ภาคผนวกที่ 1 รหัสคำสั่ง (source code) โปรแกรมแบบจำลองข้าวที่เรียบง่ายสำหรับสภาพแวดล้อมทุ่งกุลาร้องไห้

Definition of type

Symbol	Type
V	common variable
P	Input parameter
I	input system initial state
E	environmental driving variable
C	constant
L	local variable
O	output variable
F	function
S	subroutine

Model components

Module inter_vars
Sharing common variables

Name	Description	unit	Type	Origin	Used by
p	phenological state	-	V	phenology	growth
pj	phenological state at the end of juvenile	-	V	phenology	
pi	phenological state at the end of induction	-	V	phenology	
pl	phenological state at the end of leaf development	-	V	phenology	growth
pp	phenological state the end of panicle development		-	V	phenology
	growth/rlq				
pg	phenological state at the end of grain development		-	V	phenology
	rlq				
ws	shoot biomass	$g\ m^{-2}$	V	growth	weeds
l	leaf area index	$m^2\ m^{-2}$	V	growth	growth/ET
qrel	relative available soil water	-	V	su_qrel	growth/ET
lw	leaf area index of weeds	$m^2\ m^{-2}$	V	weeds	growth

```
module inter_vars
real :: pj, pi, pl, pp, pg, p
real :: ws, l
real :: qrel
real :: lw
end
```

Subroutine Model
Root subroutine of model components

Name	Description	unit	Type	Origin	Used by
evapotranspiration	calculate actual evapotranspiration		S	User	
phenology	calculates crop development		S	User	
growth	calculates leaf area, biomass of shoot and panicles		S	User	
weeds	calculate weeds leaf area		S	User	

```
subroutine model
implicit none
    call evapotranspiration
    call phenology
    call growth
    call weeds
end
```

Subroutine Phenology
Calculation of crop development

Name	Description	unit	Type	Origin	Used by
dj	duration of basic vegetation (juvenile) phase	d	P	rnspec sys	phenology
di	duration of photosensitive (induction) phase	d	P	rnspec sys	phenology
dp	duration of panicle development	d	P	rnspec sys	phenology
dg	duration of grain development	d	P	rnspec sys	phenology
tb	base temperature for development	C	P	rnspce.sys	phenology
top	optimal temperature for development	C	P	rnspec sys	phenology
po	optimal photoperiod.	h	P	rnspec sys	phenology
pc	critical photoperiod.	h	P	rnspec sys	phenology
la	lattitude	degree	P	rnspec sys	asdl
tmn	daily minimum temperature.	C	E	rnspec.wth	phenology
tmx	daily maximum temperature.	C	E	rnspec.wth	phenology
dc	crop duration under optimal conditions	d	L		
n	coefficient of temperature effect on development	-	L		
ta	daily average temperature	C	L		
u	different between ta and tb	C	L		
uop	different between top and tb	C	L		
ftop	developmental rate at optimal temperature	d^{-1}	L		
ft	function of temperature on rate of development	d^{-1}	L		
fp	function of photoperiod on rate of development	-	L		
dp	rate of phenolgical development	d^{-1}	L		
p	phenological state	-	I/O/V	phenology	growth
pj	phenological state at the end of juvenile	-	O/V	phenology	
pi	phenological state at the end of induction	-	O/V	phenology	
pl	phenological state at the end of leaf development	-	O/V	phenology	growth
pp	phenological state the end of panicle development growth		-	O/V	phenology
pg	phenological state at the end of grain development		-	O/V	phenology
asdl	calculate astronomical daylength	h	F	user	
param	read input parameter from rnspec.sys		F	driver	
incon	read input initial condition from rnspec.sys		F	driver	
step	controlling simulation step		F	driver	
intgrl	rectangular integration		F	driver	
dwthr	read daily weather variables from rnspce.wth		S	driver	
output	output variables to output.dat		S	driver	
termination	terminate simulation run		S	driver	

```
subroutine phenology
use inter_vars
implicit none
real :: dj, di, dp, dg, dc, ftop
real :: tb, top, tx, ta, n, uop, u, ft
real :: pc, po, fp
real :: asdl, la
real :: step, param, incon, intgrl
real :: sr, tmn, tmx, rh, u2, rf
call dwthr(sr, tmn, tmx, rh, u2, rf)

procedure: select case (int(step()))
case (1)
    dj  = param ('dj')
    di  = param ('di')
    dp  = param ('dp')
    dg  = param ('dg')

    tb  = param ('tb')
    top = param('top')

    n = 4.0
    uop = top-tb

    pc  = param ('pc')
```

```
    po  = param ('po')
    la  = param ('la')
    p   = incon ('p')

    dc  = dj+di+dp+dg
    pj  = dj/dc
    pi  = (dj+di)/dc
    pp  = (dj+di+dp)/dc
    pl  = pi+0.75*(pp-pi)
    pg  = (dj+di+dp+dg)/dc
    ftop= 1.0/dc

case (2)
    ta = (tmx+tmn)/2.0
    u  = ta-tb
    ft = ftop * ((n/2.0)*uop**(n-2.0)*u**2.0-u**n)/(uop**n*(n/2.0-1.0))
    if (p < pj) then
        dp = ft
    else if (p < pi) then
        fp = exp(-max(0.0,(asdl(la)-po))/(pc-po))
        dp = ft*fp
    else if (p < pp) then
        dp = ft
    else
        dp = ft
    end if

case (3)
    p = intgrl (p, dp)
end select procedure

call output ('p', p)
if(p >= pg) call termination ('maturation')
end
```

