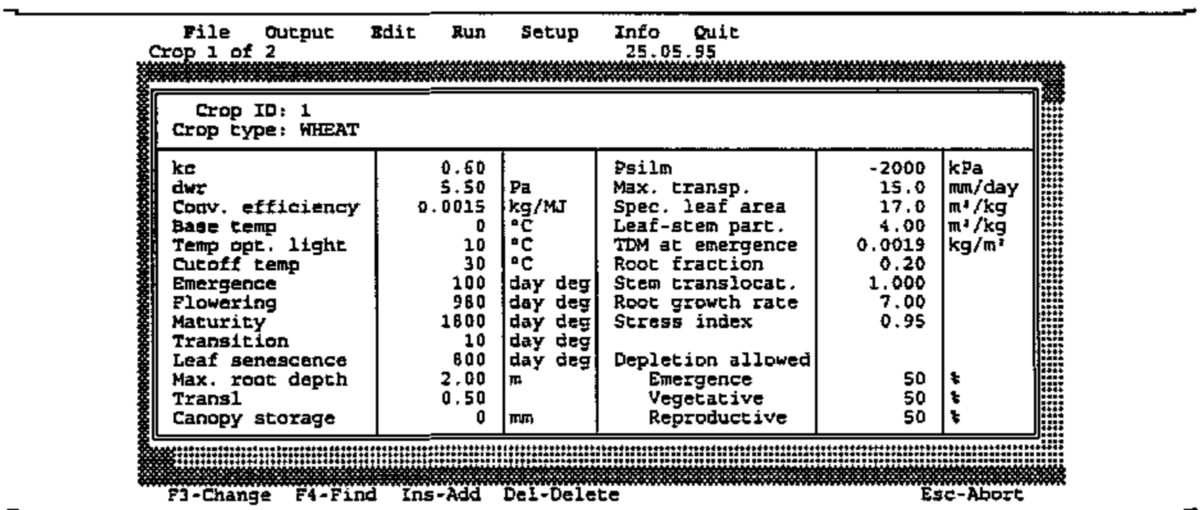


Figure 4.8 Crop type parameters

Crop ID:	Crop identification reference number.

Crop type:	Crop type description in words.
kc:	Average daily transmission coefficient.
dwr:	Dry matter water ratio in Pascals.
Conv. efficiency:	Radiation conversion efficiency, kg/MJ.
Base temp:	Base temperature in degrees celsius.
Temp opt. light:	Temperature for optimum light-limited growth in degrees celsius.
Cutoff temp:	Cutoff temperature for growing day degrees in degrees celsius.
Emergence:	Emergence day degrees.
Flowering:	Day degrees at the end of the vegetative growth.
Maturity:	Maturity day degrees.
Transition:	Transition period (day degrees) from vegetative to reproductive growth.
Leaf senescence:	Leaf senescence day degrees.
Max. root depth:	Maximum rooting depth, m.
Transl:	Fraction of SDM transl. to heads during grain fill.
Psilm:	Leaf water potential at maximum transpiration.
Max. transp.:	Maximum possible transpiration rate, mm/day.
Spec. leaf area:	Specific leaf area, m^2/kg.
Leaf-stem part:	Leaf-stem partition parameter.
TDM at emergence:	Canopy dry matter at emergence.
Root fraction:	Fraction of dry matter partitioned to roots.
Stem translocat.:	Potential grain production as fraction of SDM at anthesis.
Root growth rate:	Root growth rate.
Stress index:	Stress index.
Emerge. depl.:	Percentage depletion at emergence.
Vegetate. depl.:	Percentage depletion at the vegetation stage.
Repro. depl.:	Percentage depletion at the reproductive stage.
Canopy intercept.:	Canopy interception, mm.

4.1.5 Measured data

Measured data can easily be compared to the corresponding simulated values by entering it into the measured data table. A graph of the comparison can be viewed as described in section 4.2.

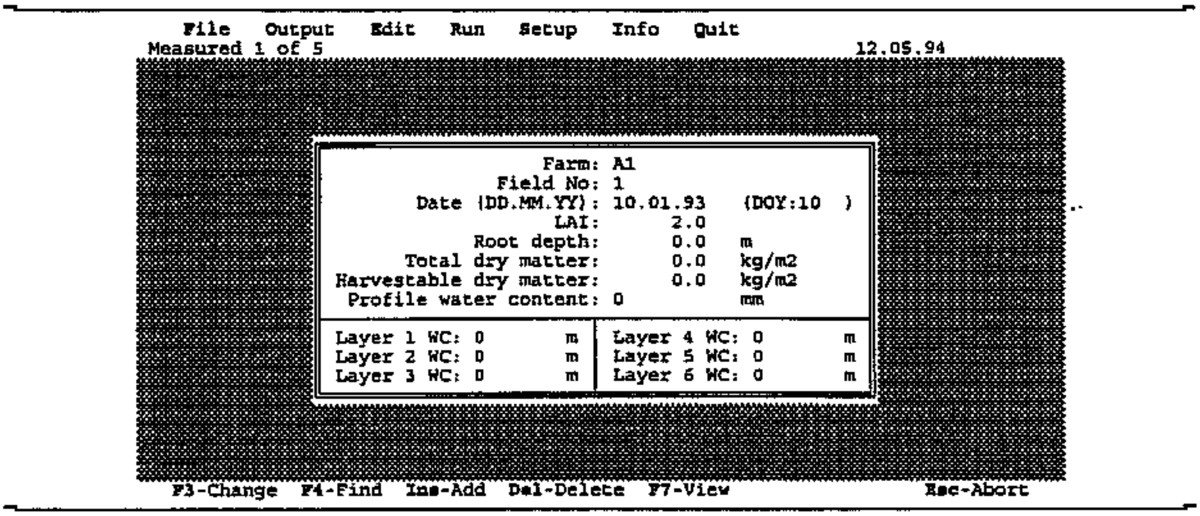


Figure 4.9 Measured data

Farm:	Farm identification.
Field No:	Field number on the specific farm.
Date:	Date of measurement, DD/MM/YY.
LAI:	Leaf area index.
Root depth:	Root depth, m.
Total dry matter:	Total dry matter, kg/m^2.
Harvestable dry matter:	Harvestable dry matter, kg/m^2
Profile water content:	Profile water content, mm.
Layer 1 to 6 WC:	Water content of layers 1 to 6, m.

4.2 WATER MANAGEMENT

```
Water

A Requests
B Transfers
C Loss parameters
D Measured volumes
E Water sales
F Water balances

G Quota awords
```

The Edit Water menu allows the user to edit and display water requests (Figure 4.10), water transfers between fields, field loss parameters, measured volumes, water sales, water balances for each field and quotas can be awarded to each field.

The keys that are used for editing any table in the Water menu is the same as in the Schedule menu.

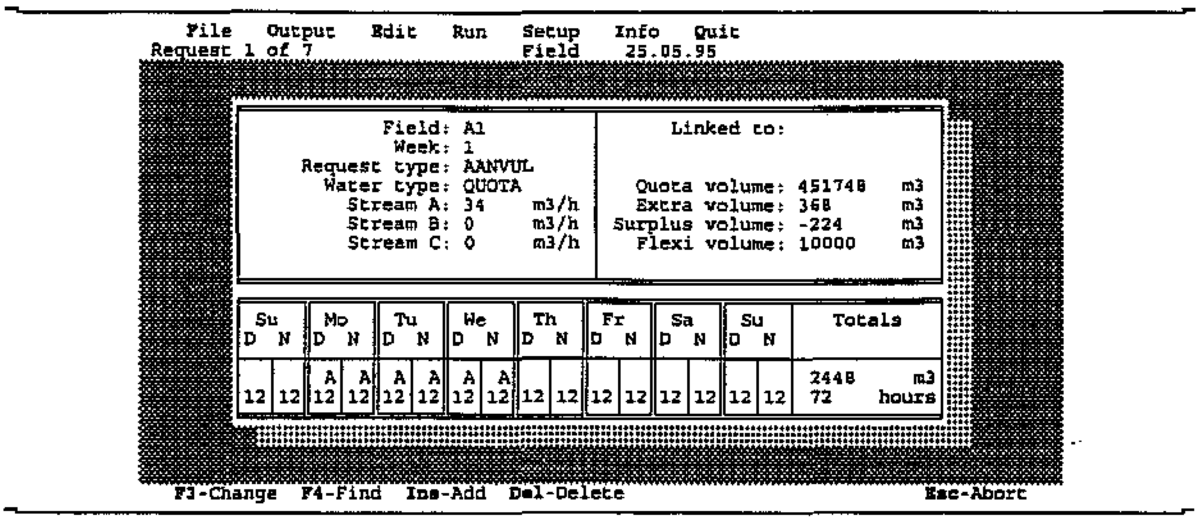


Figure 4.10 Water request form

5 RUN SIMULATION

```
Run

Single field
Group of fields
All fields
```

The **Run** menu allows the user to do simulations for a single field, a group of fields or for all the fields in its database. A simulation will be terminated if there is a lack of data, if the current date equals the specified end date or when the crop reached maturity.

5.1 Single field

This option calculates the soil water balance for a single field. The program prompts the user for the farm number, the field number, the starting date and the end date. The starting date is by default the starting date that is specified in the Field table. The end date is by default the current date on the computer. Both dates can be modified by the user. The program will prompt the user for the relevant irrigation, precipitation and weather data inputs if necessary.

If the was run successfully the soil water balance graph of the specific field will be displayed on the screen.

5.2 Group of fields

This option calculates the soil water balance for a group of fields. The program will prompt the user for the group number that identifies a specific group. The program also prompts the user for the starting date and the end date. The starting date is by default the starting date that is specified in the Field table. The end date is by default the current date on the computer. Both dates can be modified by the user. The program will prompt the user for the relevant irrigation, precipitation and weather data inputs if necessary.

If the simulation was run successfully the results for every field can be viewed in text or graphics format under the **Field data** option under the **Edit** menu, as described in paragraph 4.1.2.

5.3 All fields

This option calculates the soil water balance for all fields. The program prompts the user for the starting date and the end date. The starting date is by default the starting date that is specified in the Field table. The end date is by default the current date on the computer. Both dates can be modified by the user. The program will prompt the user for the relevant irrigation, precipitation and weather data inputs if necessary.

If the simulation was run successfully the results for every field can be viewed in text or graphics format under the **Field data** option under the **Edit** menu, as described in paragraph 4.1.2.

6 SETTING UP THE PROGRAM

```
Setup
┌────────────┐
│Printer   ► │
├────────────┤
│Load setup  │
│Save setup  │
└────────────┘
```

The **Setup** pull-down menu allows the user to configure the program for his printer.

6.1 Printer

Allows the user to configure the program for his printer in text or graphics mode. The default is adequate for most printers, but special features of printers can be utilized by using the printer setup option.

Text mode
Configures the printer in text mode.

Page length
The number of lines on a page.

Top margin
The number of lines to leave blank at the top of each page.

Bottom margin
The number of lines to leave blank at the bottom of each page.

Left margin
The number of columns to leave open at the left side of each page.

Wait after page
Determines whether the program must wait after each page has been printed. This is useful when manually inserting single sheets of paper.

Setup string
Specifies the string to be sent to the printer before each page is printed. Control characters and other unprintable characters are indicated by preceding the three digit decimal character code by the # character. The # character is indicated by ##. The string #015 is used to select compressed print on most printers. The printer manual can be consulted for further information regarding control codes.

Reset string
Specifies the string to be sent to the printer after the printing is completed. The reset string must have the same format as the set-up string. The string #027@ can be used to reset most printers.

Graphics mode
Configures the printer in graphics mode.

Top margin
The number of lines to leave blank at the top of each page.

Left margin
The number of columns to leave open at the left side of each page.

Graphics mode
Allows the user to select different graphics modes in dots per inch.

Printer
With this selection the user can toggle between an Epson and HP laserjet printer. The Epson selection will be suitable for most printers.

6.2 Load setup

Loads a program configuration from disk.

6.3 Save setup

Saves the current configuration to disk. If a configuration file with the same name as was specified exists, the action must be verified. If the configuration file is named SWB.CFG, it will automatically be loaded when the program is started.

7 PROGRAM INFORMATION

The Info menu gives information about the version number and the memory that is current available.

8 QUITTING THE PROGRAM

The last option on the main menu is used to quit the program.

9 ERROR MESSAGES

9.1 Graphics errors

Graphics errors are reported if an error occurs during a graphics operation.

(BGI) graphics not installed
The graphics system is not installed.

Device driver file not found
The device driver file cannot be found in the current directory. Copy all the .BGI files on the PROGRAM DISK to the current directory.

Graphics error
A general graphics error occurred.

Graphics hardware not detected
No graphics adapter is detected in the computer.

Graphics I/O error occurred
A graphics input/output error occurred.

Invalid device driver file
The device driver file is invalid. Replace all the .BGI files in the current directory with the .BGI files on the PROGRAM DISK.

Invalid graphics mode for selected driver
The graphics driver cannot handle the selected graphics mode.

Not enough memory to load driver
There is not enough memory available to load the device driver file.

9.2 Critical errors

A critical error occurs when DOS cannot complete an operation. The user is given the opportunity to retry or abort the operation. See the DOS Reference Manual for further information about the following critical errors:

Bad request structure length
CRC error in data
Device/Disk is write-protected
Device/Disk not ready
General failure
Printer out of paper
Read fault
Sector not found
Seek error

Unknown command
Unknown media type
Unknown unit
Write fault

9.3 Fatal errors

Fatal errors immediately terminate the program.

Division by zero
An attempt was made to divide by zero.

Floating point overflow
A floating point operation produced an overflow.

Floating point underflow
A floating point operation produced an underflow.

Heap overflow error
There is not enough free space on the heap to allocate a block of the required size.

Invalid pointer operation
An error occurred in the dynamic memory management.

Overlay manager not installed
The program is attempting to call an overlay, but the overlay manager is not installed.

Overlay file read error
An error occurred when the overlay manager tried to read an overlay.

Range check error
This error is reported when an attempt was made to assign an out of range value to a variable.

Stack overflow error
There is not enough stack space available to allocate a subprogram's variables.

9.4 Other errors

The following additional errors can occur during the execution of the program. It includes critical errors described in Appendix 9.2.

Disk read error
An attempt is made to read past the end of a file.

Disk write error
The disk being written to is full.

File access denied
This error is reported if an attempt is made to write to a read-only file, to read from a file not open for reading or to write to a file not open for writing.

File not assigned
A name has not been assigned to the file.

File not found
The file name does not specify an existing file.

File not open
An attempt is being made to perform an operation on a file that is not open.

File not open for input
The file is not open for input.

File not open for output
The file is not open for output.

Illegal format code
The numeric format code is invalid. See Section 2 for the correct format.

Illegal printer code string
The printer code string is invalid. See Section 6 for the correct format.

Invalid drive number
The specified drive number is invalid.

Invalid file access code
The file mode value of a file is invalid.

Invalid file handle
An invalid file handle was passed to a DOS system call.

Invalid numeric format
A numerical value read from a file does not conform to the proper numeric format.

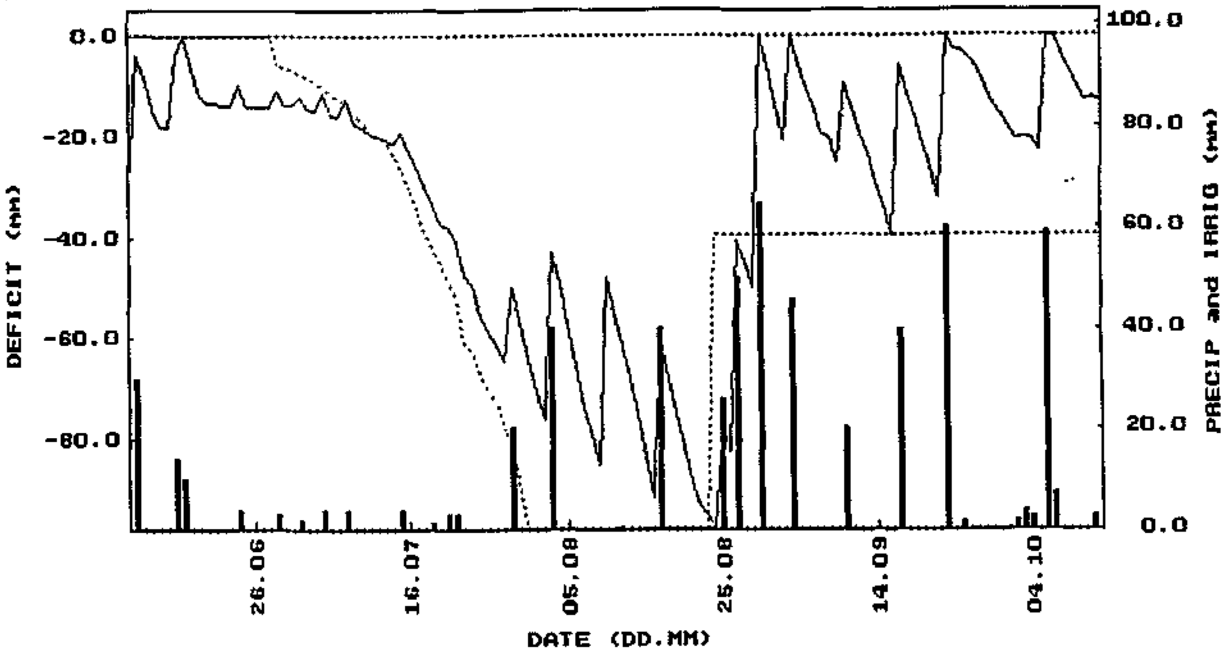
Path not found
The path name is invalid or specifies an non existing subdirectory.

Too many open files
The program has too many open files. Change the FILES=xx entry in the CONFIG.SYS file to a greater number, for instance 20.

10 SAMPLE OUTPUT

FIELD A2 WATER BALANCE | WEATHER STATION: 2

DEFICIT = 13 mm



PRECIPITATION:	570 mm	RUNOFF:	0 mm
EFFECTIVE PRECIP:	479 mm	INTERCEPTION:	20 mm
IRRIGATION:	0 mm	STRESS DAYS:	8
TRANSPIRATION:	413 mm	ROOT DEPTH:	1.1 m
EVAPORATION:	75 mm	NEXT IRRIG:	7 days
PERCOLATION:	91 mm	MB ERROR:	-46.2 mm

Figure 10.1 Soil water balance graph

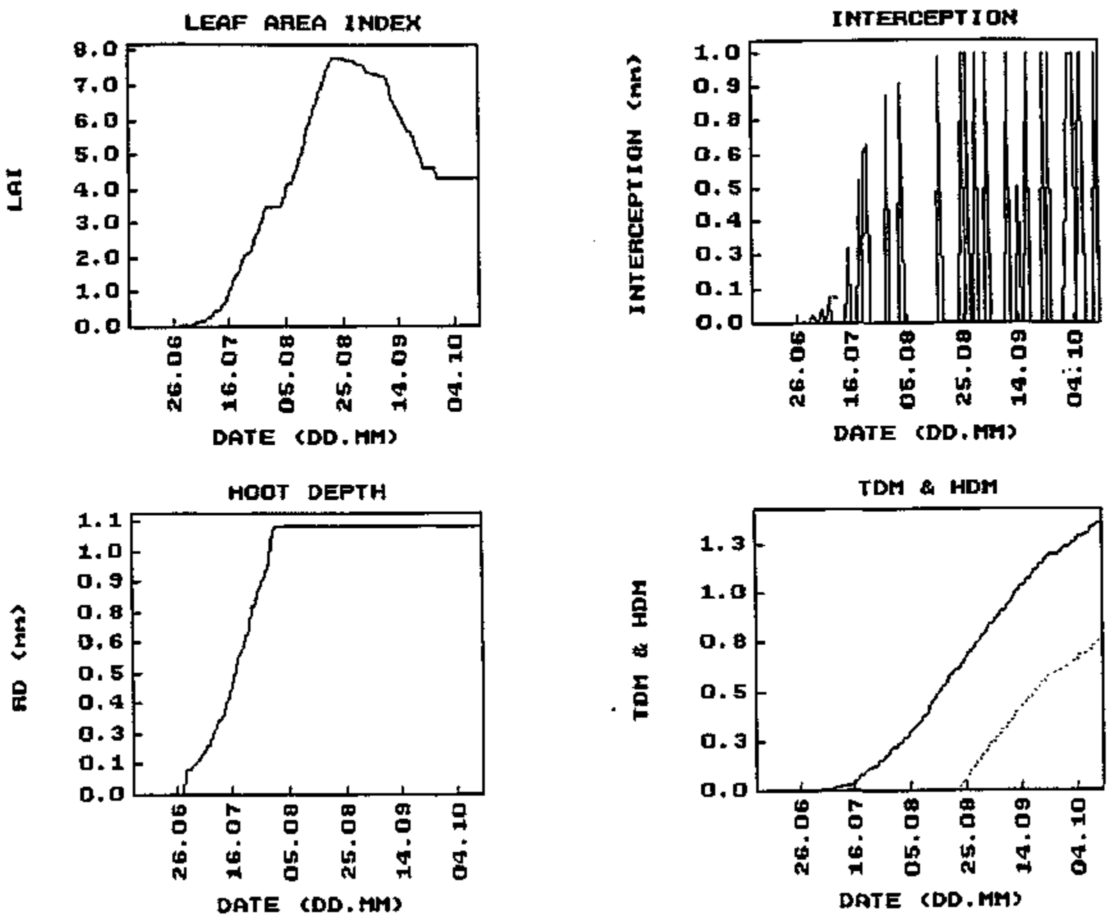


Figure 10.2 Canopy parameters graph

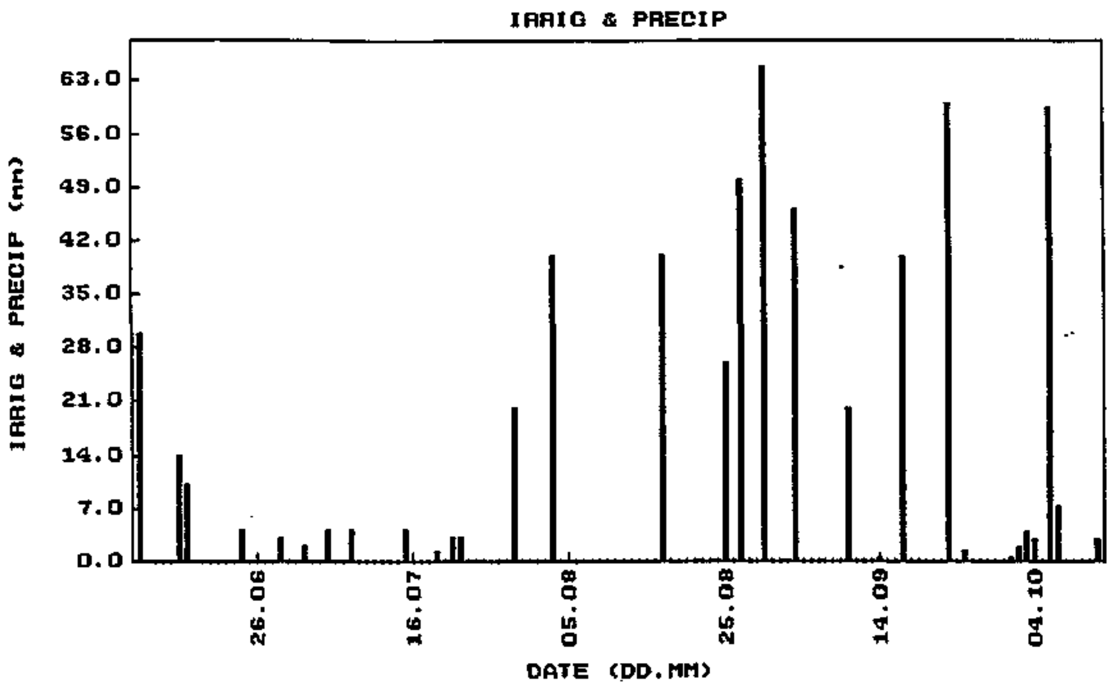


Figure 10.3 Precipitation and irrigation graph

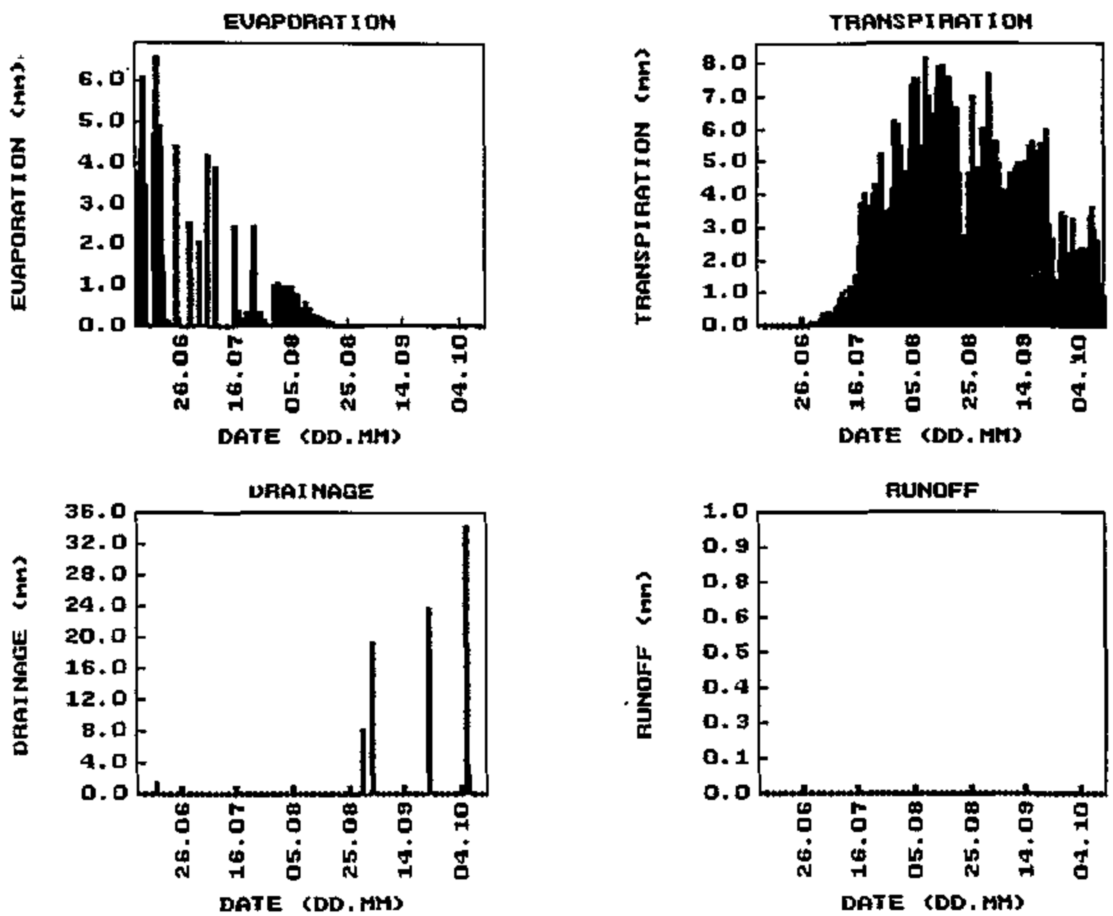


Figure 10.4 Evaporation, Transpiration, Drainage and runoff graph

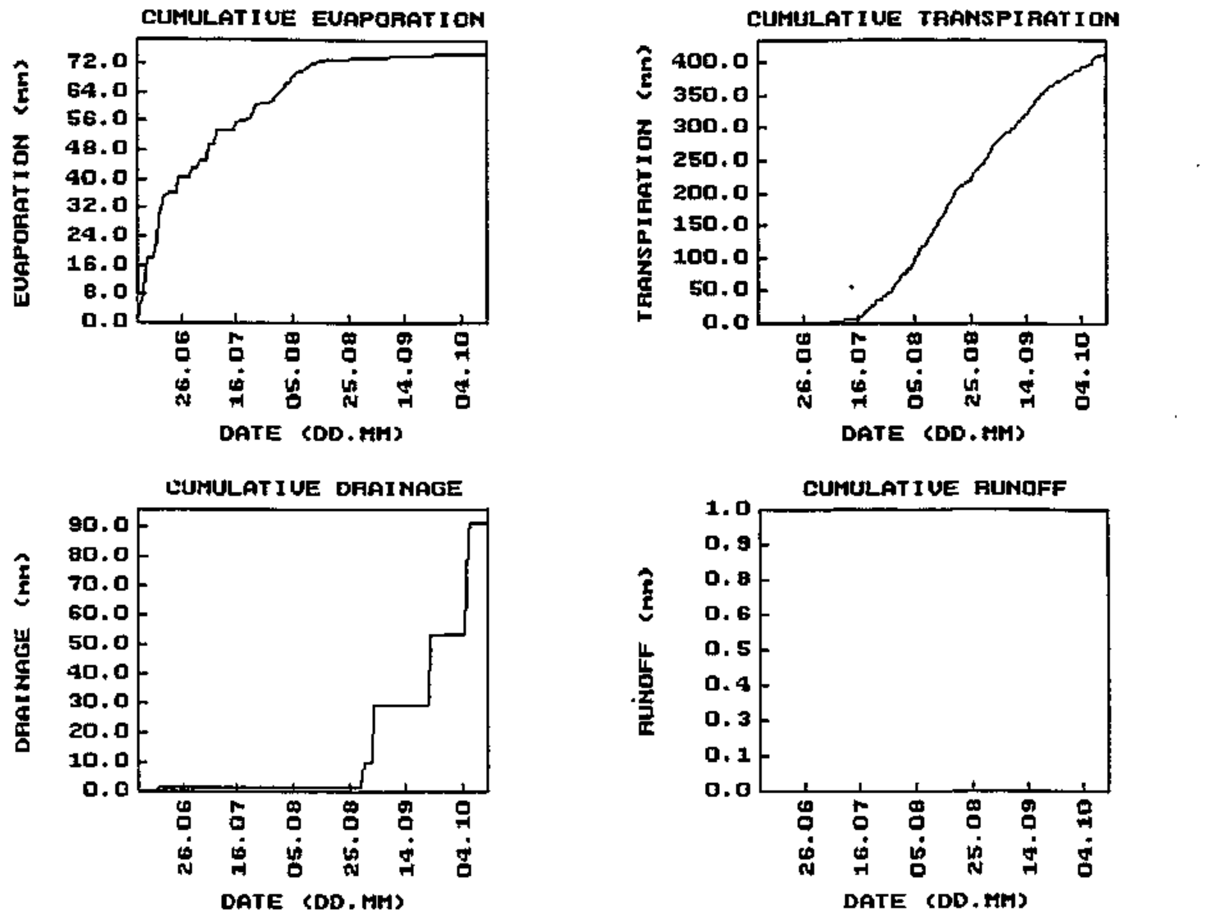


Figure 10.5 Cumulative water losses graph

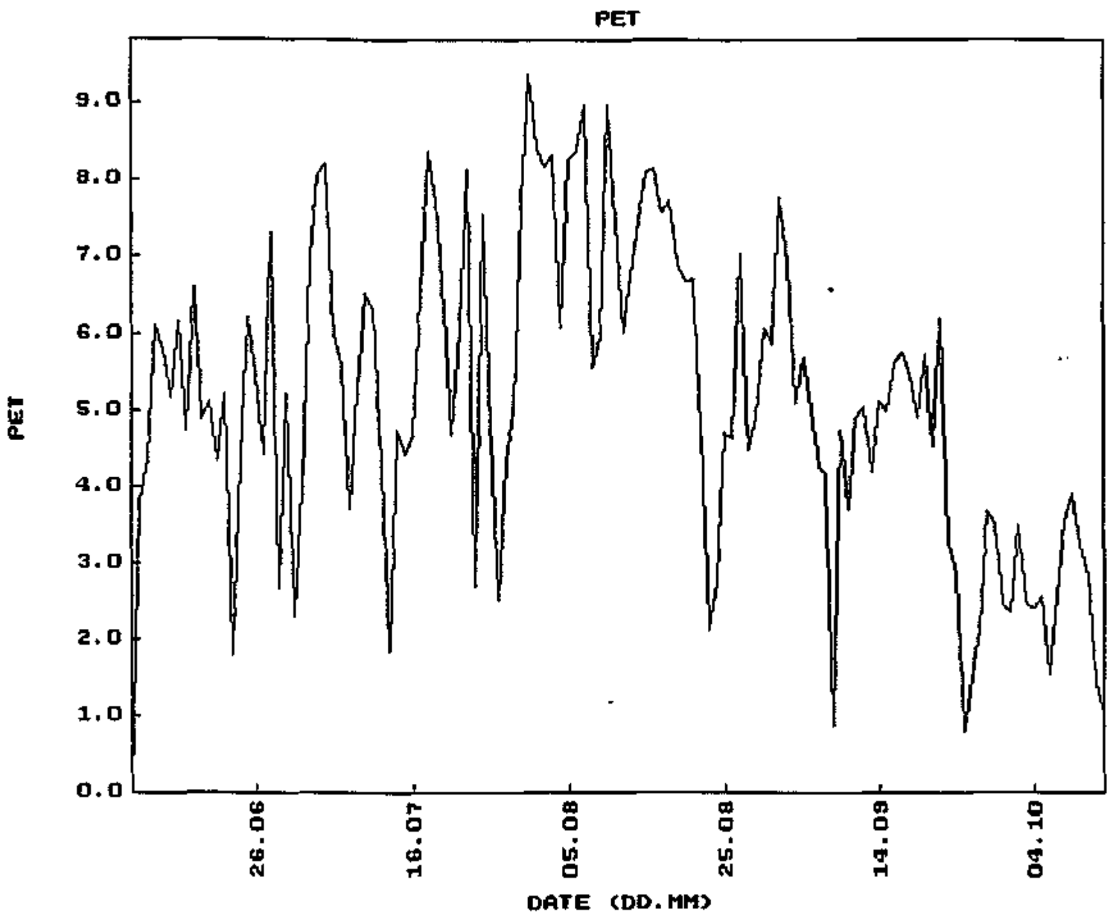


Figure 10.6 Potential evapotranspiration graph

Farm: A1
Field number: 1
Crop ID: 1
Plant date: 28.04.93
Start date: 25.04.93
PWP: -1500
FC: -10
Field size: 10.00 ha
Weather ID: 1
Schedule: Yes
Description: GEDEELTE 1