subroutine Growth
Calculate shoot and panicle biomass, leaf area

Name	Description	unit	Type	Origin	Used by
ec	solar radiation use efficiency	g DW MJ^{-1}	P	rnspec.sys	growth
kc	canopy light attenuation coefficient	-	P	rnspec.sys	growth
as	partition coefficient for shoot	g g^{-1}	P	rnspec.sys	growth
bs	partition (exponent) coefficient for shoot	-	P	rnspec.sys	growth
al	partiton coefficient for leaf area	m^2 g^{-1}	P	rnspec.sys	growth/weeds
bl	partition (exponent) coefficient for lear area	-	P	rnspec.sys	growth/weeds
lc	leaf loss coefficient	m^2 g^{-1}	P	rnspec.sys	growth
sr	daily solar radiation	MJ m^2 d^{-1}	E	rnspec.wthr	growth
p	phenological state of crop development	-	V	phenology	growth
pl	phonological state at the end of leaf development	-	V	phenology	growth
pp	phenological state the end of panicle development growth		-	V	phenology
qrel	relative available soil water	-	V	su_qrel	growth/ET
gcc	crop growth rate, potential	g m^{-2} d^{-1}	L		
gca	crop growth rate, actual	g m^{-2} d^{-1}	L		
wc	dry weight of crop	g m^{-2}	I/O	rnspec.sys/growth	
ws	dry weight of shoot	g m^{-2}	O/V	growth	weeds
wp	dry weight of panicles	g m^{-2}	I/O	growth	
l	leaf area index	m^2 m^{-2}	O/V	growth	
param	read input parameter from rnspec.sys		F	driver	
incon	read input initial condition from rnspec.sys		F	driver	
step	controlling simulation step		F	driver	
intgrl	rectangular integration		F	driver	

dwthr	read daily weather variables from rnspce.wth	S	driver
output	output variables to output.dat	S	driver

```
subroutine growth
use inter_vars
implicit none
real :: ec, kc, as, bs, ah, bh, al, bl, lc
real :: gcc, gca
real :: wc, wp
real :: step, param, incon, intgrl
real :: sr, tmn, tmx, rh, u2, rf
call dwthr(sr, tmn, tmx, rh, u2, rf)

procedure: select case (int(step()))
case (1)
    ec  =   param ('ec')
    kc  =   param ('kc')
    as  =   param ('as')
    bs  =   param ('bs')
    al  =   param ('al')
    bl  =   param ('bl')
    lc  =   param ('lc')
    wc  =   incon ('wc')
    wp  =   incon ('wp')
    ws  =   as*wc**bs
    l   =   al*ws**bl

case (2)
    gcc =   ec * (1.0-exp(-kc*l))* exp(-1.0*lw)* sr
    gca =   qrel * gcc
case (3)
    wc  =   intgrl (wc, gca)
    if (p <= pl) then
        ws  =   as*wc**bs
        l   =   al*ws**bl
    else if (p <= pp) then
        ws  =   as*wc**bs
    else
        wp  =   intgrl(wp, gca)
        l   =   max(0.0, (l - lc*gca))
   end if
end select procedure

call output ('l', l)
call output ('wp', wp)
call output ('ws', ws)
call output ('wc', wc)
end
```

Subroutine evapotranspiration
Calculate the actual rate of evapotranspiration, abbreviated to ET, using linear equations

Name	Description	unit	Type	Origin	Used by
lh	latent heat of evaporation of water 2.45	MJ kg^{-1}	C		
sr	solar radiation	MJ m^{-2} d^{-1}	E	rnspec.wth	growth/ET
qrel	relative availability of soil water	-	V	su_qrel	growth/ET
rn	net radiation	MJ m^{-2} d^{-1}	L		
et	actual evapotranspiration	mm d^{-1}	O	ET	
su_qrel	calculate relative available soil water		S	User	

step	controlling simulation step	F	Driver
dwthr	read daily weather variables from rnspce.wth	S	Driver
output	output variables to output.dat	S	Driver

```
subroutine evapotranspiration
use inter_vars
implicit none
real :: rn, et
real :: step
real, parameter :: lh=2.45
real :: sr, tmn, tmx, rh, u2, rf
call dwthr(sr, tmn, tmx, rh, u2, rf)
call su_qrel

procedure: select case (int(step()))
case (2)
    rn  = 0.7 * sr - 1.0
    et  = qrel*(0.8*rn + 2.0)/lh
end select procedure
call output ('et', et)
end
```

Subroutine su_qrel
Calculate relative availability of soil water

Name	Description	unit	Type	Origin	Used by
qs	soil water content at saturation	$m^3\ m^{-3}$	P	rnspec.sys	su_qrel
qr	soil water content at air dry	$m^3\ m^{-3}$	P	rnspec.sys	su_qrel
j	day of year	-	L		
q	current soil water content	$m^3\ m^{-3}$	L		
qrel	relative available soil water	-	O/V	su_qrel	growth/ET
param	read input parameter from rnspec.sys		F	driver	
step	controlling simulation step		F	driver	
doy	return current day of year		F	driver	
intpl	linear interpolation		F	driver	

```
subroutine su_qrel
use inter_vars
implicit none
real :: qs, qr, q
real :: step, param, intpl
real :: doy, j

procedure: select case (int(step()))
case (1)
    qs = param ('qs')
    qr = param ('qr')
case (2)
    j  = doy()
    q  = min(qs, intpl ('vwc', j))
    qrel = (q-qr)/(qs-qr)
end select procedure
end
```

Subroutine weeds
Calculate leaf area index of weeds

Name	Description	unit	Type	Origin	Used by
al	partiton coefficient for weeds leaf area	$m^2 g^{-1}$	P	rnspec.sys	growth/weeds
bl	partition coefficient (exponent) for weeds leaf area growth/weeds		-	P	rnspec.sys
few	fraction of weeds by weight	-	P	rnspec.sys	weeds
ws	rice shoot biomass	g m-2	V	growth	growth/weeds
j	day of year	-	L		
wsw	weeds shoot biomass	g m-2	L		
lw	weeds leaf area index	$m^2 m^{-2}$	O/V	weedsl	growth
param	read input parameter from rnspec.sys		F	driver	
step	controlling simulation step		F	driver	
doy	return current day of year		F	driver	
intpl	linear interpolation		F	driver	

```
subroutine weeds
use inter_vars
implicit none
real :: al, bl
real :: wsw
real :: step, param, intpl, doy
real :: j
j = doy()

procedure: select case (int(step()))
case (1)
    al  =   1.2 * param ('al')
    bl  =   param ('bl')
    wsw =   ws * intpl('fwe',j)
    lw  =   al*wsw**bl
case (2)
case (3)
    wsw =   ws * intpl('fwe',j)
    lw  =   al*wsw**bl
end select procedure

call output ('lw', lw)
end
```

Function asdl
Calculate astronomical daylength
syntax: dl (la)

Name	Description	unit	Type	Origin	Used by
la	latitude	degree	P	rnspec.sys	asdl
j	day of year	doy	L		
asdl	astronomical daylength	h decimal	O	asdl function	
pi	return the value of pi		F	Driver	
doy	return current day of year		F	Driver	

```
real function asdl(la)
implicit none
real :: la, j
real :: pi, l, d
real :: doy
j=doy()

    l    = pi()/180.0*la
    d    = 0.409*sin(2.0*pi()/365.0*j-1.39)
    asdl = 2.0 * 12.0/pi() * acos(-tan(l)*tan(d))
end
```