DATE	DAP	DOY	LAI	RD	TDM	HDM	DEFICIT	Precip	Irrig	Evap	Trans	Drain	Inter	RunOff	PET	SI	MBError
25.04.93	0	115	0.0	0.00	0.000	0.000	30	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0	0.0
26.04.93	0	116	0.0	0.00	0.000	0.000	35	0.0	0.0	4.2	0.0	0.0	0.0	0.0	4.2	0	0.0
27.04.93	0	117	0.0	0.00	0.000	0.000	37	0.0	0.0	1.9	0.0	0.0	0.0	0.0	1.9	0	0.0
28.04.93	0	118	0.0	0.00	0.000	0.000	38	0.0	0.0	1.2	0.0	0.0	0.0	0.0	1.2	0	-0.0
29.04.93	1	119	0.0	0.00	0.002	0.000	40	0.0	0.0	2.7	0.0	0.0	0.0	0.0	2.7	0	-0.0
30.04.93	2	120	0.0	0.00	0.002	0.000	43	0.0	0.0	2.9	0.0	0.0	0.0	0.0	2.9	0	0.0
01.05.93	3	121	0.0	0.00	0.002	0.000	44	0.0	0.0	1.1	0.0	0.0	0.0	0.0	1.4	0	0.0
02.05.93	4	122	0.0	0.00	0.002	0.000	46	0.0	0.0	1.5	0.0	0.0	0.0	0.0	1.3	0	0.0
03.05.93	5	123	0.0	0.00	0.002	0.000	46	0.0	0.0	0.4	0.0	0.0	0.0	0.0	2.4	0	0.0
04.05.93	6	124	0.0	0.00	0.002	0.000	47	0.0	0.0	0.2	0.0	0.0	0.0	0.0	2.2	0	-0.0
05.05.93	7	125	0.0	0.00	0.002	0.000	47	0.0	0.0	0.2	0.0	0.0	0.0	0.0	2.7	0	-0.0
06.05.93	8	126	0.0	0.00	0.002	0.000	39	9.7	0.0	1.5	0.0	0.0	0.0	0.0	1.5	0	0.0
07.05.93	9	127	0.0	0.00	0.002	0.000	41	0.0	0.0	2.3	0.0	0.0	0.0	0.0	2.3	0	0.0
08.05.93	10	128	0.0	0.00	0.002	0.000	35	8.4	0.0	2.6	0.0	0.0	0.0	0.0	2.6	0	0.0
09.05.93	11	129	0.0	0.15	0.002	0.000	37	0.0	0.0	2.1	0.0	0.0	0.0	0.0	2.1	1	-0.0
10.05.93	12	130	0.0	0.16	0.002	0.000	40	0.0	0.0	2.8	0.1	0.0	0.0	0.0	2.9	0	-0.0
11.05.93	13	131	0.0	0.18	0.003	0.000	45	0.0	0.0	4.5	0.1	0.0	0.0	0.0	4.6	0	0.0
12.05.93	14	132	0.1	0.20	0.003	0.000	47	0.0	0.0	2.1	0.1	0.0	0.0	0.0	5.0	0	0.0
13.05.93	15	133	0.1	0.22	0.004	0.000	47	0.0	0.0	0.4	0.1	0.0	0.0	0.0	5.2	0	0.0
14.05.93	16	134	0.1	0.23	0.004	0.000	48	0.0	0.0	0.1	0.1	0.0	0.0	0.0	3.0	0	-0.0
15.05.93	17	135	0.1	0.25	0.005	0.000	46	6.1	0.0	4.6	0.2	0.0	0.0	0.0	4.8	0	0.0
16.05.93	18	136	0.1	0.27	0.006	0.000	47	0.0	0.0	0.5	0.1	0.0	0.0	0.0	2.7	0	0.0
17.05.93	19	137	0.1	0.30	0.007	0.000	44	6.1	0.0	1.1	0.2	0.0	0.1	0.0	3.3	0	0.0
18.05.93	20	138	0.1	0.32	0.008	0.000	48	0.0	0.0	1.1	0.2	0.0	0.0	0.0	4.2	0	0.0
19.05.93	21	139	0.2	0.35	0.010	0.000	48	0.0	0.0	0.4	0.3	0.0	0.0	0.0	4.3	0	0.0
20.05.93	22	140	0.2	0.38	0.012	0.000	49	0.0	0.0	0.3	0.4	0.0	0.0	0.0	5.6	0	0.0
21.05.93	23	141	0.2	0.41	0.014	0.000	50	0.0	0.0	0.2	0.7	0.0	0.0	0.0	7.6	0	0.0
22.05.93	24	142	0.3	0.44	0.016	0.000	51	0.0	0.0	0.1	0.8	0.0	0.0	0.0	7.0	0	-0.0
23.05.93	25	143	0.3	0.48	0.019	0.000	52	0.0	0.0	0.1	1.0	0.0	0.0	0.0	8.2	0	0.0
24.05.93	26	144	0.3	0.51	0.022	0.000	53	0.0	0.0	0.1	1.1	0.0	0.0	0.0	7.9	0	-0.0
25.05.93	27	145	0.4	0.55	0.025	0.000	55	0.0	0.0	0.1	1.4	0.0	0.0	0.0	8.4	0	-0.0
26.05.93	28	146	0.4	0.59	0.028	0.000	56	0.0	0.0	0.0	1.6	0.0	0.0	0.0	8.9	0	-0.0
27.05.93	29	147	0.5	0.63	0.032	0.000	58	0.0	0.0	0.0	1.7	0.0	0.0	0.0	8.5	0	-0.0
28.05.93	30	148	0.5	0.67	0.036	0.000	60	0.0	0.0	0.0	2.3	0.0	0.0	0.0	10.1	0	-0.0
29.05.93	31	149	0.6	0.72	0.043	0.000	62	0.0	0.0	0.0	2.1	0.0	0.0	0.0	8.3	0	-0.0
30.05.93	32	150	0.7	0.78	0.050	0.000	65	0.0	0.0	0.0	2.4	0.0	0.0	0.0	8.4	0	-0.0
31.05.93	33	151	0.8	0.83	0.056	0.000	68	0.0	0.0	0.0	2.9	0.0	0.0	0.0	9.3	0	-0.0
01.06.93	34	152	0.9	0.87	0.062	0.000	70	0.0	0.0	0.0	2.1	0.0	0.0	0.0	5.9	0	-0.0
02.06.93	35	153	0.9	0.89	0.065	0.000	67	4.8	0.0	1.1	0.9	0.0	0.4	0.0	2.4	0	0.0
03.06.93	36	154	1.0	0.95	0.074	0.000	71	0.0	0.0	1.3	3.0	0.0	0.0	0.0	5.1	0	-0.0
04.06.93	37	155	1.1	1.01	0.083	0.000	74	0.0	0.0	0.7	2.7	0.0	0.0	0.0	6.6	0	-0.0
05.06.93	38	156	1.2	1.06	0.091	0.000	77	0.0	0.0	0.3	2.8	0.0	0.0	0.0	6.3	0	-0.0
06.06.93	39	157	1.3	1.12	0.102	0.000	81	0.0	0.0	0.2	3.7	0.0	0.0	0.0	7.7	0	-0.0
07.06.93	40	158	1.3	1.18	0.113	0.000	85	0.0	0.0	0.1	3.6	0.0	0.0	0.0	7.3	0	0.0
08.06.93	41	159	1.4	1.23	0.123	0.000	89	0.0	0.0	0.1	4.2	0.0	0.0	0.0	7.9	0	0.0
09.06.93	42	160	1.5	1.27	0.132	0.000	93	0.0	0.0	0.1	3.7	0.0	0.0	0.0	6.7	0	0.0
10.06.93	43	161	1.5	1.29	0.135	0.000	93	1.3	0.0	0.0	0.9	0.0	0.6	0.0	1.5	0	-0.0
11.06.93	44	162	1.6	1.35	0.148	0.000	92	4.8	0.0	1.4	2.1	0.0	0.6	0.0	3.5	0	-0.0
12.06.93	45	163	1.7	1.41	0.162	0.000	96	0.0	0.0	0.9	2.3	0.0	0.0	0.0	3.9	0	0.0
13.06.93	46	164	1.8	1.48	0.179	0.000	100	1.5	0.0	1.3	3.7	0.0	0.6	0.0	6.0	0	-0.0
14.06.93	47	165	1.9	1.53	0.191	0.000	104	0.0	0.0	0.5	3.4	0.0	0.0	0.0	5.4	0	0.0
15.06.93	48	166	1.9	1.56	0.199	0.000	105	1.1	0.0	1.2	2.9	0.0	0.7	0.0	4.4	0	-0.0
16.06.93	49	167	2.0	1.63	0.217	0.000	110	0.0	0.0	0.9	4.1	0.0	0.0	0.0	6.0	0	0.0
17.06.93	50	168	2.1	1.68	0.230	0.000	114	0.0	0.0	0.4	2.9	0.0	0.0	0.0	4.2	0	-0.0
18.06.93	51	169	2.1	1.73	0.243	0.000	118	0.0	0.0	0.4	4.5	0.0	0.0	0.0	6.4	0	0.0
19.06.93	52	170	2.2	1.77	0.257	0.000	122	0.0	0.0	0.2	3.3	0.0	0.0	0.0	4.6	0	-0.0
20.06.93	53	171	2.3	1.83	0.274	0.000	126	0.0	0.0	0.2	3.5	0.0	0.0	0.0	4.9	0	0.0
21.06.93	54	172	2.3	1.88	0.289	0.000	129	0.0	0.0	0.1	2.9	0.0	0.0	0.0	3.9	0	0.0
22.06.93	55	173	2.4	1.94	0.306	0.000	132	0.0	0.0	0.1	3.6	0.0	0.0	0.0	4.9	0	0.0
23.06.93	56	174	2.4	1.95	0.309	0.000	121	13.0	0.0	0.3	0.8	0.0	0.8	0.0	1.0	0	-0.0
24.06.93	57	175	2.5	2.02	0.332	0.000	117	8.9	0.0	1.0	3.1	0.0	0.8	0.0	4.1	0	-0.0
25.06.93	58	176	2.6	2.02	0.359	0.000	123	0.0	0.0	1.5	4.7	0.0	0.0	0.0	6.2	0	-0.0
26.06.93	59	177	2.6	2.02	0.380	0.000	128	0.0	0.0	1.1	3.9	0.0	0.0	0.0	5.1	0	-0.0
27.06.93	60	178	2.7	2.02	0.395	0.000	132	0.0	0.0	0.8	3.0	0.0	0.0	0.0	3.8	0	-0.0
28.06.93	61	179	2.7	2.02	0.418	0.000	134	6.1	0.0	1.5	5.8	0.0	0.8	0.0	7.2	0	-0.0
29.06.93	62	180	2.7	2.02	0.424	0.000	134	2.5	0.0	0.4	1.4	0.0	0.8	0.0	1.7	0	-0.0
30.06.93	63	181	2.8	2.02	0.454	0.000	133	6.9	0.0	1.0	4.0	0.0	0.8	0.0	5.0	0	-0.0
01.07.93	64	182	2.8	2.02	0.462	0.000	114	0.5	0.0	0.3	1.3	0.0	0.5	0.0	1.6	0	-0.0
02.07.93	65	183	2.9	2.02	0.484	0.000	130	8.6	0.0	0.7	2.9	0.0	0.8	0.0	3.5	0	-0.0
03.07.93	66	184	2.9	2.02	0.510	0.000	137	1.5	0.0	1.3	5.7	0.0	0.8	0.0	7.0	0	-0.0
04.07.93	67	185	2.9	2.02	0.535	0.000	145	0.0	0.0	1.4	6.6	0.0	0.0	0.0	8.1	0	-0.0
05.07.93	68	186	2.9	2.02	0.563	0.000	153	0.0	0.0	1.4	6.8	0.0	0.0	0.0	8.2	0	-0.0
06.07.93	69	187	3.0	2.02	0.581	0.000	158	0.0	0.0	0.9	4.5	0.0	0.0	0.0	5.5	0	-0.0
07.07.93	70	188	2.9	2.02	0.605	0.000	163	0.0	0.0	0.9	4.3	0.0	0.0	0.0	5.2	0	-0.0
08.07.93	71	189	2.9	2.02	0.616	0.223	164	2.8	0.0	0.5	2.5	0.0	0.8	0.0	3.0	0	-0.0
09.07.93	72	190	2.9	2.02	0.650	0.256	170	0.0	0.0	0.8	4.3	0.0	0.0	0.0	5.1	0	-0.0
10.07.93	73	191	2.9	2.02	0.679	0.286	176	0.0	0.0	1.0	5.4	0.0	0.0	0.0	6.5	0	-0.0
11.07.93	74	192	2.8	2.02	0.705	0.311	182	0.0	0.0	1.0	5.0	0.0	0.0	0.0	6.1	0	-0.0
12.07.93	75	193	2.8	2.02	0.713	0.320	185	0.0	0.0	0.5	2.8	0.0	0.0	0.0	3.4	0	-0.0
13.07.93	76	194	2.7	2.02	0.719	0.325	186	0.0	0.0	0.2	0.9	0.0	0.0	0.0	1.1	0	-0.0
14.07.93	77	195	2.7	2.02	0.743	0.350	187	4.8	0.0	0.7	3.6	0.0	0.8	0.0	4.4	0	-0.0
15.07.93	78	196	2.7	2.02	0.765	0.371	190	0.0	0.0	0.6	3.3	0.0	0.0	0.0	4.1	0	-0.0
16.07.93	79	197	2.6	2.02	0.794	0.401	195	0.0	0.0	0.7	3.5	0.0	0.0	0.0	4.4	0	-0.0
17.07.93	80	198	2.6	2.02	0.828	0.434	201	0.0	0.0	1.0	5.2	0.0	0.0	0.0	6.5	0	-0.0
18.07.93	81	199	2.5	2.02	0.854	0.460	209	0.5	0.0	1.3	6.7	0.0	0.5	0.0	8.4	0	-0.0
19.07.93	82	200	2.5	2.02	0.870	0.477	216	0.0	0.0	1.2	5.8	0.0	0.0	0.0	7.4	0	-0.0
20.07.93	83	201	2.4	2.02	0.884	0.491	221	0.8	0.0	0.9	4.5	0.0	0.8	0.0	5.8	0	-0.0
21.07.93	84	202	2.3	2.02	0.898	0.505	225	0.0	0.0	0.7	3.2	0.0	0.0	0.0	4.2	0	-0.0
22.07.93	85	203	2.3	2.02	0.914	0.520	230	0.0	0.0	0.9	4.3	0.0	0.0	0.0	5.6	0	-0.0
23.07.93	86	204	2.2	2.02	0.933	0.540	233	5.1	0.0	1.2	5.9	0.0	0.8	0.0	7.9	0	-0.0
24.07.93	87	205	2.1	2.02	0.938	0.545	234	1.5	0.0	0.3	1.3	0.0	0.8	0.0	1.8	0	-0.0
25.07.93	88	206	1.9	2.02	0.961	0.568	241	0.0	0.0	1.2	5.5	0.0	0.0	0.0	7.5	0	-0.0
26.07.93	89	207	1.8	2.02	0.978	0.585	238	7.0	0.0	0.7	3.1	0.0	0.8	0.0	4.3	0	-0.0
27.07.93	90	208	1.8	2.02	0.984	0.591	240	0.0	0.0	0.3	1.2	0.0	0.0	0.0	1.8	0	-0.0
28.07.93	91	209	1.7	2.02	1.004	0.610	243	0.0	0.0	0.6	2.6	0.0	0.0	0.0	3.9	0	-0.0
29.07.93	92	210	1.7	2.02	1.017	0.624	247	0.0	0.0	0.7	3.0	0.0	0.0	0.0	4.5	0	-0.0
30.07.93	93	211	1.6	2.02	1.031	0.638	253	0.0	0.0	1.2	4.8	0.0	0.0	0.0	7.5	0	-0.0

DATE	DAP	DOY	LAI	RD	TDM	HDM	DEFICIT	Precip	Irrig	Evap	Trans	Drain	Inter	RunOff	PET	SI	MBError
31.07.93	94	212	1.5	2.02	1.046	0.653	260	0.0	0.0	1.4	5.8	0.0	0.0	0.0	9.2	0	-0.0
01.08.93	95	213	1.3	2.02	1.061	0.668	266	0.0	0.0	1.3	5.0	0.0	0.0	0.0	8.2	0	-0.0
02.08.93	96	214	1.2	2.02	1.072	0.679	272	0.0	0.0	1.2	4.6	0.0	0.0	0.0	7.8	0	-0.1
03.08.93	97	215	1.2	2.02	1.085	0.692	278	0.0	0.0	1.3	4.3	0.0	0.0	0.0	8.1	0	-0.1
04.08.93	98	216	1.0	2.02	1.093	0.699	281	0.0	0.0	0.8	2.8	0.0	0.0	0.0	5.4	0	-0.1
05.08.93	99	217	0.9	2.02	1.102	0.708	287	0.0	0.0	1.2	3.9	0.0	0.0	0.0	7.9	0	-0.1
06.08.93	100	218	0.8	2.02	1.109	0.715	291	0.0	0.0	1.2	3.5	0.0	0.0	0.0	7.7	0	-0.1
07.08.93	101	219	0.7	2.02	..124	0.721	296	0.0	0.0	1.1	3.0	0.0	0.0	0.0	8.4	1	-0.1
08.08.93	102	220	0.5	2.02	1.128	0.725	298	0.0	0.0	0.7	1.7	0.0	0.0	0.0	4.6	0	-0.1
09.08.93	103	221	0.4	2.02	1.123	0.730	301	0.0	0.0	0.8	1.7	0.0	0.0	0.0	5.2	0	-0.1
10.08.93	104	222	0.3	2.02	1.128	0.735	304	0.0	0.0	1.3	1.8	0.0	0.0	0.0	8.7	1	-0.1
11.08.93	105	223	0.3	2.02	1.132	0.738	305	1.8	0.0	0.9	1.5	0.0	0.8	0.0	6.8	0	-0.1
12.08.93	106	224	0.3	2.02	1.137	0.743	307	0.0	0.0	0.5	1.1	0.0	0.0	0.0	5.8	0	-0.1
13.08.93	107	225	0.2	2.02	1.141	0.747	308	0.0	0.0	0.4	1.0	0.0	0.0	0.0	6.5	0	-0.1
14.08.93	108	226	0.1	2.02	1.143	0.750	309	0.0	0.0	0.4	0.9	0.0	0.0	0.0	7.1	1	-0.1
15.08.93	109	227	0.1	2.02	1.145	0.752	310	0.0	0.0	0.3	0.7	0.0	0.0	0.0	7.8	1	-0.1
16.08.93	110	228	0.0	2.02	1.146	0.753	311	0.0	0.0	0.2	0.4	0.0	0.0	0.0	7.8	1	-0.1
17.08.93	111	229	0.0	2.02	1.146	0.753	311	0.0	0.0	0.2	0.2	0.0	0.0	0.0	7.2	1	-0.1
18.08.93	112	230	0.0	2.02	1.146	0.753	312	0.0	0.0	0.2	0.0	0.0	0.0	0.0	7.3	1	-0.1
19.08.93	113	231	0.0	2.02	1.146	0.753	312	0.0	0.0	0.1	0.0	0.0	0.0	0.0	6.4	1	-0.1
20.08.93	114	232	0.0	2.02	1.146	0.753	312	0.0	0.0	0.1	0.0	0.0	0.0	0.0	6.0	0	-0.1

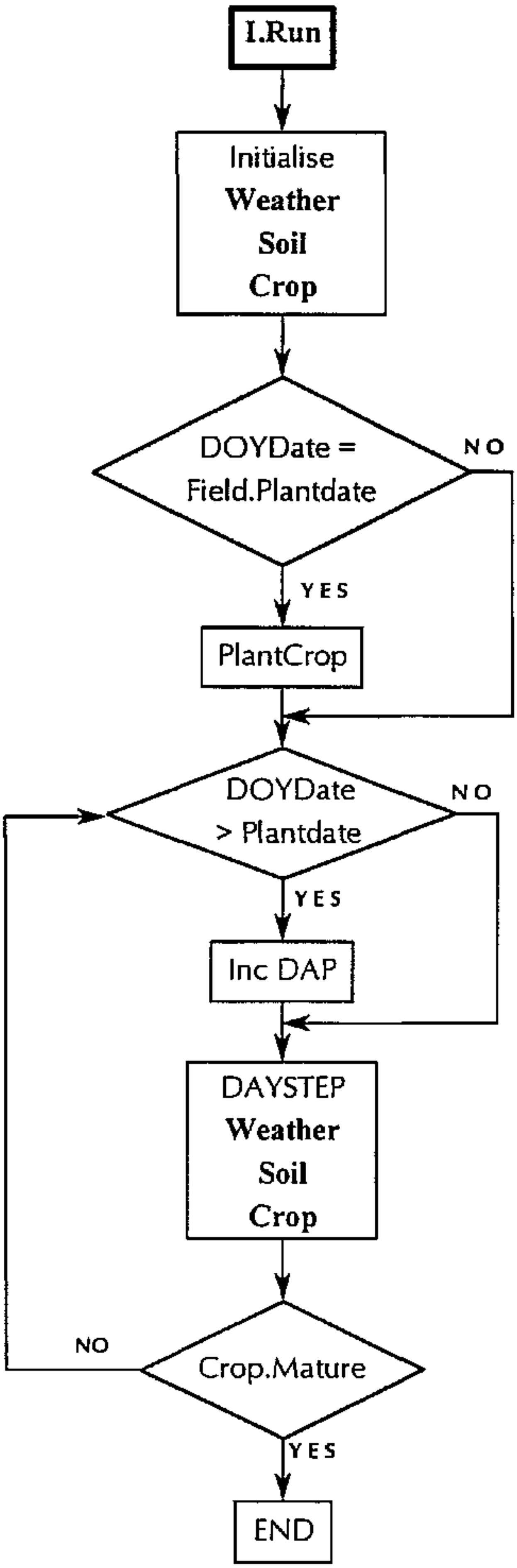
Cumulative precipitation: 118 mm
Cumulative irrigation: 0 mm
Cumulative evaporation: 108 mm
Cumulative transpiration: 277 mm
Cumulative drainage: 0 mm
Cumulative interception: 15 mm
Cumulative runoff: 0 mm
Cumulative ETD: 384 mm
Number of stress days: 9
Effective rain: 118 mm

APPENDIX B PROGRAM FLOW DIAGRAMS

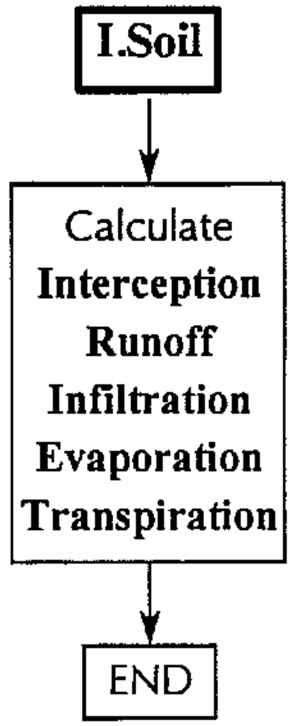
LIST OF SYMBOLS

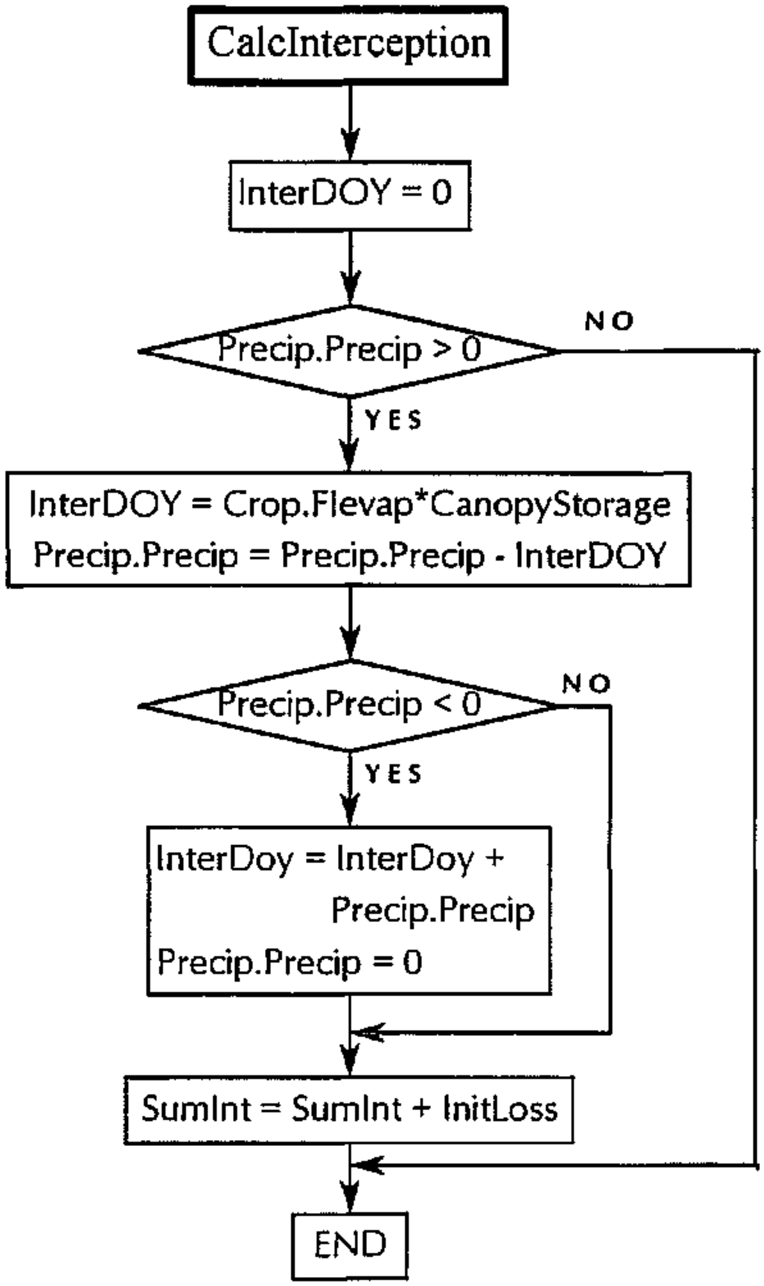
Adwc	: air dry water content
bd	: bulk density
CDM	: canopy dry matter
CDMmax	: canopy dry matter at maximum LAI
CDMstart	: canopy dry matter at emergence
coneff	: radiation conversion efficiency
cumETD	: cumulative evaporation, transpiration and drainage
DAP	: day(s) after planting
DailyLAI	: daily leaf area index
DOY	: day of year
dwr	: dry matter water ratio
dz	: soil layers depth
Eac	: atmospheric emmissivity
Ea	: clear sky emmissivity
Emax	: maximum dimensionless transpiration rate
emdd	: emergence day degrees
ETP	: potential evapotranspiration
fcwc	: field capacity water content
fdd	: flowering day degrees
FI	: fractional interception
FIevap	: fractional interception (solar radiation) evaporation
fldd	: day degrees at end of vegetative growth
froot	: fraction of dry matter partitioned to roots
GDD	: growing degree days
gpdd	: transition period (ddeg) from vegetative to reproductive growth
HDM	: harvestable dry matter
Irrig	: irrigation
kc	: average daily transmission coefficient
LAI	: leaf area index
Lni	: net isothermal long wave radiation
MaxLAI	: maximum leaf area index