Driver's functions and subroutines used by the model

Function param
Passing the value of variable from rnspec.sys to the calling routine
Syntax: param ('variableName')
Note: case sensitive

Function incon
Passing the value of system initial state from rnspec.sys to the calling routine
Syntax: incon ('variableName')
Note: case sensitive

Function step
Passing value to synchronise calculation among modules
Syntax: step ()
Note: void function

Function intgrl
Perform rectangular integration
Syntax: intgrl(sate variable, rate variable)

Function intpl
Perform linear Interpolation using co-ordinates given in rnspec.sys
Syntax: intpl ('co-ordinate table name', x value)
Note: case sensitive

Function doy
Return the current day of year
syntax: doy ()
Note: void function

Function pi
Return the value of pi=3.1415926535897932
syntax: pi ()
Note: void function

Subroutine dwthr
Passing 6 daily weather variables from rnspec.wth to the calling routine
Syntax: dwthr (sr, tmn, tmx, rh, u2, rain)

sr	daily solar radiation.	MJ m^{-2} d^{-1}	E	rnspec.wth
tmn	daily minimum temperature.	C	E	rnspec.wth
tmx	daily maximum temperature.	C	E	rnspec.wth
rh	daily relative humidity	%	E	rnspec.wth
u2	daily wind speed at 2m height.	m s^{-1}	E	rnspec.wth
rain	daily rainfall.	mm	E	rnspec.wth

Subroutine output
Output value of specified variable to output.dat at a given interval specified by OUTDEL
Syntax: output('header', variable name)

Subroutine termination
Terminate simulation run and output user specified text to the end of output.dat
Syntax: termination('specified text')