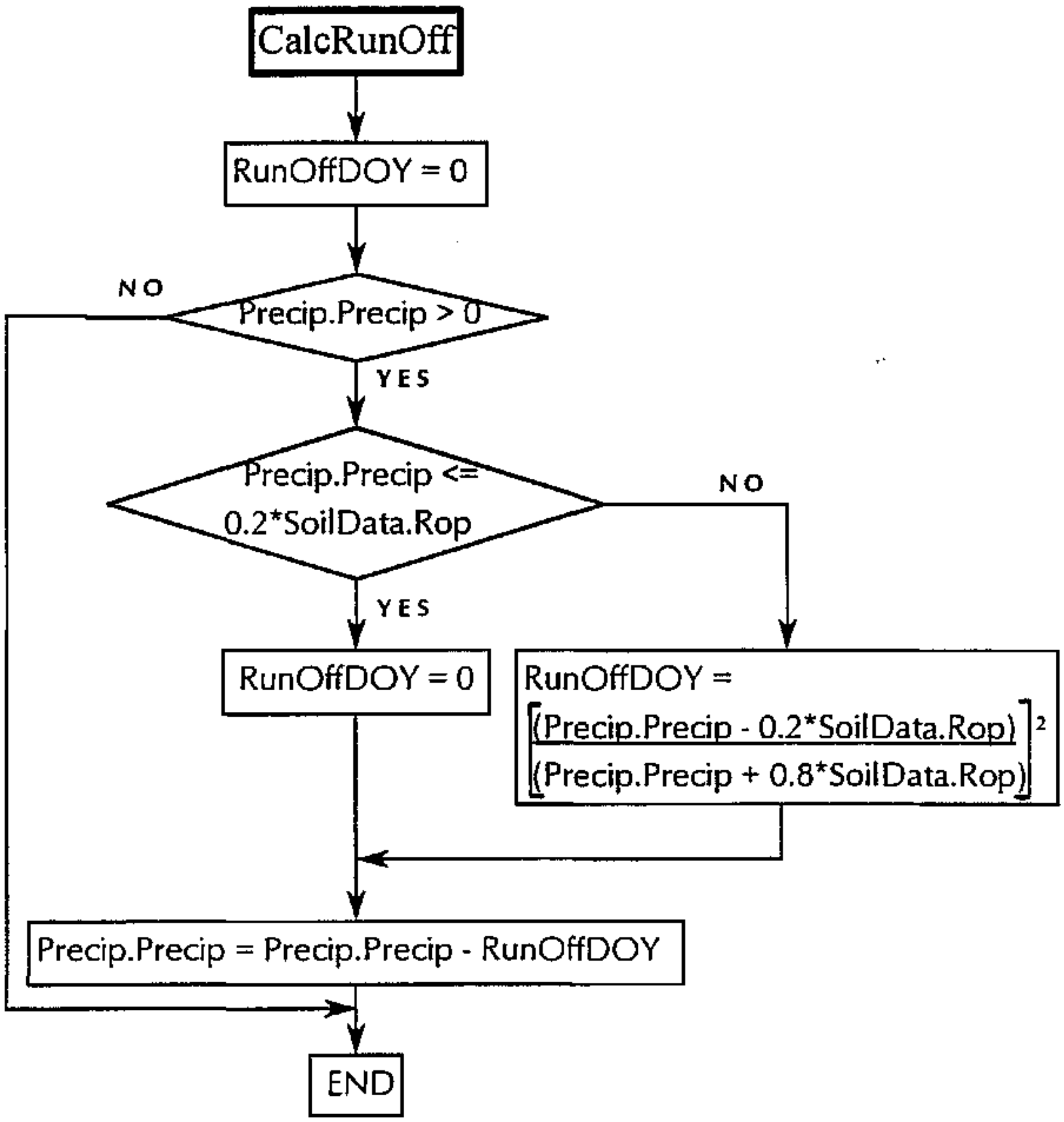
MaxLeafAge	: day degrees for leaf senescence
mtdd	: maturity day degrees
Part	: leaf-stem partition parameter
pgf	: potential grain production, as fraction of SDM at anthesis
Precip	: precipitation
Psilm	: leaf water potential at maximum transpiration
PSR	: potential solar radiation
pwpwc	: permanent wilting point water content
Rd	: root depth
RDmax	: maximum rooting depth
rgr	: root growth rate
RH	: relative humidity
Rni	: net isothermal radiation
Rop	: runoff parameter
SLA	: specific leaf area
Solar	: solar radiation
SVP	: saturation vapour pressure
Ta	: air temperature
tdm	: top dry matter
tdml	: top dry matter leaves
tdmr	: top dry matter roots
tdms	: top dry matter stems
Tbase	: base temperature for GDD
Tcutoff	: cutoff temperature for GDD
TDMstart	: canopy dry matter at emergence
Tlo	: temperature for optimum light-limited growth
Tr	: atmospheric transmissivity
Transl	: fraction of dry matter partitioned to roots
u	: wind speed
VP	: vapour pressure
VPD	: vapour pressure deficit
wc	: water content

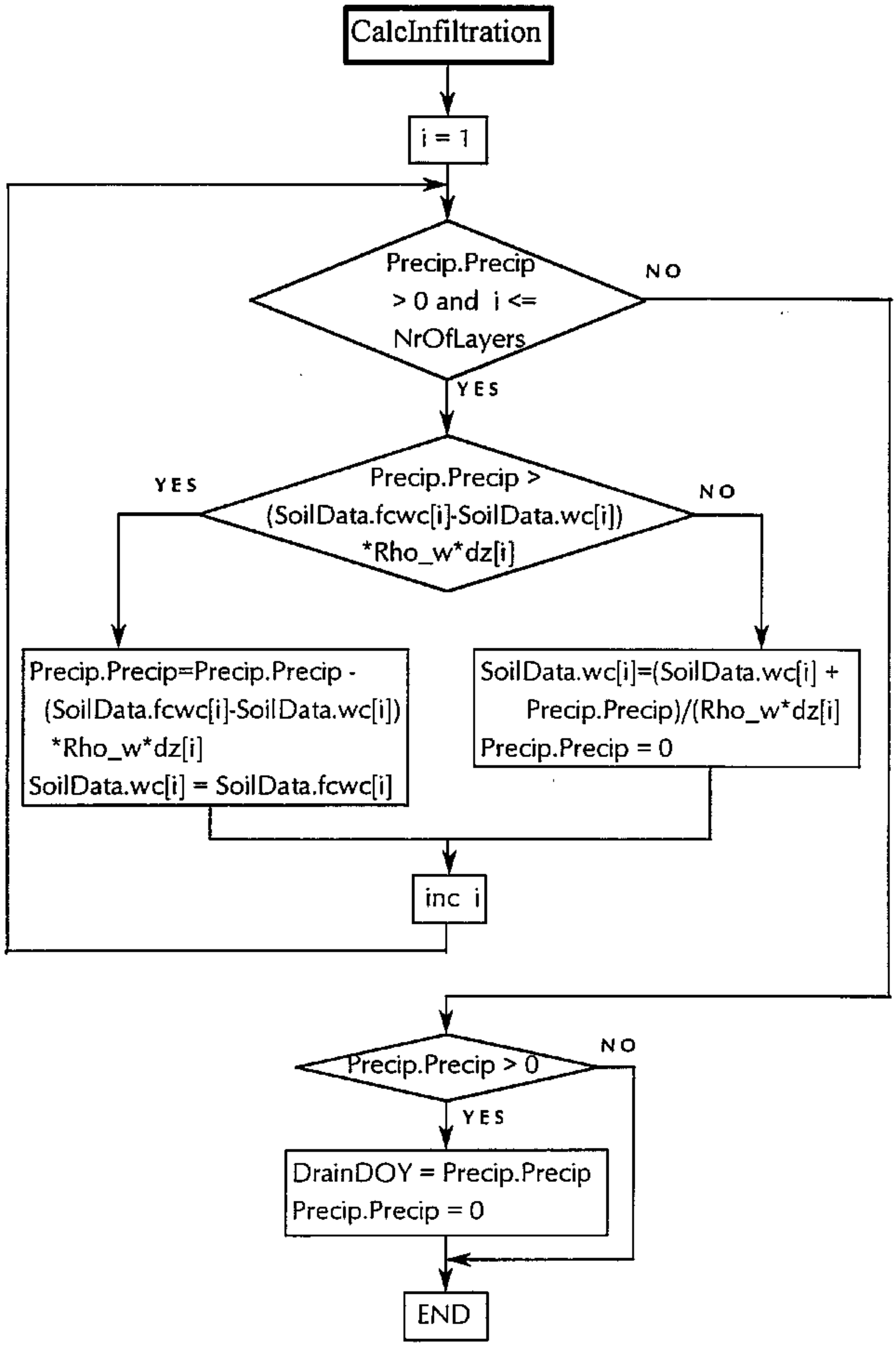

I.Run
Initialise
Weather
Soil
Crop
DOYDate = Field.Plantdate
NO
YES
PlantCrop
DOYDate > Plantdate
NO
YES
Inc DAP
DAYSTEP
Weather
Soil
Crop
NO
Crop.Mature
YES
END

1b

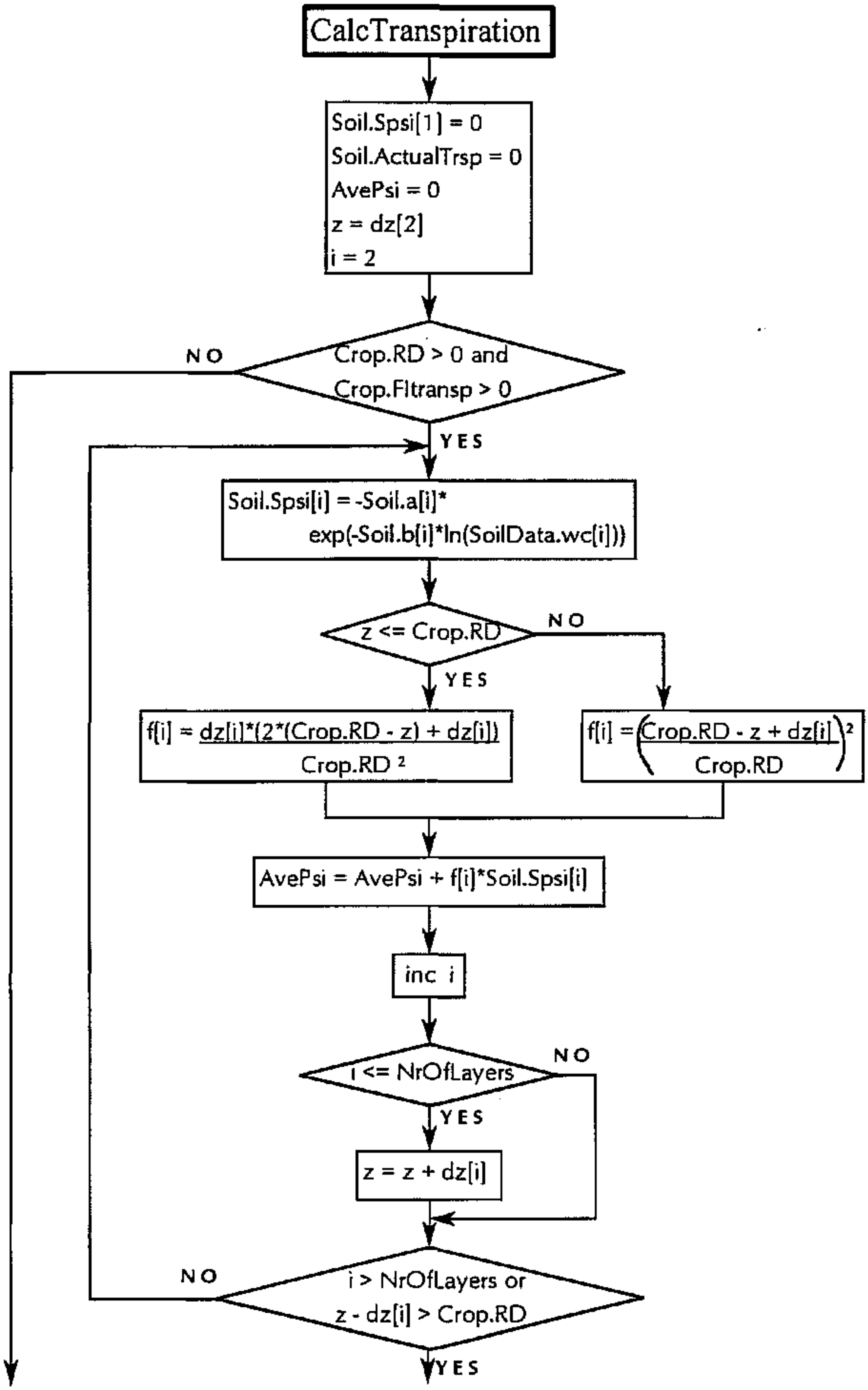

I.Soil
Calculate
Interception
Runoff
Infiltration
Evaporation
Transpiration
END


CalcInterception
InterDOY = 0
Precip.Precip > 0
NO
YES
InterDOY = Crop.Flevap*CanopyStorage
Precip.Precip = Precip.Precip - InterDOY
Precip.Precip < 0
NO
YES
InterDoy = InterDoy + Precip.Precip
Precip.Precip = 0
SumInt = SumInt + InitLoss
END

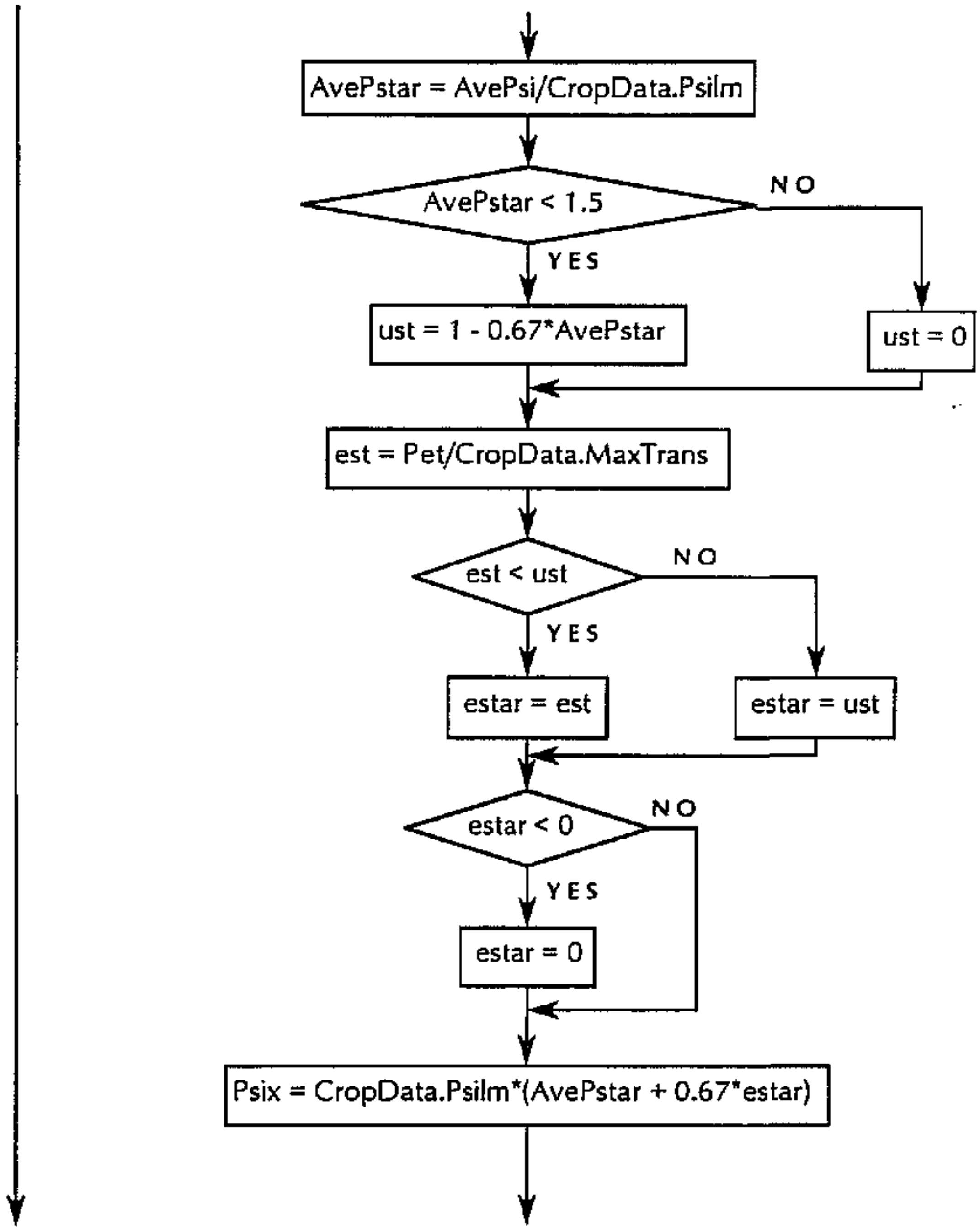

CalcRunOff
RunOffDOY = 0
NO
Precip.Precip > 0
YES
Precip.Precip <=
0.2*SoilData.Rop
NO
YES
RunOffDOY = 0
RunOffDOY =
[(Precip.Precip - 0.2*SoilData.Rop)
(Precip.Precip + 0.8*SoilData.Rop)]2
Precip.Precip = Precip.Precip - RunOffDOY
END


CalcInfiltration
i = 1
Precip.Precip > 0 and i <= NrOfLayers
NO
YES
Precip.Precip > (SoilData.fcwc[i]-SoilData.wc[i]) *Rho_w*dz[i]
YES
NO
Precip.Precip=Precip.Precip - (SoilData.fcwc[i]-SoilData.wc[i]) *Rho_w*dz[i]
SoilData.wc[i] = SoilData.fcwc[i]
SoilData.wc[i]=(SoilData.wc[i] + Precip.Precip)/(Rho_w*dz[i]
Precip.Precip = 0
inc i
Precip.Precip > 0
NO
YES
DrainDOY = Precip.Precip
Precip.Precip = 0
END