ภาคผนวกที่ 2 รหัสคำสั่ง (source code) ชุดคำสั่งกลาง (driver) ควบคุมขั้นตอนการประมวลผล

```
MODULE Bnd_con
REAL, DIMENSION(366,10):: cmt
INTEGER:: iday
END

MODULE Sys_spc
CHARACTER (LEN=10), DIMENSION(250):: name
REAL, DIMENSION (2700):: value
INTEGER, DIMENSION(0:251):: lvp
END

MODULE Ctrl_Time
CHARACTER (LEN=40):: finmsg='Not specified '
REAL:: stage, date, time, delt=1.0
INTEGER:: fincon
END

MODULE Id_Lst
CHARACTER (LEN=66), PARAMETER :: lecha= &
'AaBbCcDdEeFfGgHhIiJjKkLlMmNnOoPpQqRrSsTtUuVvWwXxYyZz=+-.0123456789'
INTEGER :: lnmx = 132
END

MODULE Msg_Lst
IMPLICIT none
CHARACTER (LEN=17), DIMENSION (0:11):: Minfo  = &
         (/ 'RNSPEC.WTH',            'RNSPEC.SYS',    &
            ': Table not found',': CHK coordinates', &
            ': Order of X axis',': INTPL off limit', &
            ': Array not found',': Input < Request', &
            ': Param not found',': Param > 1 entry', &
            'file not found',    ' '        /)
END

PROGRAM DRIVER    ! Elementary Simulation Driver
USE Bnd_con
USE Sys_spc
USE Ctrl_Time
IMPLICIT NONE

OPEN (6, CARRIAGECONTROL='FORTRAN')
WRITE (6, '(''0'',3X, A)') 'Initialization'
fincon=0
CALL Inize
CALL Rd_Bnd
CALL Rd_Sys
stage = 1.0; CALL T_Step; CALL Model
   DO
     stage = 2.0; CALL Model
     IF (fincon /= 0) EXIT
     stage = 3.0; CALL T_Step; CALL Model
   END DO
stage = 4.0; CALL Model

CALL Logmsg(' ',11, 0, 0, 'FINAL')
WRITE (6, '(4X, A)') 'Termination'
END
```

```
SUBROUTINE T_Step
USE Bnd_con
USE Ctrl_time
IMPLICIT NONE
REAL :: nday, sdate, param, t = 0.0

IF (t==0.0) THEN
   nday  = REAL(iday)
   sdate = Param('SDATE')
   time  = 0.0
   t     = 1.0
   date  = MOD((sdate+time-1.0), nday) + 1.0
   WRITE (6,'(4X,A,5X,A,F10.2)')'Simulation:','TIME', time
ELSE
   time = time + delt
   date = MOD((sdate+time-1.0), nday) + 1.0
   WRITE (6,'(''+'',3x,A,5X,A,F10.2)')'Simulation:','TIME', time
END IF
IF (time>=nday) fincon=10
END

SUBROUTINE Output (vname, vval)
USE Ctrl_Time
IMPLICIT NONE
CHARACTER (LEN=*) :: vname
REAL :: vval

CHARACTER (LEN=7), DIMENSION(31) :: names
CHARACTER (LEN=4) :: cnos
CHARACTER (LEN=40) :: edlstn, edlstv
REAL, DIMENSION(31) :: vals
REAL :: ostage, param, outdel
INTEGER ::n, icmx,  ic=1

IF (stage == 1.0) THEN
   ic        = ic + 1
   icmx      = ic
   names(ic)  = vname
ELSE IF (stage == 2.0) THEN
   IF (ostage == 1.0) THEN
      CALL Onfile(30, 'OUTPUT.DAT', 'SEQUENTIAL', 'FORMATTED')
      WRITE (cnos, '(I4)') icmx
      edlstn  = '(3X,'//cnos//'(A6,6X))'
      edlstv  = '('//cnos//'(1X,G11.5))'
      names(1)= 'TIME'
      WRITE (30, edlstn)(names(n), n = 1, icmx)
      outdel = Param('OUTDEL')
   END IF
   IF (ic==icmx) ic = 1
   ic       = ic + 1
   vals(ic) = vval
   IF (MOD(time,outdel)==0.0 .AND. ic==icmx) THEN
      vals(1) = time
      WRITE (30, edlstv)(vals(n), n = 1, icmx)
      ic      = 1
   ENDIF
ELSE IF (stage==4.0) THEN
   IF (ic>1) THEN
      vals(1) = time
```

```
      WRITE (30, edlstv)(vals(n), n = 1, icmx)
      ic = 1
   ENDIF
   IF (ostage/=4.0) WRITE(30,'(/,''  Termination: '',A)') finmsg
ENDIF
ostage = stage
END

FUNCTION Step
USE Ctrl_time
IMPLICIT NONE
REAL :: step
step = stage
END

SUBROUTINE Termination (finterm)
USE Ctrl_time
IMPLICIT NONE
CHARACTER (LEN=*) ::  finterm
finmsg = finterm
fincon = 10
END

FUNCTION DOY
USE Ctrl_time
IMPLICIT NONE
REAL :: doy
doy = date
END

SUBROUTINE Timer (rntime)
USE Ctrl_time
IMPLICIT REAL (A-Z)
rntime = time
END

SUBROUTINE Dwthr (v1, v2, v3, v4, v5, v6)
USE Bnd_con
USE Ctrl_time
IMPLICIT NONE
REAL, INTENT(out) :: v1, v2, v3, v4, v5, v6
SAVE

IF (stage==2.0) THEN
  v1    = cmt(INT(date),1)
  v2    = cmt(INT(date),2)
  v3    = cmt(INT(date),3)
  v4    = cmt(INT(date),4)
  v5    = cmt(INT(date),5)
  v6    = cmt(INT(date),6)
ENDIF
END

FUNCTION Pi()
IMPLICIT NONE
REAL :: pi
     pi=3.1415926535897932
END
```

```
FUNCTION Incon (rqname)
IMPLICIT NONE
CHARACTER (LEN=*) :: rqname
REAL :: incon, param
Incon = Param (rqname)
END

FUNCTION Param (rqname)
USE Sys_spc
USE Ctrl_time
IMPLICIT NONE
CHARACTER (LEN=*) :: rqname
REAL :: param

CHARACTER (LEN=10), DIMENSION(250):: locrec
INTEGER, DIMENSION(251) :: recid = 0
INTEGER ::  i, j

DO i = 1, 251
    IF (rqname==locrec(i) .OR. recid(i)==0) EXIT
END DO

IF (recid(i)==0) THEN
    locrec(i) = rqname;  recid(i) = -8
    DO j = 1, 251
      IF (rqname==name(j)) THEN
        IF (lvp(j)-lvp(j-1)==1) THEN
          recid(i) = j
          Param = value(lvp(recid(i)))
        ELSE
          recid(i) = -9
        ENDIF
      ENDIF
      IF (lvp(j)<0 .OR. recid(i)/=-8) exit
    END DO
    IF (recid(i)<0) THEN
      fincon = -recid(i)
      CALL Logmsg (rqname, fincon,-1,-1,'OPENS')
    END IF
ELSE IF (recid(i)>0) THEN
    Param = value(lvp(recid(i)))
ELSE
    END IF
END


FUNCTION ParamS (rqname, indx)
USE Sys_spc
USE Ctrl_time
IMPLICIT NONE
CHARACTER (LEN=*) :: rqname
REAL :: params
INTEGER :: indx

CHARACTER (LEN=10), DIMENSION(250):: locrec
INTEGER,  DIMENSION (251) :: recid
INTEGER :: i, j, n

DO i = 1, 251
   IF (rqname==locrec(i) .OR. recid(i)==0) EXIT
END DO
```

```
IF (recid(i)==0) THEN
   locrec(i) = rqname; recid(i) = -6
   DO j = 1, 251
     IF (rqname==name(j)) THEN
       recid(i)= j
       n = lvp(j)-lvp(j-1)
     END IF
     IF (lvp(j)<0 .OR. recid(i)/=-6) EXIT
   END DO
   IF (recid(i)<0) THEN
     fincon = -recid(i)
     CALL Logmsg (rqname, fincon,-1,-1,'OPENS')
   ENDIF
END IF

IF (recid(i)>0) THEN
  IF (indx<=n) THEN
     Params = value(lvp(recid(i)-1)+indx)
  ELSE
     recid(i)= -7
     fincon  = -recid(i)
     CALL Logmsg (rqname, fincon,-1,-1,'OPENS')
  ENDIF
ELSE
  ENDIF
END

FUNCTION Intpl(rqname, x)
USE Sys_spc
USE Ctrl_time
USE Msg_Lst
IMPLICIT NONE
CHARACTER (LEN=*) :: rqname
REAL :: intpl, x

CHARACTER (LEN=10), DIMENSION(250):: locrec(250)
REAL :: mnx, mxx
INTEGER, DIMENSION(251) :: recid
INTEGER :: i, j, ix

DO i = 1, 251
   IF (rqname==locrec(i) .OR. recid(i)==0) EXIT
END DO

IF (recid(i)==0) THEN
   locrec(i) = rqname; recid(i) = -2
   DO j = 1,251
       IF (rqname==name(j)) THEN
         recid(i) = j
         IF (MOD((lvp(j)-lvp(j-1)+3),2)<1) THEN
             recid(i) = -3
         ELSE
             ix = lvp(j-1) + 3
             DO
               IF ((value(ix)-value(ix-2))<=0) recid(i) = -4
               ix = ix + 2
               IF ((recid(i)<0).OR.(ix>lvp(j))) EXIT
             END DO
         ENDIF
       END IF
```

```
        IF (lvp(j)<0 .OR. recid(i)/=-2) EXIT
    END DO