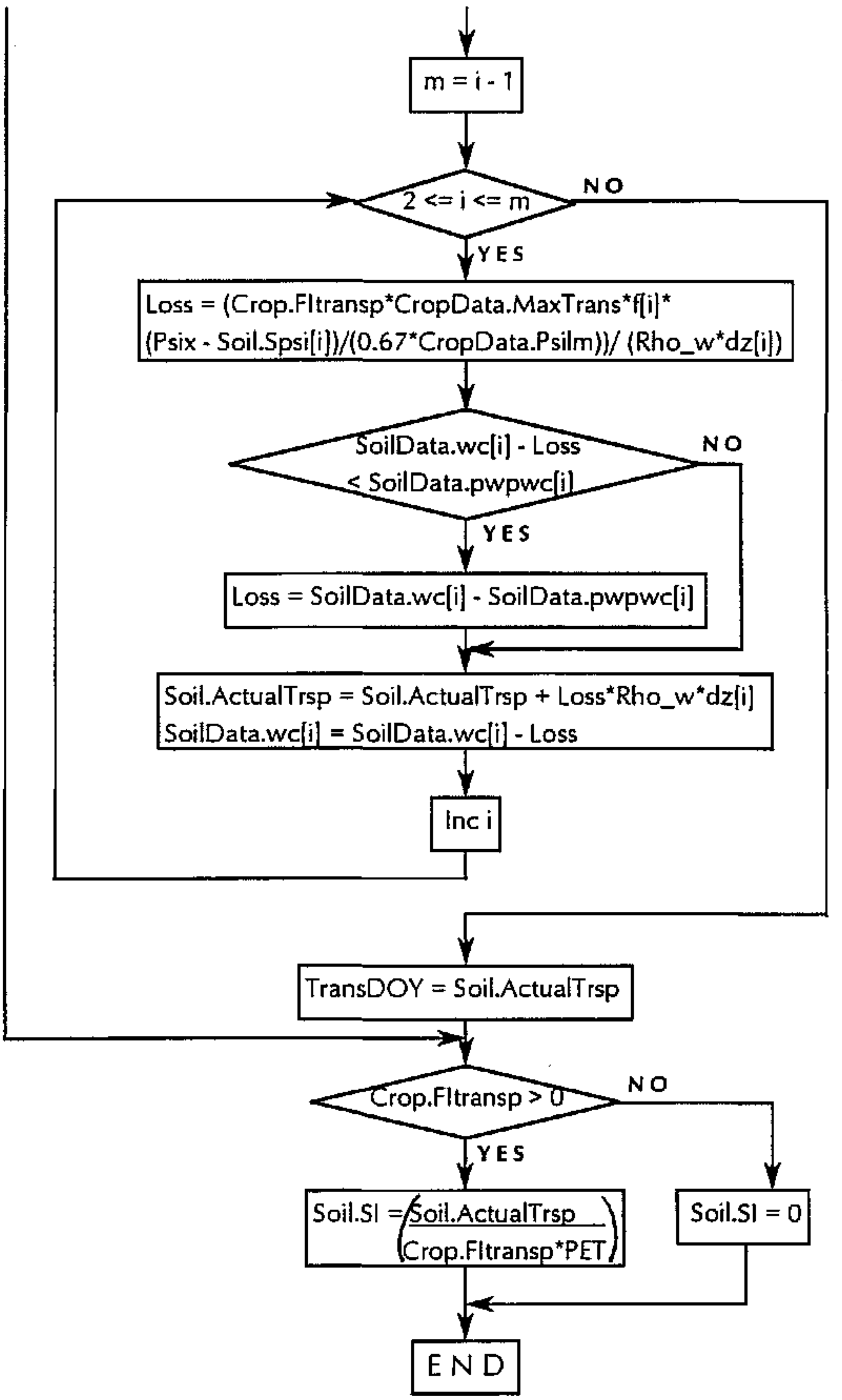
5b

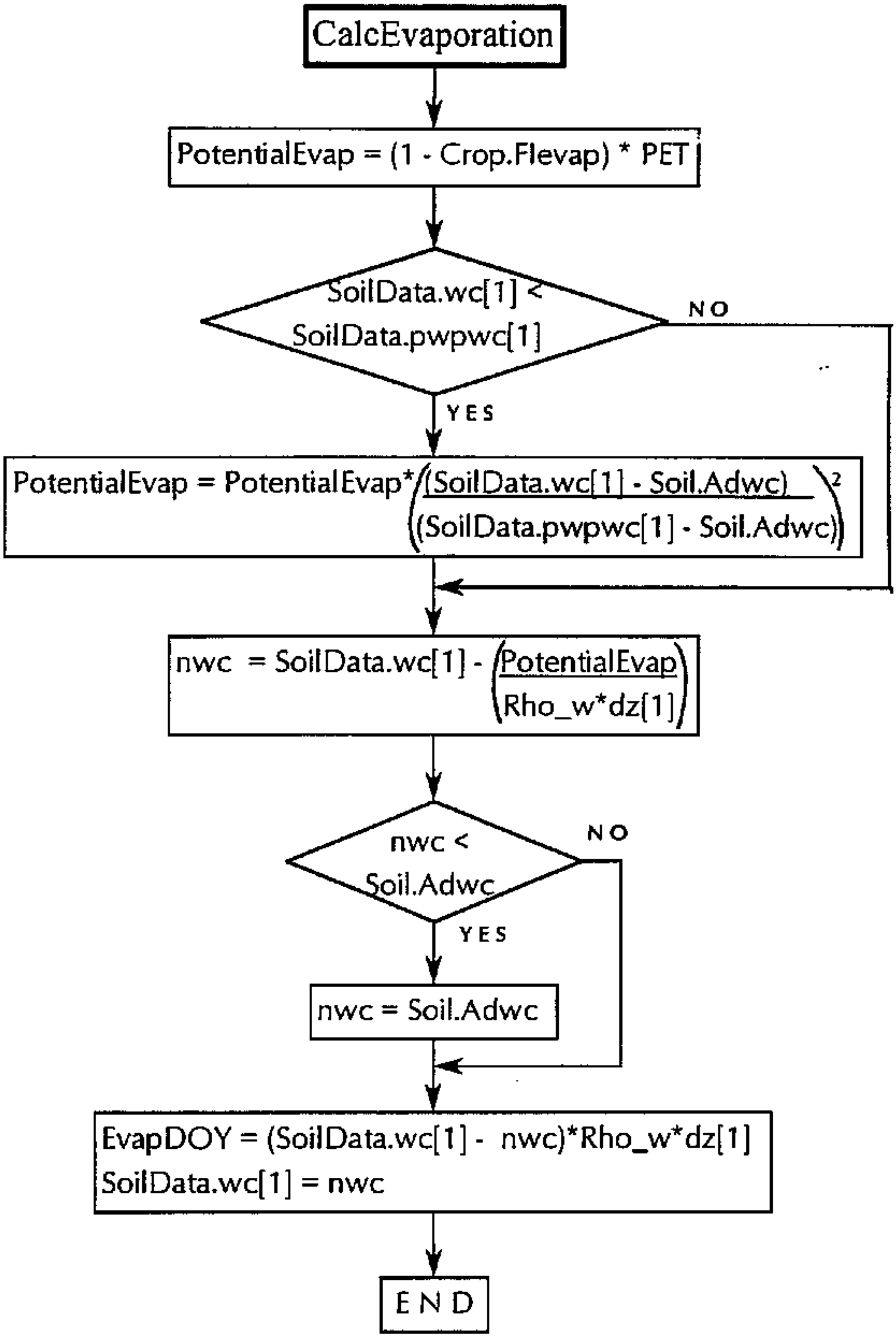


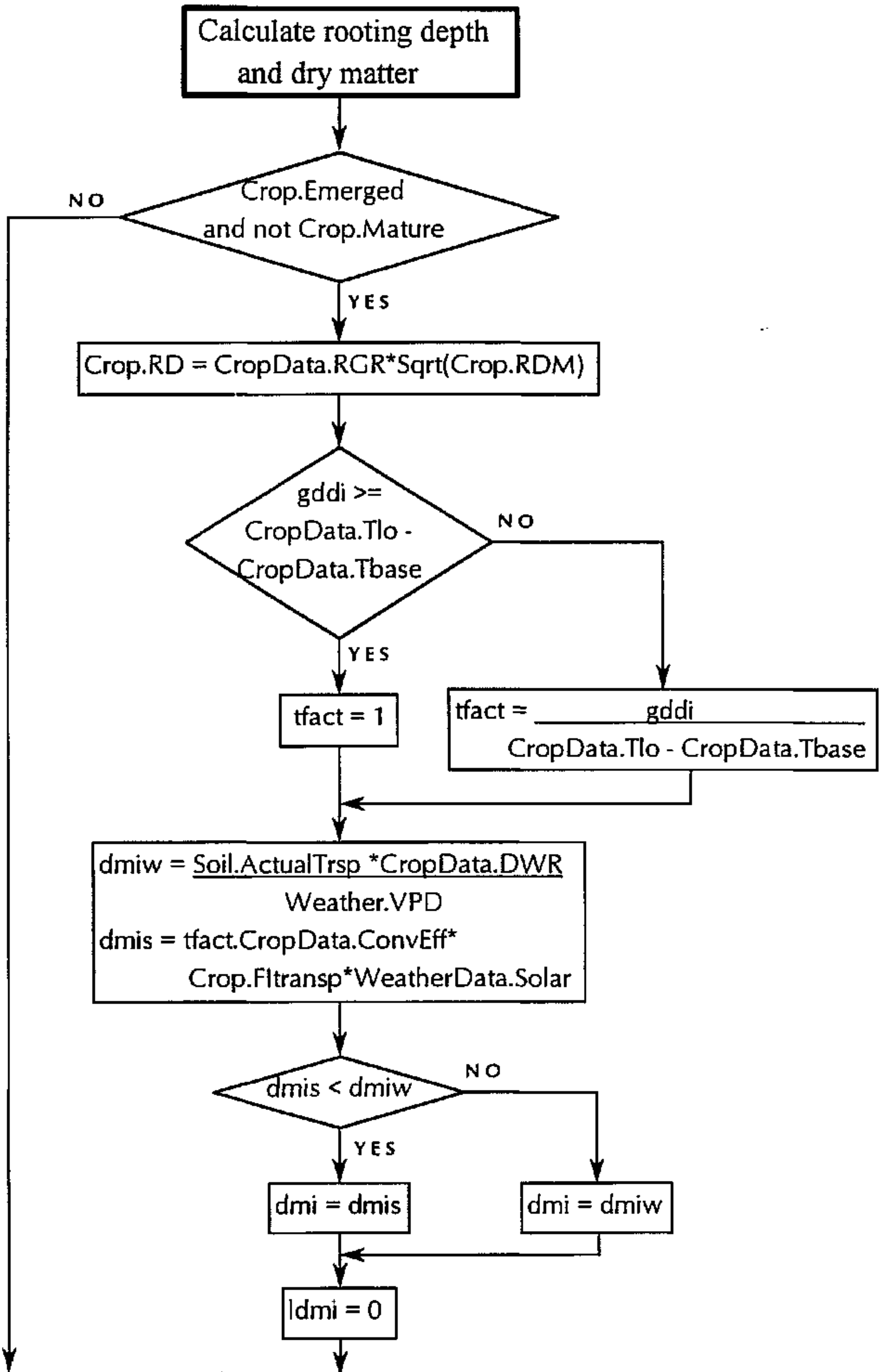
Continue on next page



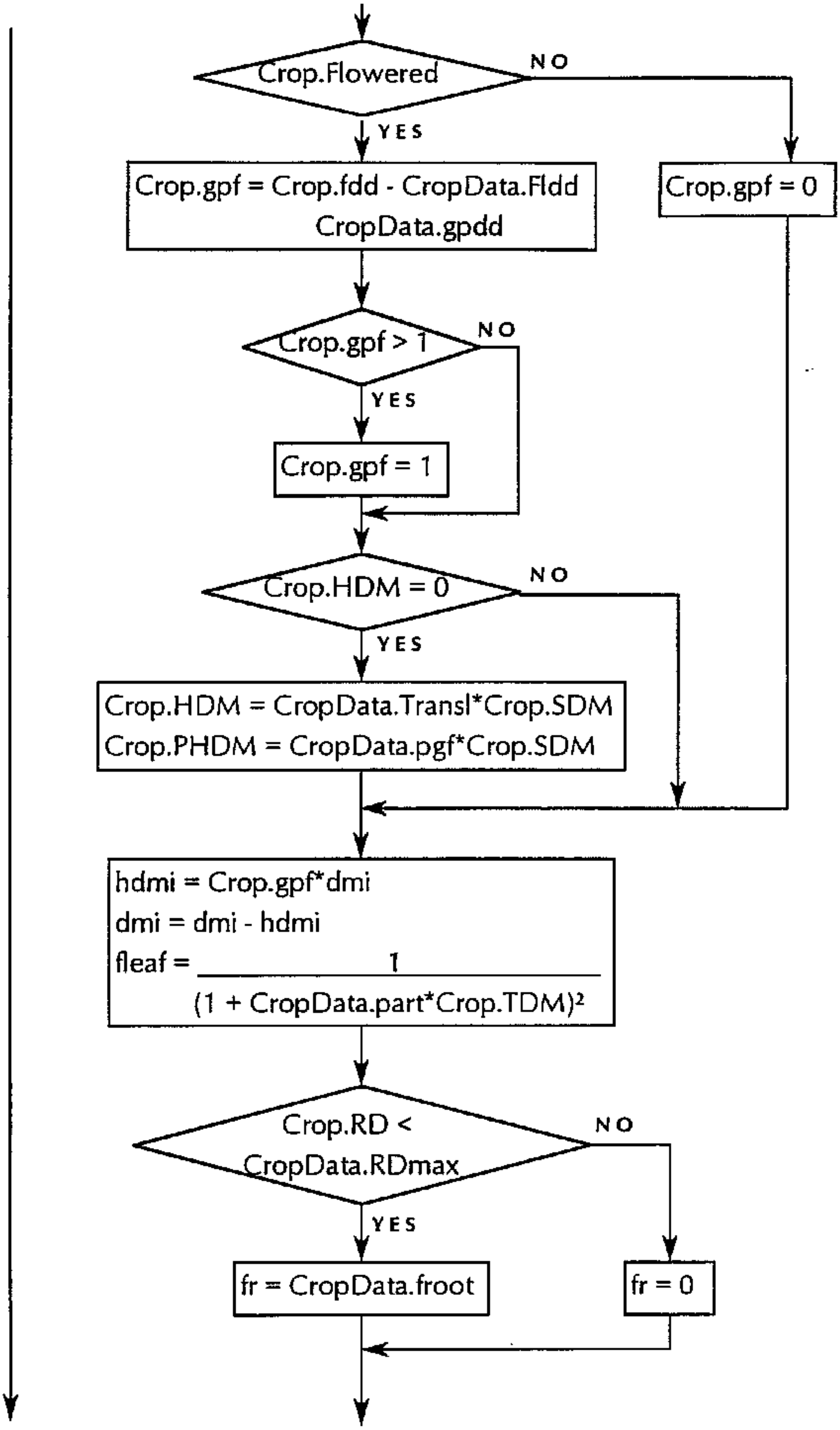
Continue on next page


m = i - 1
2 <= i <= m
NO
YES
Loss = (Crop.Fltransp*CropData.MaxTrans*f[i]*
(Psix - Soil.Spsi[i])/(0.67*CropData.Psilm))/ (Rho_w*dz[i])
SoilData.wc[i] - Loss
< SoilData.pwpwc[i]
NO
YES
Loss = SoilData.wc[i] - SoilData.pwpwc[i]
Soil.ActualTrsp = Soil.ActualTrsp + Loss*Rho_w*dz[i]
SoilData.wc[i] = SoilData.wc[i] - Loss
Inc i
TransDOY = Soil.ActualTrsp
Crop.Fltransp > 0
NO
YES
Soil.Sl = (Soil.ActualTrsp / Crop.Fltransp*PET)
Soil.Sl = 0
END

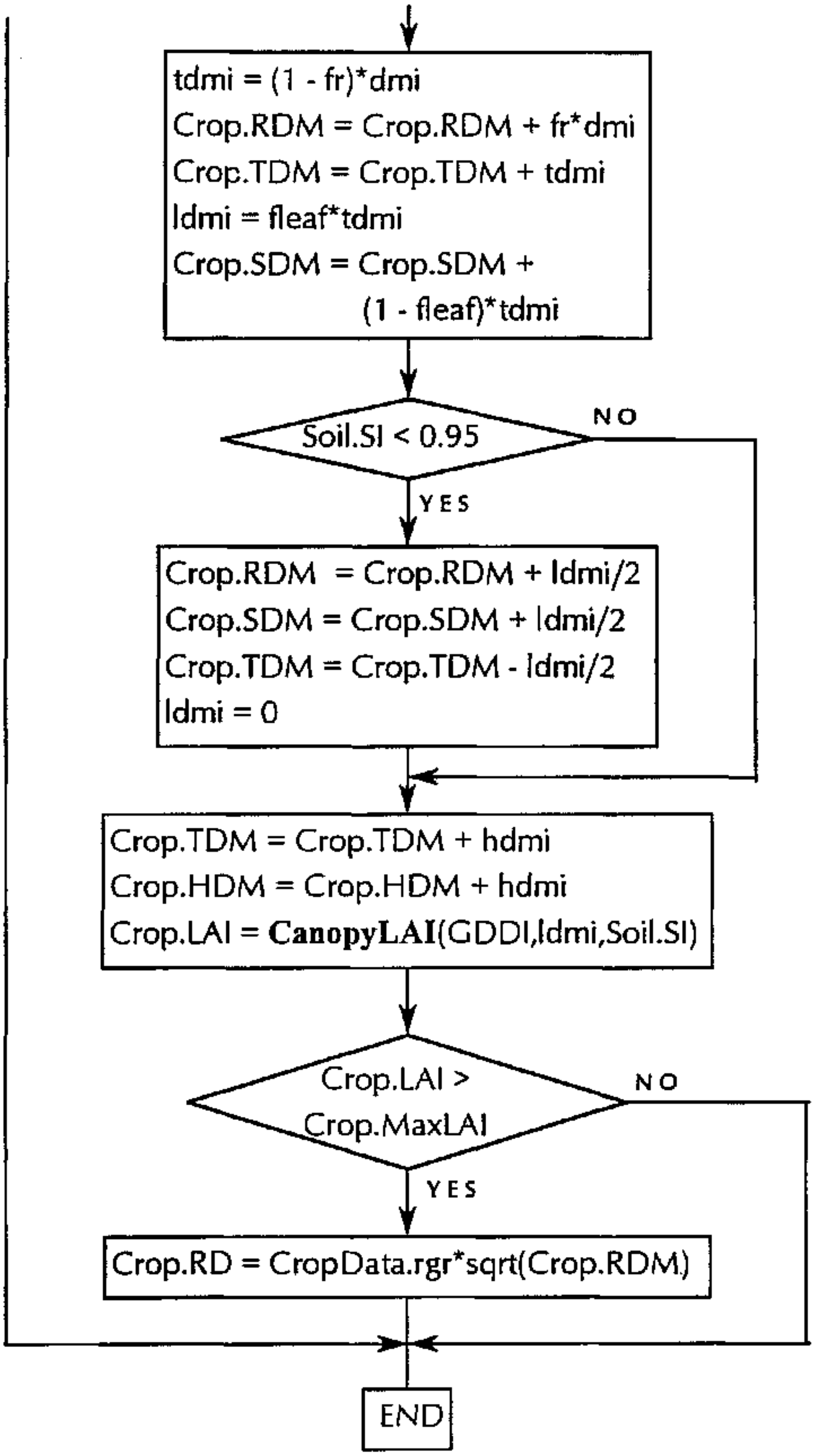

CalcEvaporation
PotentialEvap = (1 - Crop.Flevap) * PET
SoilData.wc[1] < SoilData.pwpwc[1]
NO
YES
PotentialEvap = PotentialEvap*((SoilData.wc[1] - Soil.Adwc) / (SoilData.pwpwc[1] - Soil.Adwc))^2
nwc = SoilData.wc[1] - (PotentialEvap / Rho_w*dz[1])
nwc < Soil.Adwc
NO
YES
nwc = Soil.Adwc
EvapDOY = (SoilData.wc[1] - nwc)*Rho_w*dz[1]
SoilData.wc[1] = nwc
END