    IF (recid(i)<0) THEN
        fincon = -recid(i); finmsg = Minfo(fincon)//', '//rqname
        CALL Logmsg (rqname, fincon,-1,-1,'OPENS')
    ENDIF
END IF

IF (recid(i)>0) THEN
   mnx = value(lvp(recid(i)-1)+1)
   mxx = value(lvp(recid(i))-1)
   IF (x<mnx.OR.x>mxx) THEN
        recid(i) = -5
        fincon   = 5
        CALL Logmsg (rqname, fincon,-1,-1,'OPENS')
        finmsg = Minfo(fincon)//', '//rqname
   ELSE
        ix = lvp(recid(i)-1) + 3
        DO
          IF ((x<value(ix)) .OR. (ix>=lvp(recid(i))-1)) EXIT
          ix  = ix + 2
        END DO
        Intpl=value(ix-1)+(x-value(ix-2))*(value(i+1)- &
             value(ix-1))/(value(ix)-value(ix-2))
   ENDIF
ENDIF
END


FUNCTION Intgrl(statev, ratev)
USE Ctrl_Time
IMPLICIT NONE
REAL :: intgrl, statev, ratev
Intgrl = statev + ratev * delt
END

SUBROUTINE Inize
USE Msg_Lst
IMPLICIT NONE
CHARACTER (LEN=133), ALLOCATABLE, DIMENSION(:) :: line
INTEGER, ALLOCATABLE, DIMENSION(:,:,:) :: itpos
INTEGER :: ilmx, itmx(366)

INTEGER :: error, i, id
INTEGER, DIMENSION(26) :: wtinfo=(/ (2,i=1,3), (3,i=4,26) /), icinfo=1
LOGICAL :: Finds

ALLOCATE (line(366),itpos(2, 26, 366), STAT=error)
IF (error/=0) STOP '  Unable to ALLOCATE buffer'

CALL Chkofile
id = 0
IF (Finds(Minfo(id))) THEN
   CALL Locate(id, ilmx, itmx,  itpos, line)
   CALL Scanin(id, ilmx, itmx, itpos, line, wtinfo)
   CALL Wrwthr(ilmx, itmx, itpos, line, wtinfo)
ELSE
   CALL Logmsg(Minfo(id),10, -1, -1,'OPENS')
ENDIF
id = 1
```

```
IF (Finds(Minfo(id))) THEN
   CALL Locate(id, ilmx, itmx,  itpos, line)
   CALL Scanin(id, ilmx, itmx, itpos, line, icinfo)
   CALL Wrincn(ilmx, itmx, itpos, line)
ELSE
   CALL Logmsg(Minfo(id),10, -1, -1,'OPENS')
ENDIF

DEALLOCATE (line,itpos, STAT=error)
IF (error/=0) STOP '  Unable to DEALLOCATE buffer'

IF (Finds('MSG.LOG')) THEN
   CALL Logmsg('Inize',11, 0, 0, 'FINAL')
   IF (Finds('WTH.BIN '))   CALL Deletes('WTH.BIN ')
   IF (Finds('SYS.BIN ')) CALL Deletes('SYS.BIN ')
   IF (Finds('AINFO.BIN'))  CALL Deletes('AINFO.BIN')
   STOP '   Termination'
END IF
END


SUBROUTINE chkofile
IMPLICIT NONE
LOGICAL :: finds

IF (Finds('MSG.LOG')) CALL Deletes('MSG.LOG')
IF (Finds('RUN.LOG')) CALL Deletes('RUN.LOG')
IF (Finds('OUTPUT.DAT')) CALL Deletes('OUTPUT.DAT')
END

SUBROUTINE Wrwthr (ilmx, itmx, itpos, line, info)
IMPLICIT NONE
CHARACTER (LEN=*), DIMENSION (366) :: line
INTEGER :: ilmx
INTEGER, DIMENSION(366) :: itmx
INTEGER, DIMENSION(2,26,366) :: itpos
INTEGER, DIMENSION(20) :: info

REAL :: rdata
INTEGER :: il, it, sum, mean, error, ival
INTEGER, DIMENSION(3) :: fp, lp
INTEGER, PARAMETER:: untt=3
LOGICAL :: Finds

sum   = 0
DO il = 1, ilmx
   sum = sum + itmx(il)
END DO
mean  =  nint (REAL(sum)/REAL(ilmx))
error = 0
DO il = 1, ilmx
   IF (itmx(il)/=mean) THEN
      CALL Logmsg('Unequal Entry',0,1000, IL,'OPENS')
      error = 1
   END IF
END DO
CALL Fedsp (mean, info, fp, lp)
ival = lp(3) - fp(3) + 1
IF (ival>10) CALL Logmsg('>10 Fields',11,-1,-1,'OPENS')
IF (ival>10 .OR. error>0) THEN
```

```
   CALL Logmsg('TRSCWT',11, 0, 0, 'CLOSES')
   RETURN
END IF

IF (Finds ('MSG.LOG')) RETURN
CALL Onfile (untt, 'WTH.BIN ', 'SEQUENTIAL', 'BINARY')
WRITE (untt) ilmx
WRITE (untt) ival
DO il = 1, ilmx
   DO it = fp(3), lp(3)
      READ (line(il)(itpos(1,it,il):itpos(2,it,il)),'(G12.7)') rdata
      WRITE (untt)rdata
   END DO
END DO
CLOSE (untt)
END

SUBROUTINE Fedsp (itmx, info, fp, lp)
IMPLICIT NONE
INTEGER itmx
INTEGER, DIMENSION(20) :: info
INTEGER, DIMENSION(3) :: fp, lp

INTEGER :: linfo, j

fp = 1
lp = 0
linfo = 0
DO j = 1, itmx
   IF(info(j)/=linfo) THEN
      fp(info(j)) = j
      lp(info(j)) = j
      linfo = info(j)
   ELSE
      lp(info(j)) = j
   ENDIF
END DO
END

SUBROUTINE Locate (id, ilmx, itmx, itpos, line)
USE Msg_Lst
IMPLICIT NONE
CHARACTER (LEN=*), DIMENSION(366) :: line
INTEGER :: id, ilmx
INTEGER, DIMENSION(366) :: itmx
INTEGER, DIMENSION(2,26,366) :: itpos

INTEGER :: il, al, iochk
INTEGER, PARAMETER:: untr=20
LOGICAL :: datum
SAVE

OPEN(untr, FILE=Minfo(id), STATUS='OLD')
al = 1
il = 1
ilmx  = 0
DO
   CALL Aline (il, al, 'IN')
   READ (untr, '(A)', IOSTAT=iochk) line(il)
   IF (iochk /= 0) EXIT
```

```
   CALL lineid (id, il, line(il), datum)
   IF (datum) THEN
      CALL Posid (il, line(il), itmx(il), itpos)
   ELSE
      il = il-1
   ENDIF
   ilmx = il
   IF (il >= 366) EXIT
   al = al+1
   il = il+1
END DO
CLOSE(untr)
END

SUBROUTINE Aline (il, al, sw)
IMPLICIT NONE
CHARACTER (LEN=*) :: sw
INTEGER :: il, al
INTEGER, DIMENSION(366) :: lines

IF (sw=='IN') lines(il) = al
IF (sw=='OUT') al = lines(il)
END

SUBROUTINE Lineid (id, il, lcline, datum)
USE Id_Lst
IMPLICIT NONE
CHARACTER (LEN=*) :: lcline
INTEGER :: id, il
LOGICAL :: datum

INTEGER ::ip, col
CHARACTER (11) :: msg='Columns>132'

datum = .false.
col   = LEN_TRIM(lcline)
DO ip = 1, col
     IF (INDEX(lecha, lcline(ip:ip))/=0) THEN
        datum = .true.
        IF (col>lnmx) CALL Logmsg(msg, id, col, il,'OPENS') EXIT
     ELSE IF (lcline(ip:ip)=='!') THEN
        EXIT
     END IF
END DO
END

SUBROUTINE Posid (il, lcline, it, itpos)
USE Id_Lst
IMPLICIT NONE
CHARACTER (LEN=*) :: lcline
INTEGER :: il, it
INTEGER, DIMENSION(2,26,366) :: itpos

INTEGER :: ip, ipmx
SAVE

IF (INDEX(lcline, '!')<=0) THEN
     ipmx = MIN(LEN_TRIM(lcline), lnmx)
ELSE
     ipmx = MIN(INDEX(lcline, '!')-1, lnmx)
```

```
END IF
it = 0
ip = 1
DO
  IF (INDEX(lecha,lcline(ip:ip))/=0 ) THEN
     it = it + 1
     itpos(1,it,il) = ip
     DO
        IF (INDEX(lecha,lcline(ip:ip))==0) THEN
          itpos(2,it,il) = ip - 1
          EXIT
        END IF
        ip =ip + 1
     END DO
  END IF
  ip = ip + 1
  IF (ip>ipmx) EXIT
END DO
END

SUBROUTINE Logmsg (msg, id, pn, il, order)
USE Msg_Lst
IMPLICIT NONE
INTEGER :: pn, il, id
CHARACTER (LEN=*) :: msg, order

INTEGER :: al
INTEGER,PARAMETER:: untm=90
LOGICAL :: fexist, fopen
SAVE

INQUIRE (FILE='MSG.LOG', EXIST=fexist, OPENED=fopen)
IF (order=='OPENS') THEN
   OPEN (untm, FILE='MSG.LOG', ACCESS='APPEND', STATUS='UNKNOWN')
   IF (il>0 .OR. pn>0) THEN
      CALL Aline(il, al, 'OUT')
      WRITE (untm, '(1X, 2A, T33, A, T44, A, I3, A, I3)') &
      'Message: ',msg, Minfo(id), 'Line: ',al,'  Column:',pn
   ELSE
      WRITE (untm, '(1X, 2A,T24, A)') 'Message: ',msg, Minfo(id)
   END IF
   CLOSE (untm)
ELSE
   IF (fexist .AND. order=='CLOSES') THEN
      IF (fopen) CLOSE (untm)
   ELSE IF (fexist .AND. order=='FINAL') THEN
      OPEN (untm, FILE='MSG.LOG', ACCESS='APPEND', STATUS='UNKNOWN')
      WRITE (untm,'(/, 1X, A)') 'End of message'
      WRITE (6, '( 4X, A)') 'MESSAGE: Message in MSG.LOG'
      CLOSE(UNTM)
   ENDIF
END IF
END

SUBROUTINE Onfile(unt, fname, faccs, fform)
IMPLICIT NONE
INTEGER :: unt
CHARACTER (LEN=*):: fname, faccs, fform
LOGICAL :: Finds
```

```
IF (Finds(fname)) CALL Deletes(fname)
OPEN(unt, FILE=fname, ACCESS=faccs, STATUS='NEW', FORM=fform)
END


SUBROUTINE Rd_Sys
USE Sys_spc
IMPLICIT NONE
INTEGER :: nn, vn
INTEGER, PARAMETER:: untt1 = 55, untt2 = 60
LOGICAL :: Finds

IF (.NOT.(Finds('SYS.BIN ').AND.Finds('AINFO.BIN'))) RETURN
OPEN(untt1, FILE='SYS.BIN ', STATUS='OLD', FORM='BINARY')
OPEN(untt2, FILE='AINFO.BIN', STATUS='OLD', FORM='BINARY')

nn = 0
lvp(nn) = 0
DO WHILE(lvp(nn)>=0)
   nn = nn + 1
   READ (untt2) lvp(nn)
END DO

DO nn = 1, nn-1
   READ(untt1) name(nn)
   READ(untt1) (value(vn), vn = 1+lvp(nn-1), lvp(nn))
END DO
CLOSE(untt1, STATUS='DELETE')
CLOSE(untt2, STATUS='DELETE')
END


SUBROUTINE Rd_Bnd
USE Bnd_con
IMPLICIT NONE
INTEGER :: il, it, itmx
INTEGER, PARAMETER:: untt = 50
LOGICAL :: Finds

IF (.NOT. Finds('WTH.BIN ')) RETURN
OPEN (untt, FILE='WTH.BIN ', STATUS='OLD', FORM='BINARY')

READ (untt) iday
READ (untt) itmx
DO il = 1, iday
   READ (untt)(cmt(il, it), it = 1, itmx)
END DO
CLOSE (untt, STATUS='DELETE')
END


SUBROUTINE Scanin(id, ilmx, itmx, itpos, line, info)
USE Id_Lst
IMPLICIT NONE
CHARACTER (LEN=*), DIMENSION(366) :: line
INTEGER :: id, ilmx
INTEGER, DIMENSION(366) :: itmx
INTEGER, DIMENSION(2,26,366) ::  itpos
INTEGER, DIMENSION(26) :: info
```

```
INTEGER :: i, il, it, ip, indx, pnt, sts, fp, lp, fo=1, lo=1, ol=1
INTEGER, DIMENSION(0:66), PARAMETER :: class= &
        (/ 0, (1,i=1,8), 2, 2, (1,i=11,52), 3, 4, 4, 5, (6,i=57,66) /)
INTEGER, DIMENSION(0:10,0:7), PARAMETER :: pnttb = &
    RESHAPE(source=(/0,0,0,0,0,0,0,0,0,0,0, 0,4, 0,0,4,0,0,0,0,0,0,   &
                     0,4,0,0,4,0,2,0,0,0,0, 0,3,0,0,3,0,0,0,0,0,0,   &
                     0,7,10,7,0,0,0,0,0,0,0, 0,9,0,9,0,0,0,9,6,0,0,  &
                     0,8,5,8,4,5,6,8,8,6,5, 0,0, 0,1,1,1,1,0,0,0,0/), &
                     shape=(/11,8/) )
INTEGER, DIMENSION(0:6,0:6), PARAMETER :: ststb = &
    RESHAPE(source=(/0,0,0,0,0,0,0, 1,1,0,0,0,1,1, 1,1,0,0,0,1,1,   &
                     1,0,0,0,1,0,0,   1,0,0,1,0,1,1, 1,0,0,1,0,1,1, &
                     1,0,0,1,0,1,1/), shape=(/7,7/))
IF (id==0) THEN
   DO il = 1, ilmx
      DO it = 1, itmx(il)
         fp  = itpos(1,it,il)
         lp  = itpos(2,it,il)
         ip  = fp-1
         pnt = info(IT)
         DO WHILE (pnt>0 .AND. ip<lp)
            ip   = ip + 1
            indx = INDEX(lecha, line(il)(ip:ip))
            pnt  = pnttb(pnt, class(indx))
         END DO
         IF (pnttb(pnt, 7) == 0) &
            CALL Logmsg(line(il)(fp:lp),id,ip,il,'OPENS')
      END DO
   END DO
ELSE IF (id==1) THEN
   pnt = 1
   DO il = 1, ilmx
      DO it = 1, itmx(il)
         fp  = itpos(1,it,il)
         lp  = itpos(2,it,il)
         indx= INDEX(lecha, line(il)(fp:fp))
         sts = ststb(pnt, class(indx))
         pnt = pnttb(1, class(indx))
         ip  = fp
         DO WHILE (pnt>0 .AND. ip<lp)
            ip   = ip + 1
            indx = INDEX(lecha, line(il)(ip:ip))
            pnt  = pnttb(pnt, class(indx))
         END DO
         IF (sts==0) CALL Logmsg &
            (line(ol)(fo:lo)//' '//line(il)(fp:lp),id,lo,ol,'OPENS')
         IF (pnttb(pnt, 7)==0) THEN
            CALL Logmsg(line(il)(fp:lp),id,ip,il,'OPENS')
            pnt = 0
         ENDIF
         fo = fp
         lo = lp
         ol = il
      END DO
   END DO
   IF (pnt<5 .AND. pnt>0) &
       CALL Logmsg(line(ol)(fo:lo)//'  ',id,lo,ol,'OPENS')
ENDIF
END
```

```
SUBROUTINE Inc (il, ilmx, it, itmx, itpos, fp, lp, stat)
IMPLICIT NONE
INTEGER :: il, ilmx, it, fp, lp, stat
INTEGER, DIMENSION(366) :: itmx
INTEGER, DIMENSION(2,26,366) ::  itpos

IF (it<itmx(il)) THEN
   it = it + 1
ELSE IF (il<ilmx) THEN
   il = il + 1
   it = 1
ELSE
   stat = 100
ENDIF
fp = itpos(1,it,il)
lp = itpos(2,it,il)
END

SUBROUTINE Wrincn(ilmx, itmx, itpos, line)
IMPLICIT NONE
CHARACTER (LEN=*), DIMENSION(366) :: line
INTEGER :: ilmx
INTEGER, DIMENSION(366) :: itmx
INTEGER, DIMENSION(2,26,366) :: itpos

CHARACTER (LEN=10) :: name
REAL :: value
INTEGER :: il, it, fp, lp, qp, bp, stat, nn, vn
INTEGER, PARAMETER:: untt1=50, untt2=60, nmx=250, vmx=2700
LOGICAL :: Finds

IF (Finds('MSG.LOG')) RETURN
CALL Onfile(untt1, 'SYS.BIN ', 'SEQUENTIAL', 'BINARY')
CALL Onfile(untt2, 'AINFO.BIN', 'SEQUENTIAL', 'BINARY')
il = 1
it = 1
fp = itpos(1,it,il)
lp = itpos(2,it,il)
DO WHILE (stat<9)
   qp = INDEX(line(il)(fp:lp), '=')
   bp = (fp-1) + MOD(((lp-fp)+qp+1),((lp-fp)+2))
   name = line(il)(fp:bp)
   WRITE (untt1) name
   nn = nn + 1
   stat = 2

   IF ((lp-bp)>1) THEN
      fp = bp+2
   ELSE IF ((lp-bp)==1) THEN
      CALL Inc(il, ilmx, it, itmx, itpos, fp, lp, stat)
   ELSE
      CALL Inc(il, ilmx, it, itmx, itpos, fp, lp, stat)
      IF (fp==lp) THEN
         CALL Inc(il, ilmx, it, itmx, itpos, fp, lp, stat)
      ELSE
         fp = fp+1
      ENDIF
   ENDIF

   DO WHILE (stat == 2)
      READ (line(il)(fp:lp),'(G12.0)') value
```

```
      WRITE (untt1) value
      vn = vn + 1
      CALL Inc(il, ilmx, it, itmx, itpos, fp, lp, stat)
      IF (line(il)(fp:fp)>'='.AND. stat<10) stat = 1
   END DO
   WRITE (untt2) vn
   IF (nn>nmx .OR. vn>vmx) Stat = 9
END DO

IF (stat==9) THEN
   CALL Logmsg(' > Storage',1, 1000, 1000,'OPENS')
   CALL Logmsg ('TRSCIC', 11,0, 0,'CLOSES')
ENDIF
WRITE (untt2) -1
CLOSE (untt1)
CLOSE (untt2)
END

FUNCTION Finds(files)
IMPLICIT NONE
CHARACTER (LEN=*):: files
LOGICAL :: fexist, finds

INQUIRE (FILE=files, EXIST=fexist)
finds = fexist
END


SUBROUTINE Deletes(files)
IMPLICIT NONE
CHARACTER (LEN=*) :: files

OPEN(32, FILE=files, STATUS='OLD', FORM='FORMATTED')
CLOSE(32, STATUS='DELETE')
END


SUBROUTINE Logrun (caller, msg, val)
USE Ctrl_Time
IMPLICIT NONE
CHARACTER (LEN=*):: caller, msg
REAL :: val

LOGICAL :: opened=.false.

IF (.NOT.opened .AND. msg/='FINAL') THEN
   CALL Onfile(31, 'RUN.LOG', 'APPEND', 'FORMATTED')
   opened = .true.
ENDIF
IF (opened) THEN
   IF (msg=='FINAL') THEN
      WRITE (31,'(/, 1X, A)') 'End of message'
      WRITE (6, '(1X, A)') 'MESSAGE: Message in RUN.LOG'
      CLOSE (31)
   ELSE
      WRITE &
         (31,'(1X,''From '',A,'':'','' At TIME='',F6.2,'','',3X,A,F8.2)') &
         caller, time, msg, val
   ENDIF
ENDIF
END
```