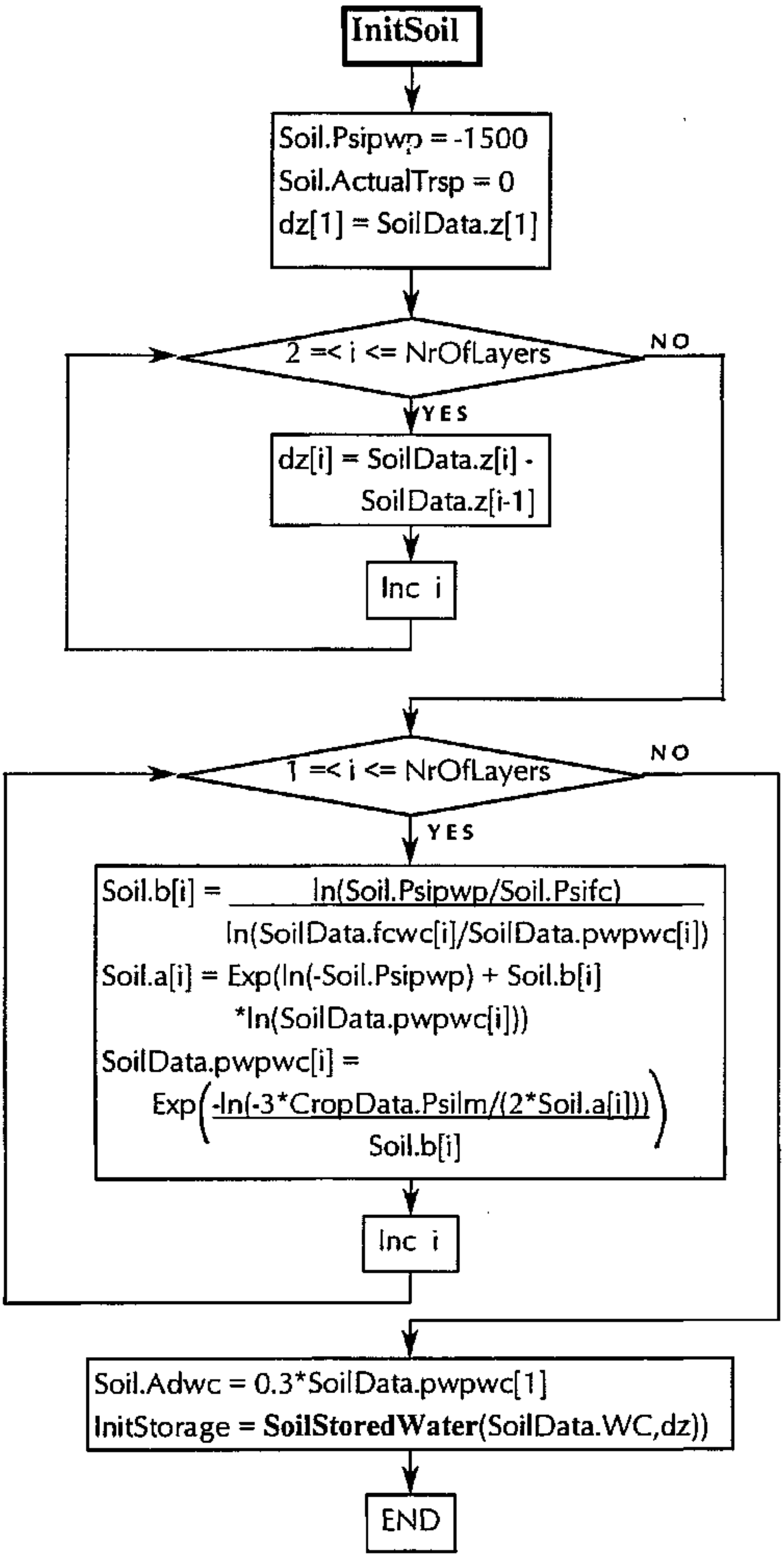


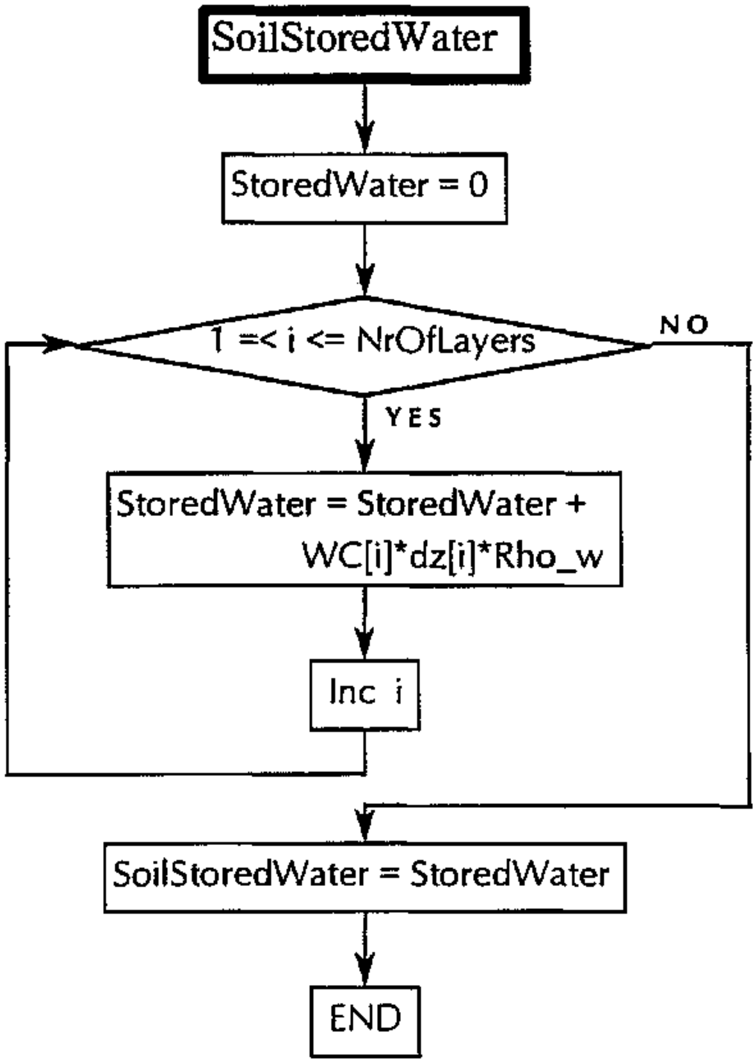
Continue on next page

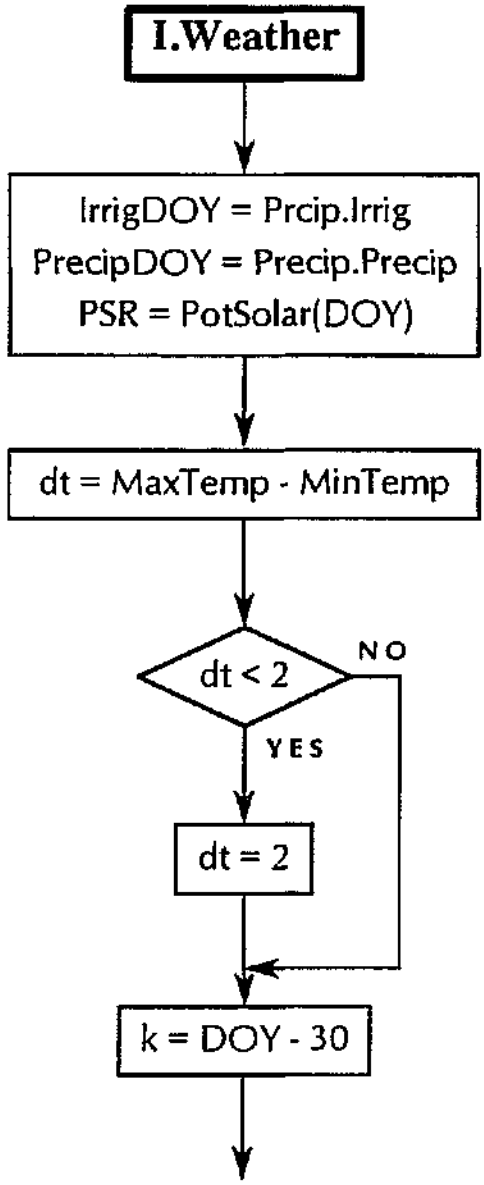


Continue on next page

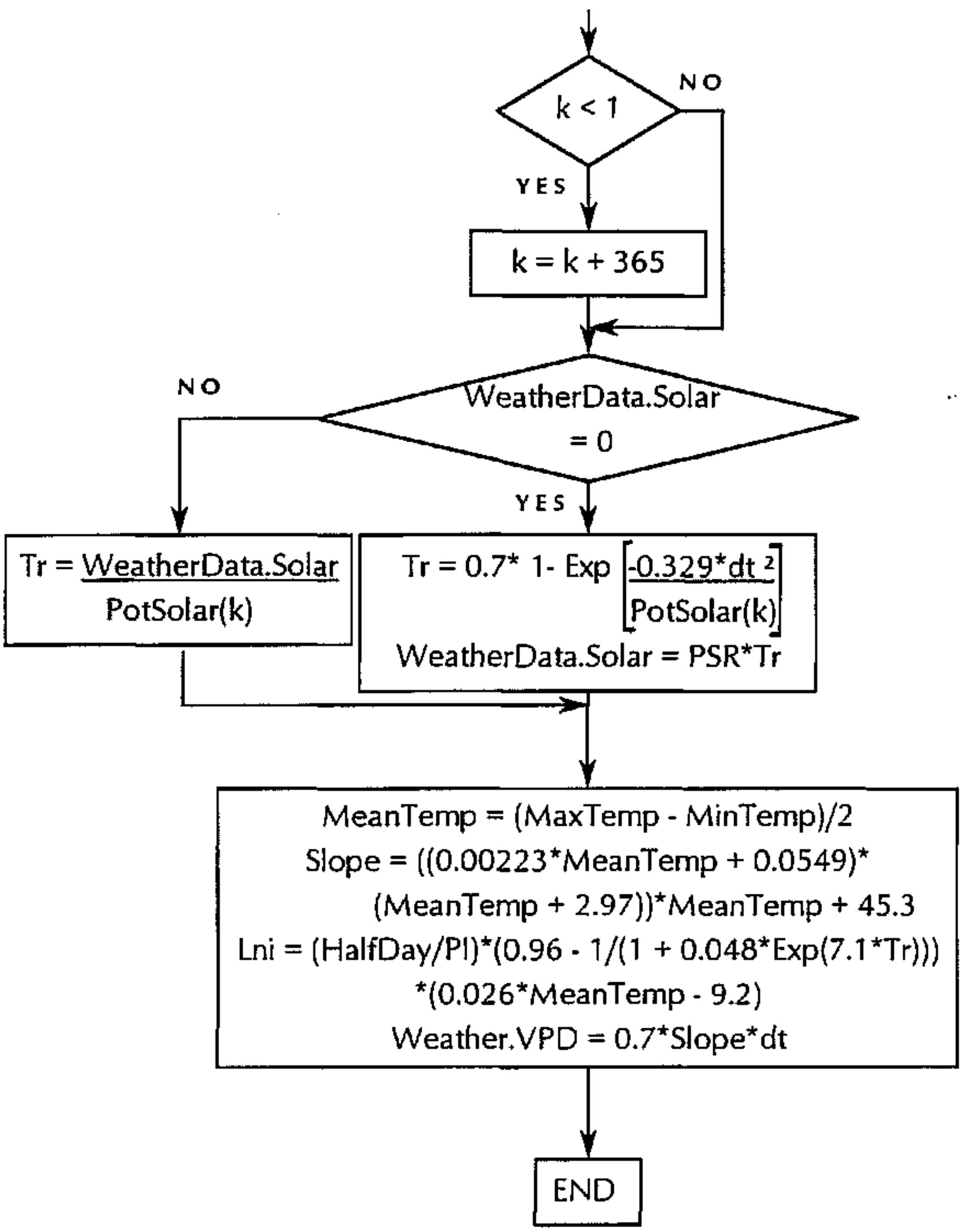

tdmi = (1 - fr)*dmi
Crop.RDM = Crop.RDM + fr*dmi
Crop.TDM = Crop.TDM + tdmi
ldmi = fleaf*tdmi
Crop.SDM = Crop.SDM + (1 - fleaf)*tdmi
Soil.SI < 0.95
NO
YES
Crop.RDM = Crop.RDM + ldmi/2
Crop.SDM = Crop.SDM + ldmi/2
Crop.TDM = Crop.TDM - ldmi/2
ldmi = 0
Crop.TDM = Crop.TDM + hdmi
Crop.HDM = Crop.HDM + hdmi
Crop.LAI = CanopyLAI(GDDI,ldmi,Soil.SI)
Crop.LAI > Crop.MaxLAI
NO
YES
Crop.RD = CropData.rgr*sqrt(Crop.RDM)
END

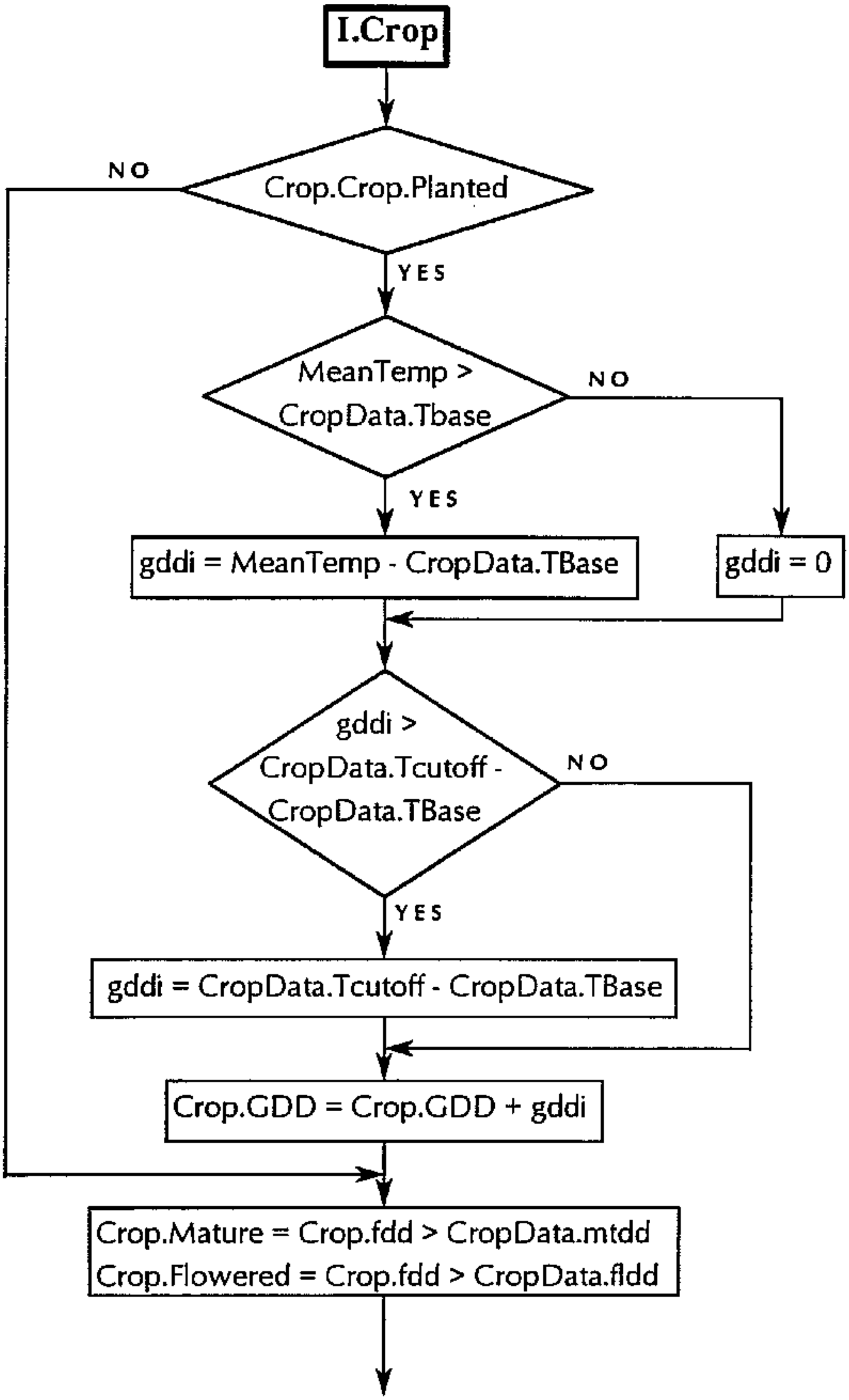

InitSoil
Soil.Psipwp = -1500
Soil.ActualTrsp = 0
dz[1] = SoilData.z[1]
2 =< i <= NrOfLayers
NO
YES
dz[i] = SoilData.z[i] - SoilData.z[i-1]
Inc i
1 =< i <= NrOfLayers
NO
YES
Soil.b[i] = ln(Soil.Psipwp/Soil.Psifc) / ln(SoilData.fcwc[i]/SoilData.pwpwc[i])
Soil.a[i] = Exp(ln(-Soil.Psipwp) + Soil.b[i] *ln(SoilData.pwpwc[i]))
SoilData.pwpwc[i] = Exp(-ln(-3*CropData.Psilm/(2*Soil.a[i])) / Soil.b[i])
Inc i
Soil.Adwc = 0.3*SoilData.pwpwc[1]
InitStorage = SoilStoredWater(SoilData.WC,dz))
END


SoilStoredWater
StoredWater = 0
1 =< i <= NrOfLayers
NO
YES
StoredWater = StoredWater + WC[i]*dz[i]*Rho_w
Inc i
SoilStoredWater = StoredWater
END

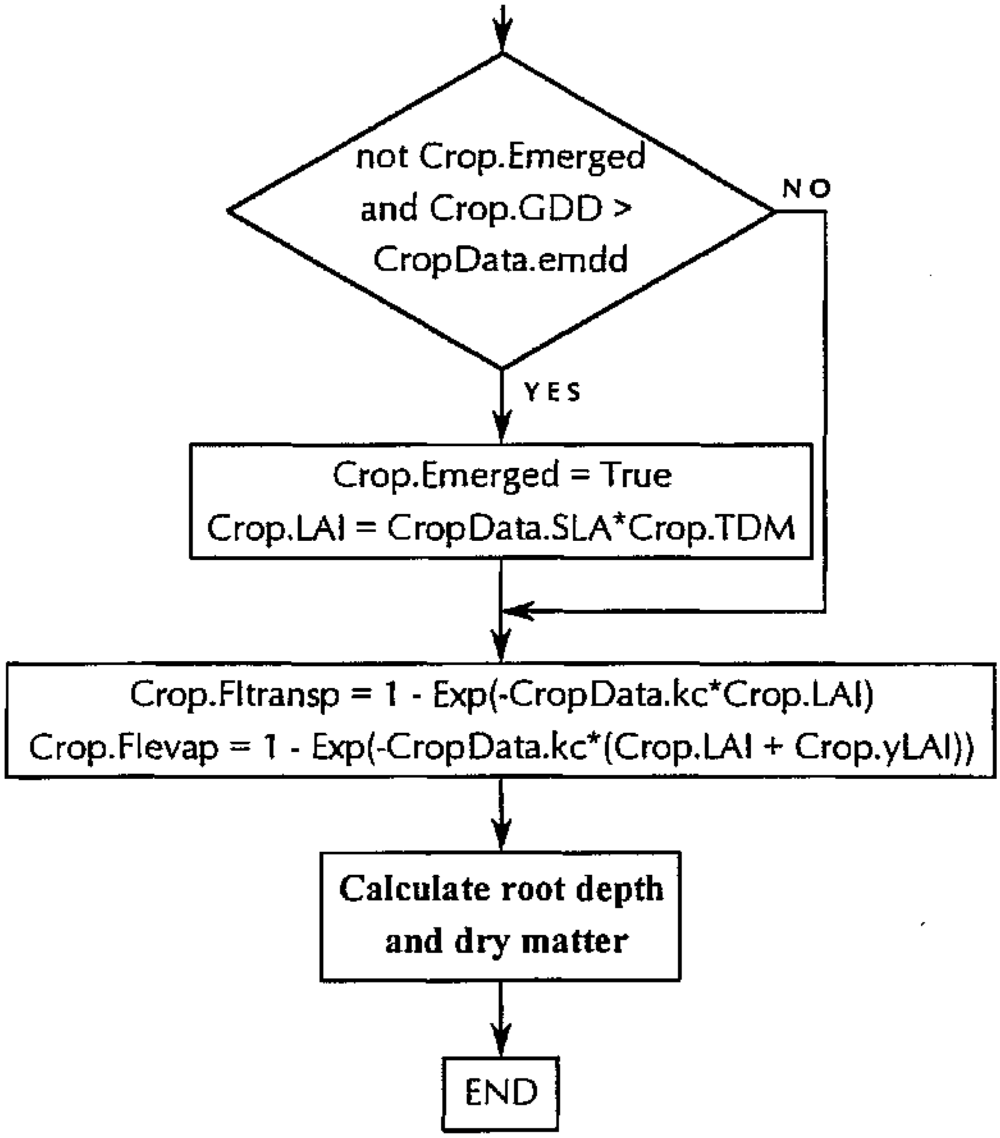


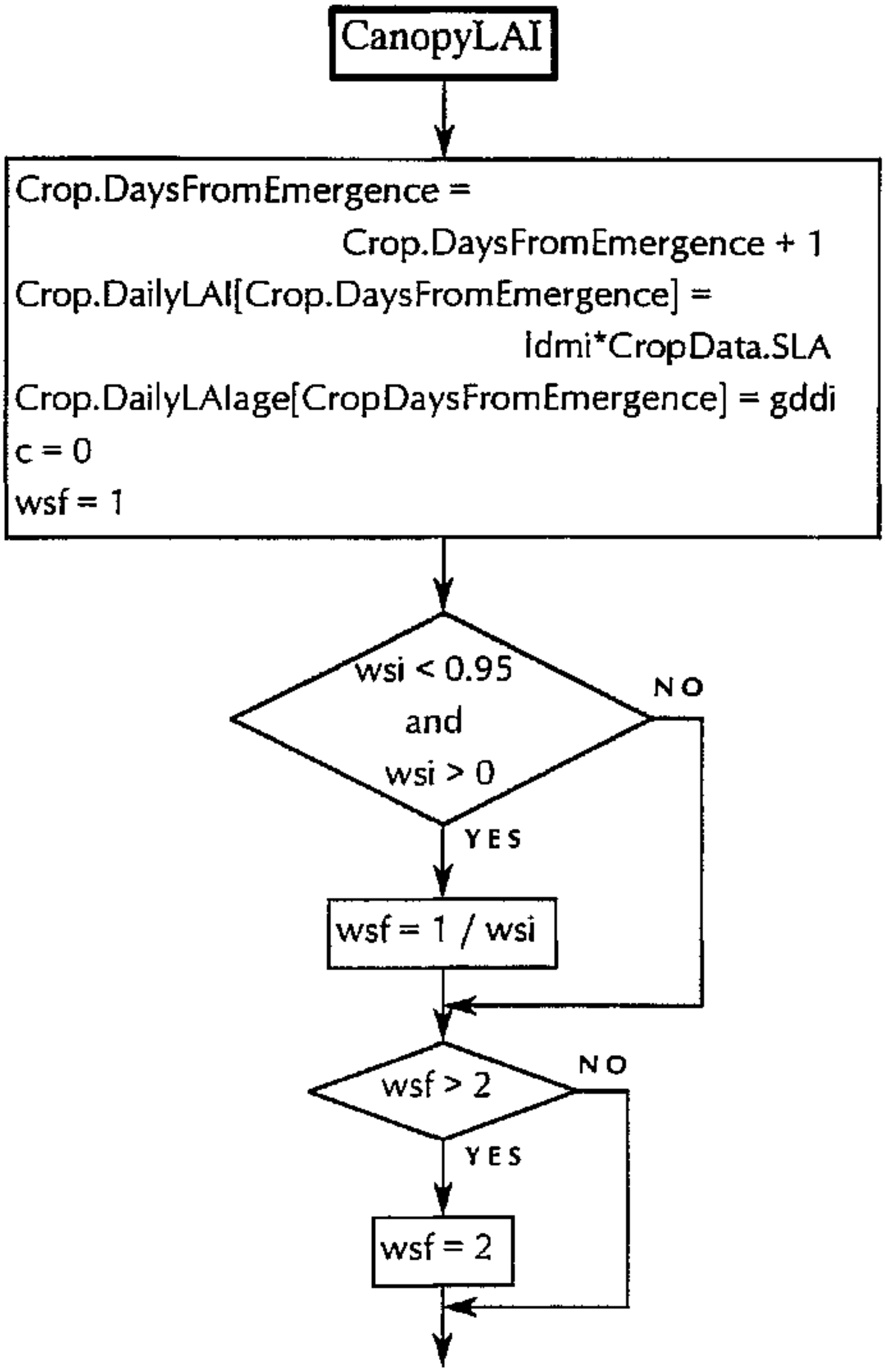
Continue on next page


k < 1
NO
YES
k = k + 365
WeatherData.Solar = 0
NO
YES
Tr = WeatherData.Solar / PotSolar(k)
Tr = 0.7* 1- Exp [-0.329*dt 2 / PotSolar(k)]
WeatherData.Solar = PSR*Tr
MeanTemp = (MaxTemp - MinTemp)/2
Slope = ((0.00223*MeanTemp + 0.0549)*
(MeanTemp + 2.97))*MeanTemp + 45.3
Lni = (HalfDay/PI)*(0.96 - 1/(1 + 0.048*Exp(7.1*Tr)))
*(0.026*MeanTemp - 9.2)
Weather.VPD = 0.7*Slope*dt
END