ภาคผนวกที่ 3 ตัวอย่าง file ข้อมูลนำเข้า (rnspec.sys) ค่าคุณสมบัติพืช ดินและภาวะเริ่มต้นระบบ

```
!  simulation control and initial conditions at Khon Kaen 2548
     SDATE =234.0, OUTDEL=7.0, p=0.25,  wc=90.71, wp=0.0

! latitude
   la=16.43

! phenological parameters for s-model
    dj=21.0, di=7.0, dp=28.0, dg=28.0
    tb=8.0, top=27.0
    pc=12.15, po=11.9

! light interception, and photosynthesis
    kc=0.7, ec=1.5

! biomass
    bs=1.0163, as=0.7976

! leaf area
     bl=0.6916, al=0.0373, lc=0.0075

! soil properties
    qs=0.47, qr=0.04

! volumetric soil water content
     vwc=  1.0,0.13, 100.0,0.04,
         160.0,0.47, 224.0,0.47, 285.0,0.47,
         303.0,0.47, 365.0,0.13

! no weed
    fwe= 0.0, 0.0
       366.0, 0.0
```

ภาคผนวกที่ 4 ตัวอย่าง file ข้อมูลนำเข้า (rnspec.wth) สภาพฟ้าอากาศรายวัน

```
! Station:    Muang, Khon Kaen, Thailand
! Column      Description
! 1           Station code
! 2           Year
! 3           Day
! 4           Solar radiation MJ/m2-d
! 5           Daily minimum temperature C
! 6           Daily maximum temperature C
! 7           % Relative humidity
! 8           Wind speed m/s
! 9           Rainfall mm
```