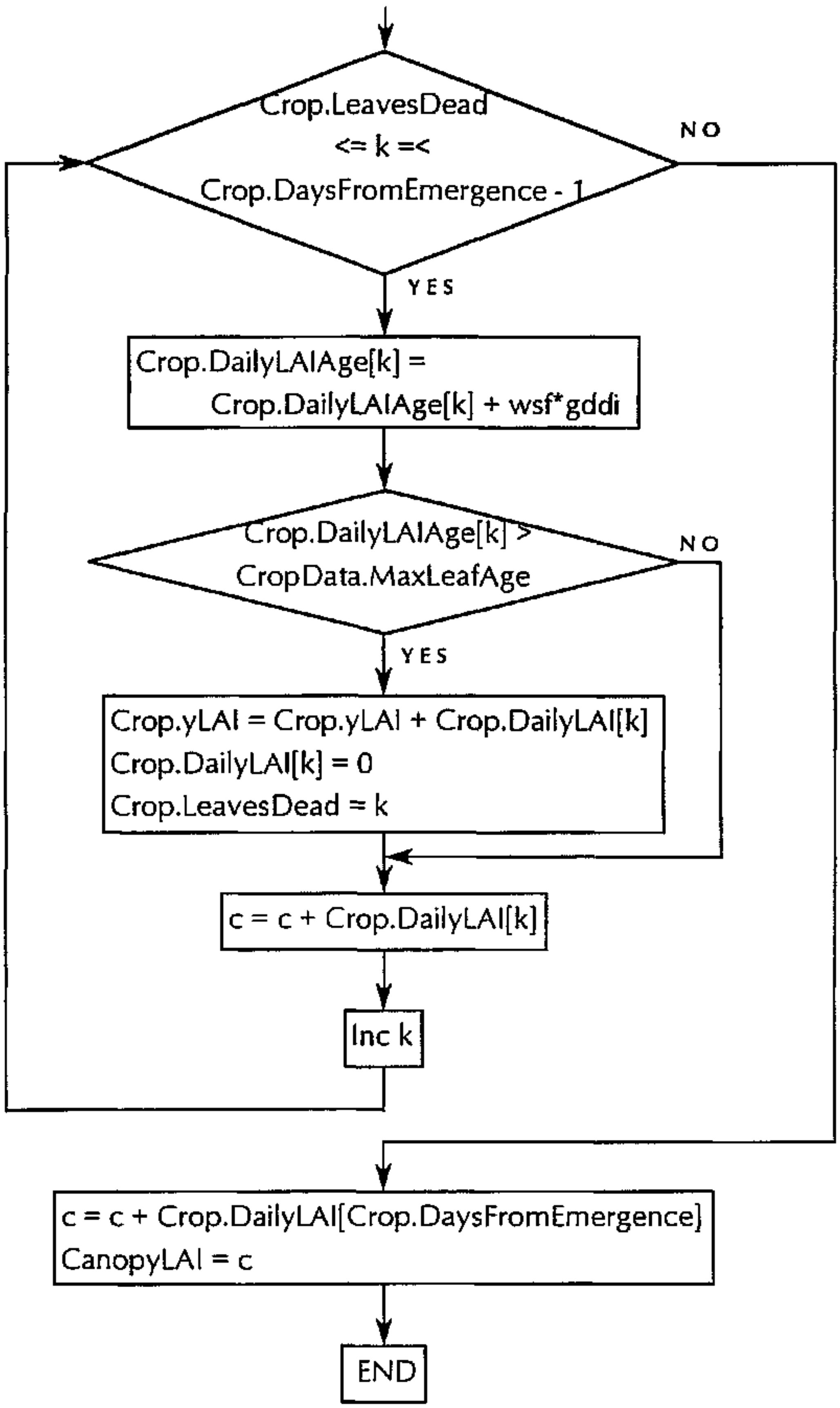


Continue on next page


not Crop.Emerged
and Crop.GDD >
CropData.emdd
NO
YES
Crop.Emerged = True
Crop.LAI = CropData.SLA*Crop.TDM
Crop.FItransp = 1 - Exp(-CropData.kc*Crop.LAI)
Crop.Flevap = 1 - Exp(-CropData.kc*(Crop.LAI + Crop.yLAI))
Calculate root depth
and dry matter
END



Continue on next page


Crop.LeavesDead
<= k =<
Crop.DaysFromEmergence - 1
NO
YES
Crop.DailyLAIAge[k] =
Crop.DailyLAIAge[k] + wsf*gddi
Crop.DailyLAIAge[k] >
CropData.MaxLeafAge
NO
YES
Crop.yLAI = Crop.yLAI + Crop.DailyLAI[k]
Crop.DailyLAI[k] = 0
Crop.LeavesDead = k
c = c + Crop.DailyLAI[k]
Inc k
c = c + Crop.DailyLAI[Crop.DaysFromEmergence]
CanopyLAI = c
END

APPENDIX C SOURCE CODE

```
unit _IRun;

interface

uses
   Crt,
   PXEngine,
   PXMsg,
  _InOut,
  _Menus,
  _Env,
  _Strings,
  _System,
  _IEnv,
  _IPXEng,
  _IGlob,
  _IUtil,
  _IWeathr,
  _ISoil,
  _ICrop,
  _FldTbl,
  _CropTbl,
  _SoilTbl,
  _DayWTbl,
  _WCorTbl,
  _ResTbl,
  _PrcpTbl,
  _WIDTbl,
  _IView,
  _IOutp1,
  _IOutp5;

procedure RunProc;

implementation

var
  DOYDate: DateStr;

procedure InitVars;

begin {procedure InitVars}
  FillChar(Crop,SizeOf(Crop),#0);
  FillChar(SoilData,SizeOf(SoilData),#0);
  FillChar(ResultData,SizeOf(ResultData),#0);
  DAP       := 0;
  DOY       := 0;
  PrecipDOY := 0;
  IrrigDOY  := 0;
  EvapDOY   := 0;
  TransDOY  := 0;
  DrainDOY  := 0;
  InterDOY  := 0;
  RunOffDOY := 0;
  PET       := 0;
end; {procedure InitVars}

procedure FillResultBuff;

begin {procedure FillResultBuff}
```

```
  ResultData.DAP    := DAP;
  ResultData.DOY    := DOY;
  ResultData.LAI    := Crop.LAI;
  ResultData.RD     := Crop.RD;
  ResultData.TDM    := Crop.TDM;
  ResultData.HDM    := Crop.HDM;
  ResultData.WC     := SoilData.WC;
  ResultData.Precip := PrecipDOY;
  ResultData.Irrig  := IrrigDOY;
  ResultData.Evap   := EvapDOY;
  ResultData.Trans  := TransDOY;
  ResultData.Drain  := DrainDOY;
  ResultData.Inter  := InterDOY;
  ResultData.RunOff := RunOffDOY;
  ResultData.PET    := PET;
  ResultData.ProfileWC := SoilStoredWater(SoilData.WC,dz,SoilData.z[NrOfLayers]{CropData.RDmax});
  ResultData.AllowDepl := AllowableDepletionProc(SoilData.FCWC,SoilData.PWPWC,dz,ResultData.RD,PercentDepletion);
  ResultData.Deficit   := ResultData.ProfileWC - ProfileFC;
  ResultData.StressDay := StressDay;
end; {procedure FillResultBuff}

procedure CheckWCUpdateTbl(PerseelID: PerseelStr;
                           DOYDate:   DateStr);
var
  PxErr: Integer;

begin {procedure CheckWCUpdateTbl}
  PxErr := FindWCorr(PerseelID,DOYDate);
  if PxErr = PxSuccess then
    SoilData.WC := WCorr.WCorr;
end; {procedure CheckWCUpdateTbl}

function Run(PerseelID: PerseelStr;
             StartDate,
             EndDate:   DateStr): Integer;
var
  PxErr:        Integer;
  DOYDateEnc,
  PlantDateEnc,
  StartDateEnc,
  EndDateEnc:   TDate;

begin {function Run}
  InitVars;
  DisplayNewMessageLine(MessageLine,'  Calculate . . .',[]);
  DateEncode(Field.PlantDate,PlantDateEnc);
  DateEncode(EndDate,EndDateEnc);
  if StartDate <> '' then
    begin
      DateEncode(StartDate,StartDateEnc);
      if StartDateEnc > PlantDateEnc then
        DOYDate := Field.PlantDate
      else
        DOYDate := StartDate;
    end
  else
    DOYDate := Field.PlantDate;
  DOY := DayOfYearFromDate(DOYDate) + DAP;
  PxErr := FindWeathrID(Field.WeatherID);
  if PxErr = PxSuccess then
```

```
    PxErr := FindSoil(Field.PerseelID);
  if PxErr = PxSuccess then
    PxErr := FindCrop(Field.Crop);

  if PxErr = PxSuccess then
    begin
      CheckWCUpdateTbl(Field.PerseelID,DOYDate);
      InitCrop;
      InitSoil;
      InitWeather;
    end; {if}

  if PxErr = PxSuccess then
    begin
      DateEncode(DOYDate,DOYDateEnc);
      FillResultBuff;
      PxErr := InsertResult(Field.PerseelID,DOYDate);
    end; {if}

  if (PxErr = PxSuccess) and (DOYDateEnc < EndDateEnc) then
    repeat
      ChangeMessageLine(MessageLine,' Field: '+Field.PerseelID+'  Date: '+DOYDate,[]);
      if DOYDate = Field.PlantDate then
        PlantCrop;

      DOYDate := IncDate(DOYDate,1);
      DOY := DayOfYearFromDate(DOYDate);
      DateEncode(DOYDate,DOYDateEnc);
      if DOYDateEnc > PlantDateEnc then
        Inc(DAP);

      if Crop.CropPlanted and not Crop.Emerged then    {Emerged}
        PercentDepletion := CropData.EmergeDepl
      else if Crop.Emerged and not Crop.Flowered then  {Vegetative}
        PercentDepletion := CropData.VegetateDepl
      else if Crop.Flowered then                       {Reproductive}
        PercentDepletion := CropData.ReproDepl
      else
        PercentDepletion := 0;

      if PxErr = PxSuccess then
        PxErr := WeatherDayStep(WeatherID,DOYDate,DOY);
      if PxErr = PxSuccess then
        begin
          CheckWCUpdateTbl(Field.PerseelID,DOYDate);
          SoilDayStep;
          CropDayStep;
          FillResultBuff;
          PxErr := InsertResult(Field.PerseelID,DOYDate);
        end; {if}
    until Crop.Mature or (DOYDateEnc >= EndDateEnc) or (PxErr <> PxSuccess);
  DisplayPrevMessageLine(MessageLine);
  Run := PxErr;
end; {function Run}

function SingleFromStart: Integer;

var
  RunFromDate,
  RunToDate:   DateStr;
```

```
  PxErr:       Integer;
  EscPressed:  Boolean;

begin {function SingleFromStart}
  PxErr := PxTblEmpty(ResultTblName);
  if PxErr = PxSuccess then
    PxErr := OpenTbl(ResultTblName,ResultTblHandle,ResultRecHandle,0,False);
  if PxErr = PxSuccess then
    begin
      ResultTblOpened := True;
      InitField;
      Field.PerseelID := ToUpper(AutoGetString('Field ID',20,''));
      if Field.PerseelID <> #0 then
        begin
          PxErr := FindField(Field.PerseelID);
          if PxErr = PxSuccess then
            begin
              RunFromDate := AutoGetDate('From Date (DD.MM.YY)',24,Field.StartDate);
              if RunFromDate <> #0 then
                begin
                  RunToDate := AutoGetDate('To Date (DD.MM.YY)',24,GlobDate);
                  if RunToDate <> #0 then
                    begin
                      PxErr := Run(Field.PerseelID,RunFromDate,RunToDate);
                      if PxErr = PxSuccess then
                        PxErr := DrawGraphProc(1,Field.PerseelID,VarArray);
                    end; {if}
                end; {if}
            end; {if}
        end; {if}
    end; {if}
  CloseTbl(ResultRecHandle,ResultTblHandle);
  ResultTblOpened := False;
  SingleFromStart := PxErr;
end; {function SingleFromStart}

function GroupFromStart: Integer;

var
  TermCh:      Char;
  RunFromDate,
  RunToDate:   DateStr;
  PxErr:       Integer;
  EscPressed:  Boolean;
  GroupNr:     Word;

begin {function GroupFromStart}
  PxErr := PxTblEmpty(ResultTblName);
  if PxErr = PxSuccess then
    PxErr := OpenTbl(ResultTblName,ResultTblHandle,ResultRecHandle,0,False);
  if PxErr = PxSuccess then
    PxErr := OpenTbl(FieldTblName,FieldTblHandle,FieldRecHandle,0,False);
  if PxErr = PxSuccess then
    begin
      ResultTblOpened := True;
      FieldTblOpened := True;
      GroupNr := AutoGetInt('Group number',1,1,10000,EscPressed);
      if not EscPressed then
        begin
          PxErr := PXRecFirst(FieldTblHandle);
```

```
            if PxErr = PxSuccess then
              repeat
                PxErr := GetFieldRec;
                if (PxErr = PxSuccess) and (Field.Group = GroupNr) and Field.SchedFlag then
                  PxErr := Run(Field.PerseelID,Field.StartDate,GlobDate);
                if (PxErr = PXERR_TABLEEMPTY) or (PxErr = PXERR_RECNOTFOUND) or (PxErr = PXERR_ENDOFTABLE) then
                  PxErr := PxSuccess;
                PxErr := PXRecNext(FieldTblHandle);
                if KeyPressed then
                  TermCh := ReadKey;
              until (TermCh = EscCh) or (PxErr <> PxSuccess);
          end; {if}
      end; {if}
    if PxErr = PXERR_ENDOFTABLE then
      PxErr := PxSuccess;
    CloseTbl(ResultRecHandle,ResultTblHandle);
    CloseTbl(FieldRecHandle,FieldTblHandle);
    ResultTblOpened := False;
    FieldTblOpened := False;
    GroupFromStart := PxErr;
  end; {function GroupFromStart}

  function AllFromStart: Integer;

  var
    TermCh:      Char;
    RunFromDate,
    RunToDate:   DateStr;
    PxErr:       Integer;
    EscPressed:  Boolean;

  begin {function AllFromStart}
    PxErr := PxTblEmpty(ResultTblName);
    if PxErr = PxSuccess then
      PxErr := OpenTbl(ResultTblName,ResultTblHandle,ResultRecHandle,0,False);
    if PxErr = PxSuccess then
      PxErr := OpenTbl(FieldTblName,FieldTblHandle,FieldRecHandle,0,False);
    if PxErr = PxSuccess then
      begin
        ResultTblOpened := True;
        FieldTblOpened := True;
        PxErr := PXRecFirst(FieldTblHandle);
        if PxErr = PxSuccess then
          repeat
            PxErr := GetFieldRec;
            if (PxErr = PxSuccess) and Field.SchedFlag then
              PxErr := Run(Field.PerseelID,Field.StartDate,GlobDate);
            if (PxErr = PXERR_TABLEEMPTY) or (PxErr = PXERR_RECNOTFOUND) or (PxErr = PXERR_ENDOFTABLE) then
              PxErr := PxSuccess;
            PxErr := PXRecNext(FieldTblHandle);
            if KeyPressed then
              TermCh := ReadKey;
          until (TermCh = EscCh) or (PxErr <> PxSuccess);
      end; {if}
    if PxErr = PXERR_ENDOFTABLE then
      PxErr := PxSuccess;
    CloseTbl(ResultRecHandle,ResultTblHandle);
    CloseTbl(FieldRecHandle,FieldTblHandle);
    ResultTblOpened := False;
    FieldTblOpened := False;
```

```
  AllFromStart := PxErr;
end; {function AllFromStart}

procedure RunProc;

var
  PxErr: Integer;

begin {procedure RunProc}
  PxErr := PxSuccess;
  OutputDest := TheScreen;
  UpdateEnviroment;
  FillChar(VarArray,SizeOf(VarArray),0);
  VarArray[1] := 5;        {WC}
  VarArray[2] := 6;        {precip+irrig}
  VarArray[3] := 13;       {field capacity}
  VarArray[4] := 14;       {wilting point}
  case CurrentMenu^.CurrentItemNo of
    1: PxErr := SingleFromStart;
    2: PxErr := GroupFromStart;
    3: PxErr := AllFromStart;
  end; {case}
  if PxErr <> PxSuccess then
    OutputErrorMsg(PxErrMsg(PxErr),Nil);
end; {procedure RunProc}

end. {unit _IRun}
```

Vrae:

1. Plant crop is volgens my op die verkeerde plek.
2. Gaan die Inc(DOY) en Inc(DAP) na. Wanneer moet hulle geinkrementeer word?

```
unit _ICrop;   {crop unit with dry matter to roots}

interface

uses
  _IGlob,
  _IWeathr,
  _CropTbl,
  _DayWTbl;

type
  CropRecord = record
    TDM,                 {canopy dry matter at emergence}
    SDM,
    HDM,                 {havestable dry matter}
    PHDM,
    LAI,                 {leaf area index}
    yLAI,
    GDD,                 {growing day degrees}
    FItransp,
    FIevap,              {fractional inteception (solar radiation) evaporation}
    RD,                  {root depth}
    RDM,
    gpf,

    MaxLAI,              {maximum leaf area index}
    fdd:               Real;
    DaysFromEmergence,
    LeavesDead:        Integer;
    DailyLAI,
    DailyLAIAge:       Array[0..400] of Real;
    CropPlanted,
    Mature,
    Flowered,
    Vegetative,
    Emerged:           Boolean;
  end; {CropRecord}

var
  Crop: CropRecord;

procedure InitCrop;
procedure PlantCrop;
procedure CropDayStep;

implementation

uses
  _ISoil;

procedure InitCrop;

var
  i: Integer;

begin {procedure InitCrop}
  Crop.TDM      := 0;
  Crop.LAI      := 0;
  Crop.RD       := 0;
```

```
  Crop.yLAI      := 0;
  Crop.FIevap    := 0;
  Crop.FItransp  := 0;
  Crop.PHDM      := 0;
  Crop.HDM       := 0;
  Crop.GDD       := 0;
  Crop.fdd       := 0;
  Crop.SDM       := 0;
  FillChar(Crop.DailyLAI,SizeOf(Crop.DailyLAI),#0);
  FillChar(Crop.DailyLAIAge,SizeOf(Crop.DailyLAIAge),#0);
  Crop.CropPlanted        := False;
  Crop.DaysFromEmergence := 0;
  Crop.LeavesDead         := 0;
  Crop.Emerged            := False;
  Crop.Flowered           := False;
  Crop.Vegetative         := Crop.Emerged and not Crop.Flowered;
  Crop.Mature             := False;
  Crop.MaxLAI             := 0;
end; {procedure InitCrop}

procedure PlantCrop;

begin {procedure PlantCrop}
  Crop.CropPlanted        := True;
  Crop.GDD                := 0;
  Crop.fdd                := 0;
  Crop.HDM                := 0;
  Crop.TDM                := CropData.TDMstart;
  Crop.RDM                := CropData.froot*Crop.TDM/(1-CropData.froot);
  Crop.FItransp           := 0;
  Crop.SDM                := 0;
  Crop.DailyLAI[0]        := CropData.SLA*Crop.TDM;
  Crop.DaysFromEmergence := 0;
  Crop.LeavesDead         := 0;
  Crop.Emerged            := False;
  Crop.Flowered           := False;
  Crop.Mature             := False;
  Crop.MaxLAI             := 0;
end; {procedure PlantCrop}

procedure CropDayStep;

var
  i:                  Integer;
  gddi,tfact,dmiw,
  dmis,dmi,ldmi,tdmi,
  hdmi,fleaf,fr:      Real;

  function CanopyLAI(gddi,ldmi,wsi: Real): Real;

  var
    k,l,m: Integer;
    wsf,c: Real;

  begin {function CanopyLAI}
    Inc(Crop.DaysFromEmergence);
    Crop.DailyLAI[Crop.DaysFromEmergence]    := ldmi*CropData.SLA;
    Crop.DailyLAIAge[Crop.DaysFromEmergence] := gddi;
    c   := 0;
```

```
    wsf := 1;
    if (wsi < CropData.StressIndex) and (wsi > 0) then   {!}
      wsf := 1/wsi;
    if wsf > 2 then
      wsf := 2;
    for k := Crop.LeavesDead to Crop.DaysFromEmergence-1 do
      begin
        Crop.DailyLAIAge[k] := Crop.DailyLAIAge[k] + wsf*gddi;
        if Crop.DailyLAIAge[k] > CropData.MaxLeafAge then
          begin
            Crop.yLAI        := Crop.yLAI + Crop.DailyLAI[k];
            Crop.DailyLAI[k] := 0;
            Crop.LeavesDead  := k;
          end; {if}
        c := c + Crop.DailyLAI[k];
      end; {for}
    c := c + Crop.DailyLAI[Crop.DaysFromEmergence];
    CanopyLAI := c;
  end; {function CanopyLAI}

begin {procedure CropDayStep}
  StressDay := 0;                               {!}
  if Crop.CropPlanted then
    begin
      {CALCULATION OF GROWING DAY DEGREES, Tb = 0 C AND PHYS. TIME}
      if MeanTemp > CropData.Tbase then
        gddi := MeanTemp-CropData.TBase
      else
        gddi := 0;
      if gddi > CropData.Tcutoff(-CropData.TBase) then   {!}
        gddi := CropData.Tcutoff-CropData.TBase;
      Crop.GDD := Crop.GDD + gddi;
{
      if (DOY > 31) and (DOY < 343) then
        Crop.fdd := Crop.fdd + gddi
      else
        Crop.fdd := 0;
}
    end;  {if CropPlanted}

  Crop.Mature   := Crop.gdd > CropData.mtdd; {!}
  Crop.Flowered := Crop.gdd > CropData.fldd; {!} {fdd instead of gdd for vernalization}

  if not Crop.Emerged and (Crop.GDD > CropData.emdd) then
    begin
      Crop.Emerged := True;
      Crop.LAI     := CropData.SLA*Crop.TDM;
    end; {if}

  Crop.FItransp := 1 - Exp(-CropData.kc*Crop.LAI);
  Crop.FIevap   := 1 - Exp(-CropData.kc*(Crop.LAI+Crop.yLAI));

  {rooting depth and dry matter calculation}
  if Crop.Emerged and not Crop.Mature then
    begin
      if gddi >= (CropData.Tlo-CropData.Tbase) then
        tfact := 1
      else
        tfact := gddi/(CropData.Tlo-CropData.Tbase);
```

```
        dmiw := Soil.ActualTrsp*CropData.DWR/WeatherData.VPD;    {dm is in kg/m2 (*10 to get to t/ha), at is in mm}
        dmis := tfact*CropData.ConvEff*Crop.FItransp*WeatherData.Solar;
        if dmis < dmiw then
          dmi := dmis
        else
          dmi := dmiw;
        ldmi := 0;
        if Crop.Flowered then
          begin
            Crop.gpf := (Crop.gdd-CropData.fldd)/CropData.gpdd;   {!}
            if Crop.gpf > 1 then
              Crop.gpf := 1;
            if Crop.HDM = 0 then
              begin                    {initiate grain fill first day flowering}
                Crop.HDM  := CropData.Transl*Crop.SDM;
                Crop.PHDM := CropData.pgf*Crop.SDM;
              end; {if}
          end {if}
        else
          Crop.gpf := 0;
        hdmi  := Crop.gpf*dmi;
        dmi   := dmi - hdmi;                {subtract reproductive growth first}
        fleaf := 1/sqr(1+CropData.part*Crop.TDM);
        if Crop.RD < CropData.RDmax then
          fr := CropData.froot{/(1+2*RDM)}
        else
          fr := 0;
        tdmi := (1-fr)*dmi;
        Crop.RDM  := Crop.RDM + fr*dmi;
        Crop.TDM  := Crop.TDM + tdmi;
        ldmi      := fleaf*tdmi;
        Crop.SDM  := Crop.SDM + (1-fleaf)*tdmi;
        if Soil.SI < CropData.StressIndex then
          begin                  {partition more to roots if stressed}

            if fr > 0 then                        {!}
              begin
                Crop.RDM  := Crop.RDM + ldmi/2;
                Crop.SDM  := Crop.SDM + ldmi/2;
                Crop.TDM  := Crop.TDM - ldmi/2;
              end
            else
              Crop.SDM  := Crop.SDM + ldmi;       {!}

            ldmi      := 0;
            StressDay := 1;                       {!}
          end; {if}
        Crop.TDM := Crop.TDM + hdmi;
        Crop.HDM := Crop.HDM + hdmi;
        Crop.LAI := {SLA*TDM/(1+part*TDM)}CanopyLAI(GDDI,ldmi,Soil.SI);
        if Crop.LAI > Crop.MaxLAI then
          Crop.MaxLAI := Crop.LAI;
        Crop.RD := CropData.RGR*Sqrt(Crop.RDM);{2.07*RDM/(0.0427+RDM);RDmax/(1+48*Exp(-8.5*GDD/fldd));}
      end; {if}
end; {procedure CropDayStep}

end. {unit _ICrop}
```

```
unit _ISoil;

interface

uses
   PXEngine,
   PXMsg,
  _InOut,
  _IGlob,
  _Strings,
  _IWeathr,
  _FldTbl,
  _CropTbl,
  _SoilTbl,
  _DayWTbl,
  _PrcpTbl,
  _ICrop;

type
  SoilRecord = record
    ActualTrsp,
    Adwc,                      {air dry water content}
    SI:        Real;
    a,
    b,
    Spsi:      SoilArray;
  end; {SoilRecord}

var
{  InitStorage: Real;}
  dz:          SoilArray;     {soil layers}
  Soil:        SoilRecord;

procedure CalcSoildz(z:      SoilArray;
                     var dz: SoilArray);
procedure InitSoil;
procedure SoilDayStep;
function SoilStoredWater(WC,dz: SoilArray;
                         RD:    Real): Real;
function AllowableDepletionProc(FCWC,PWPWC,dz:    SoilArray;
                                RD:               Real;
                                PercentDepletion: Real): Real;

implementation

procedure CalcSoildz(z:      SoilArray;
                     var dz: SoilArray);
var
  i: Integer;

begin {procedure CalcSoildz}
  dz[1] := z[1];
  for i := 2 to NrOfLayers do
    dz[i] := z[i] - z[i-1];
end; {procedure CalcSoildz}

procedure InitSoil;

var
```

```
  i: Integer;

begin {procedure InitSoil}
  CalcSoildz(SoilData.z,dz);
  for i := 1 to NrOfLayers do
    begin
      Soil.b[i] := ln(Field.PsiPWP/Field.PsiFC)/ln(SoilData.fcwc[i]/SoilData.pwpwc[i]);
      Soil.a[i] := Exp(ln(-Field.PsiPWP) + Soil.b[i]*ln(SoilData.pwpwc[i]));
      SoilData.pwpwc[i] := Exp(-ln(-3*CropData.Psilm/(2*Soil.a[i]))/Soil.b[i]);    {plant lower limit}
      if SoilData.wc[i] > SoilData.fcwc[i] then
        begin
          SoilData.wc[i] := SoilData.fcwc[i];
          OutputErrorMsg(Field.PerseelID+': Initial WC['+IntToStr(i)
                         +'] exceeds field capacity',Nil);
        end; {if}
    end; {for}
  Soil.Adwc := 0.3*SoilData.pwpwc[1];   {air dry water content}
  ProfileFC := SoilStoredWater(SoilData.FCWC,dz,SoilData.z[NrOfLayers](CropData.RDmax));
end; {procedure InitSoil}

procedure CalcInterception;  {calculate interception by canopy}

begin {procedure CalcInterception}
  InterDOY := 0;
  if Precip.Precip > 0 then
    begin
      InterDOY := Crop.FIevap;
      Precip.Precip := Precip.Precip - InterDOY;     {1 mm interception for all cover}
      if Precip.Precip < 0 then
        begin
          InterDOY := InterDOY + Precip.Precip;
          Precip.Precip := 0;
        end; {if}
    end; {if}
end; {procedure CalcInterception}

procedure CalcRunOff;

begin {procedure CalcRunOff}
  RunOffDOY := 0;
  if Precip.Precip > 0 then
    begin
      if Precip.Precip <= 0.2*SoilData.Rop then
        RunOffDOY := 0
      else
        RunOffDOY := Sqr(Precip.Precip - 0.2*SoilData.Rop)/(Precip.Precip + 0.8*SoilData.Rop);
      Precip.Precip := Precip.Precip - RunOffDOY;
    end; {if}
end; {procedure CalcRunOff}

procedure CalcInfiltration;

var
  i: Integer;

begin {procedure CalcInfiltration}
  i := 1;
  DrainDOY := 0;
  while (Precip.Precip > 0) and (i <= NrOfLayers) do
```

```
    begin
      if Precip.Precip >= (SoilData.fcwc[i]-SoilData.wc[i])*Rho_w*dz[i] then
        begin
          Precip.Precip := Precip.Precip - (SoilData.fcwc[i]-SoilData.wc[i])*Rho_w*dz[i];
          SoilData.wc[i] := SoilData.fcwc[i];
        end
      else
        begin
          SoilData.wc[i] := SoilData.wc[i] + Precip.Precip/(Rho_w*dz[i]);
          Precip.Precip := 0;
        end;
      Inc(i);
    end; {while}
  if Precip.Precip > 0 then
    begin
      DrainDOY := Precip.Precip;
      Precip.Precip := 0;
    end; {if}
end; {procedure CalcInfiltration}

procedure CalcEvaporation;

var
  nwc,
  PotentialEvap: Real;

begin {procedure CalcEvaporation}
  EvapDOY := 0;        {!}
  PotentialEvap := (1-Crop.FIevap)*PET;
  if SoilData.wc[1] < SoilData.pwpwc[1] then
    PotentialEvap := PotentialEvap*Sqr((SoilData.wc[1]-Soil.Adwc)/(SoilData.pwpwc[1]-Soil.Adwc));
  nwc := SoilData.wc[1] - PotentialEvap/(Rho_w*dz[1]);
  if nwc < Soil.Adwc then
    nwc := Soil.Adwc;
  EvapDOY := (SoilData.wc[1] - nwc)*Rho_w*dz[1];
  SoilData.wc[1] := nwc;
end; {procedure CalcEvaporation}

procedure CalcTranspiration;

var
  i,m:      Integer;
  z,
  ust,
  est,
  estar,
  Psix,
  Loss,
  AvePstar,
  AvePsi:   Real;
  f:        SoilArray;

begin {procedure CalcTranspiration}
  TransDOY := 0;                {!}
  Soil.Spsi[1] := 0;
  Soil.ActualTrsp := 0;
  AvePsi := 0;
  z := dz[2];                   {z:=0;}
  i := 2;                       {rdd:=Crop.RD/4.6;}
```

```
  if (Crop.RD > 0) and (Crop.FItransp > 0) then
    begin
      repeat    {transpiration calculation}
        Soil.Spsi[i] := -Soil.a[i]*Exp(-Soil.b[i]*ln(SoilData.wc[i]));
        if z <= Crop.RD then
          f[i] := dz[i]*(2*(Crop.RD - z) + dz[i])/Sqr(Crop.RD)
        else
          f[i] := Sqr((Crop.RD - z + dz[i])/Crop.RD);      {f[i] := Exp(-z/rdd)*(1-Exp(-dz[i]/rdd));}
        AvePsi := AvePsi + f[i]*Soil.Spsi[i];
        Inc(i);
        if i <= NrOfLayers then
          z := z + dz[i];
      until (i > NrOfLayers) or (z - dz[i] > Crop.RD);
      AvePstar := AvePsi/CropData.Psilm;
      if AvePstar < 1.5 then
        ust := 1 - 0.67*AvePstar
      else
        ust := 0;
      est := PET/CropData.MaxTrans;  {if est < 0 then est := 0;}
      if est < ust then
        estar := est
      else
        estar := ust;
      if estar < 0 then
        estar := 0;
      Psix := CropData.Psilm*(AvePstar + 0.67*estar);
      m := i - 1;
      for i := 2 to m do
        begin
          Loss := (Crop.FItransp*CropData.MaxTrans*f[i]*(Psix - Soil.Spsi[i])/(0.67*CropData.Psilm))/(Rho_w*dz[i]);
          if SoilData.wc[i] - Loss < SoilData.pwpwc[i] then
            Loss := SoilData.wc[i] - SoilData.pwpwc[i];
          Soil.ActualTrsp := Soil.ActualTrsp + Loss*Rho_w*dz[i];
          SoilData.wc[i]  := SoilData.wc[i] - Loss;
        end; {for}
      TransDOY := Soil.ActualTrsp;
    end; {if}
  if Crop.FItransp > 0 then
    Soil.SI := Soil.ActualTrsp/(Crop.FItransp*PET)
  else
    Soil.SI := 0;
end; {procedure CalcTranspiration}

procedure SoilDayStep;

begin {procedure SoilDayStep}
  CalcEvaporation;
  CalcTranspiration;
  CalcInterception;
  CalcRunOff;
  CalcInfiltration;
end; {procedure SoilDayStep}

function SoilStoredWater(WC,dz: SoilArray;
                         RD:    Real): Real;
var
  z,
  StoredWater: Real;
  i:           Integer;
```

```
begin {function SoilStoredWater}
  i := 0;
  z := 0;
  StoredWater := 0;
  repeat
    Inc(i);
    StoredWater := StoredWater + WC[i]*dz[i]*Rho_w;
    z := z + dz[i];
  until (i >= NrOfLayers) or (z >= RD);
  SoilStoredWater := StoredWater;
end; {function SoilStoredWater}

function AllowableDepletionProc(FCWC,PWPWC,dz:    SoilArray;
                                RD:               Real;
                                PercentDepletion: Real): Real;
var
  z,
  Depletion: Real;
  i:         Integer;

begin {function AllowableDepletionProc}
  z := 0;
  i := 0;
  Depletion := 0;
  repeat
    Inc(i);
    Depletion := Depletion + (FCWC[i]-PWPWC[i])*dz[i]*Rho_w;
    z := z + dz[i];
    if z > RD then
      Depletion := Depletion - (z - RD)*(FCWC[i]-PWPWC[i])*Rho_w;
  until (i >= NrOfLayers) or (z >= RD);
  AllowableDepletionProc := -Depletion*PercentDepletion/100;
end; {function AllowableDepletionProc}

end. {unit _ISoil}
```

```
unit _IWeathr;

interface

uses
   Dos,
   PXEngine,
   PXMsg,
  _IPXEng,
  _Math,
  _InOut,
  _IGlob,
  _IPET,
  _FldTbl,
  _WIDTbl,
  _DayWTbl,
  _PrcpTbl,
  _IInpWRD;

type
  WeatherRecord = record
    Rni,                   {Net isothermal radiation W/m2}
    SVP,                   {saturation vapour pressure kPa}
    VP,                    {vapour pressure kPa}
    Eac,                   {Emissivity Atmosphere}
    Ea,                    {Clear sky emissivity}
    Ta,                    {Air temperature  C}
    Tw,                    {Wet bulb temperature  C}
    Latitude: Double;
  end; {WeatherRecord}

var
  PET,
  MeanTemp,
  Lni,                        {net isothermal long wave radiation}
  Slope:    Real;
  Weather:  WeatherRecord;

procedure InitWeather;
function PotSolar(DOY: Integer): Real;
function WeatherDayStep(WeatherID: Integer;
                        DOYDate:   DateStr;
                        DOY:       Integer): Integer;

implementation

uses
  _IInpPr;

var
  HalfDay: Real;              {half day length}

procedure InitWeather;

begin {procedure InitWeather}
  FillChar(Weather,SizeOf(Weather),#0);
end; {procedure InitWeather}

function PotSolar(DOY: Integer): Real;
```

```
var
  x,
  Lat,Dec,RelDist,HrAngle,
  SinDec,CosDec,Sinhs,Coshs,
  SinLatSinDec,CosLatCosDec: Real;

begin {function PotSolar}

  Lat      := -WeathrID.Latitude*Pi/180;          {Latitude in radians}
  Dec      := -0.4093*Sin(2*Pi*(284 + DOY)/365);  {Southern hemisphere}
  SinDec   := Sin(Dec);
  CosDec   := Cos(Dec);
  RelDist  := 1 + 0.033*Cos(2*Pi*DOY/365);
  HrAngle  := ArcCos(-Tan(Lat)*Tan(Dec));         {SunsetHourAngle}

  x        := -Sin(Lat)*SinDec/(Cos(Lat)*CosDec); {!}
  HalfDay  := Pi/2 - ArcTan(x/Sqrt(1-x*x));       {!}

  PotSolar := 118.08*RelDist/Pi*(HrAngle*Sin(Lat)*SinDec + HrAngle*Cos(Lat)*CosDec);
(*
  Lat          := WeathrID.Latitude*Pi/180;          {convert latitude to radians}
  SinDec       := 0.39785*Sin(4.869 + 0.0172*DOY + 0.03345*Sin(6.224 + 0.0172*DOY));
  CosDec       := Sqrt(1 - Sqr(SinDec));
  SinLatSinDec := Sin(Lat)*SinDec;
  CosLatCosDec := Cos(Lat)*CosDec;
  Coshs        := -SinLatSinDec/CosLatCosDec;
  Sinhs        := Sqrt(1 - Sqr(Coshs));
  HalfDay      := Pi/2 - ArcTan(Coshs/Sinhs);  {!}
  PotSolar     := 117.5*(HalfDay*SinLatSinDec + CosLatCosDec*Sinhs)/pi;
*)
end; {function PotSolar}

function WeatherDayStep(WeatherID: Integer;
                        DOYDate:   DateStr;
                        DOY:       Integer): Integer;
var
  k,
  PxErr: Integer;
  dt,
  Tr,            {atmospheric transmissivity}
  PSR:  Real;

begin {function WeatherDayStep}
  IrrigDOY  := 0;          {!}
  PrecipDOY := 0;          {!}
  PxErr := FindWeather(WeatherID,DOYDate);
  if PxErr = PXERR_RECNOTFOUND then
    begin
      InputWeatherProc(WeatherID,DOYDate);
      PxErr := FindWeather(WeatherID,DOYDate);
    end; {if}
  if PxErr = PxSuccess then
    begin
      PxErr := FindPrecip(Field.PerseelID,DOYDate);
      if PxErr = PXERR_RECNOTFOUND then
        begin
          InputPrecipProc(Field.PerseelID,DOYDate);
          PxErr := FindPrecip(Field.PerseelID,DOYDate);
```

```
        end; {if}
      end; {if}
    if PxErr = PxSuccess then
      begin
        IrrigDOY  := Precip.Irrig;
        PrecipDOY := Precip.Precip;
        Precip.Precip := Precip.Precip + Precip.Irrig;   {!}
        PSR := PotSolar(DOY);
        dt := WeatherData.MaxTemp - WeatherData.MinTemp;
        if dt < 2 then
          dt := 2;
        k := DOY - 30;
        if k < 1 then
          k := k + 365;
        if WeatherData.Solar = 0 then
          begin
            Tr := 0.7*(1 - Exp(-0.329*Sqr(dt)/PotSolar(k)));
            WeatherData.Solar := PSR*Tr;
          end
        else
          Tr := WeatherData.Solar/PotSolar(k);
        MeanTemp := (WeatherData.MaxTemp + WeatherData.MinTemp)/2;
        Slope := ((0.00223*MeanTemp + 0.0549)*MeanTemp + 2.97)*MeanTemp + 45.3;     {gives Slope in Pa}
        Lni   := (HalfDay/pi)*(0.96 - 1/(1 + 0.048*Exp(7.1*Tr)))*(0.026*MeanTemp - 9.2);

        if WeatherData.VPD = 0 then
          WeatherData.VPD := 0.7*Slope*dt
        else
          WeatherData.VPD := WeatherData.VPD*1000;    {to convert kPa to Pa}
      end; {if}
    if PxErr = PxSuccess then
      if WeatherData.PET = 0 then
        PET := CalcPriestlyPET
      else
        PET := WeatherData.PET;
    WeatherDayStep := PxErr;
  end; {function WeatherDayStep}

  end. {unit _IWeathr}
```

```
unit _IPET;

interface

uses
   Crt,
   PXEngine,
   PXMsg,
  _IPXEng,
  _InOut,
  _IGlob,
  _DayWTbl,
  _HrWTbl,
  _FldTbl;

function CalcPriestlyPET: Real;

implementation

uses
  _IWeathr;

function CalcPriestlyPET: Real;   {potential ET in mm or kg/m2/day}

var
  Rni: Real;

begin {function CalcPriestlyPET}
  Rni := 0.8*WeatherData.Solar - Lni;
  CalcPriestlyPET := 0.9*((1 + 0.0003*WeatherData.VPD)*Slope*Rni/(66 + Slope))/2.43; {G=0.1*Rni}
end; {function CalcPriestlyPET}

end. {unit _IPET}
```