201	2005	1	17.0	10.8	23.5	75.00	1.14	0.0	201	2005	51	12.6	19.9	29.0	71.00	0.91	0.0
201	2005	2	17.0	9.4	25.4	80.00	0.77	0.0	201	2005	52	17.2	17.8	31.7	78.00	0.69	0.0
201	2005	3	16.2	11.5	28.8	80.00	0.64	0.0	201	2005	53	17.7	19.3	34.5	79.00	0.28	0.0
201	2005	4	16.1	13.6	29.1	81.00	0.31	0.0	201	2005	54	16.3	21.8	39.0	81.00	0.43	0.0
201	2005	5	16.6	13.6	28.7	86.00	1.09	0.0	201	2005	55	18.7	24.5	40.2	73.00	0.61	0.0
201	2005	6	16.1	14.8	29.3	84.00	0.91	0.0	201	2005	56	18.5	21.8	39.9	72.00	0.91	0.0
201	2005	7	16.5	15.0	30.1	89.00	1.04	0.0	201	2005	57	18.6	23.3	39.9	73.00	0.95	0.0
201	2005	8	16.2	15.8	30.1	87.00	0.75	0.0	201	2005	58	19.4	22.7	39.6	67.00	0.48	0.0
201	2005	9	16.1	16.2	29.3	88.00	1.07	0.0	201	2005	59	18.0	23.9	38.7	63.00	0.68	0.0
201	2005	10	16.6	15.5	29.8	89.00	1.15	0.0	201	2005	60	18.4	23.0	34.8	71.00	1.15	0.0
201	2005	11	16.7	15.7	30.8	92.00	0.99	0.0	201	2005	61	17.3	23.0	37.2	76.00	0.70	0.0
201	2005	12	15.3	16.6	30.7	91.00	0.97	0.0	201	2005	62	10.8	21.8	28.7	75.00	0.95	0.0
201	2005	13	14.9	17.9	29.3	92.00	0.72	0.0	201	2005	63	8.1	20.5	29.0	59.00	1.72	0.0
201	2005	14	2.4	16.7	20.0	89.00	1.16	0.0	201	2005	64	14.4	18.7	26.3	53.00	1.56	0.0
201	2005	15	8.3	15.0	22.3	90.00	1.01	0.0	201	2005	65	20.2	13.0	28.4	69.00	1.36	0.0
201	2005	16	14.1	16.2	26.1	72.00	0.92	0.0	201	2005	66	19.7	14.8	32.0	73.00	0.88	0.0
201	2005	17	15.9	15.5	28.1	77.00	0.73	0.0	201	2005	67	18.5	17.5	34.5	66.00	0.59	0.0
201	2005	18	15.5	16.2	31.0	85.00	0.70	0.0	201	2005	68	18.0	19.9	38.4	75.00	0.48	0.0
201	2005	19	13.8	18.1	31.3	87.00	0.72	0.0	201	2005	69	17.3	25.1	38.4	70.00	0.55	0.0
201	2005	20	14.3	20.0	30.8	89.00	0.88	0.0	201	2005	70	18.8	26.6	40.5	70.00	1.03	0.0
201	2005	21	15.4	18.6	29.5	88.00	0.97	0.0	201	2005	71	18.7	27.2	40.8	74.00	0.77	0.0
201	2005	22	15.6	17.6	30.9	89.00	0.72	0.0	201	2005	72	18.9	25.4	36.6	62.00	0.72	0.0
201	2005	23	16.1	18.1	31.5	93.00	0.49	0.0	201	2005	73	7.5	18.7	28.1	67.00	1.80	10.6
201	2005	24	13.9	19.1	33.8	90.00	0.54	0.0	201	2005	74	16.9	22.1	35.4	84.00	0.37	0.0
201	2005	25	14.3	20.7	34.2	90.00	0.81	0.0	201	2005	75	16.3	25.4	37.2	76.00	0.74	0.0
201	2005	26	15.1	18.8	34.3	95.00	0.75	0.0	201	2005	76	19.1	25.4	38.4	74.00	0.82	0.0
201	2005	27	16.1	20.1	35.2	90.00	0.52	0.0	201	2005	77	12.8	26.0	37.8	69.00	0.94	0.0
201	2005	28	15.5	20.6	35.8	92.00	0.75	0.0	201	2005	78	18.7	23.6	34.8	70.00	1.32	0.0
201	2005	29	14.7	21.8	35.5	88.00	0.72	0.0	201	2005	79	20.7	22.7	36.0	76.00	1.04	0.0
201	2005	30	14.2	19.9	36.3	78.00	0.74	0.0	201	2005	80	18.8	25.1	39.0	72.00	0.60	0.0
201	2005	31	14.2	21.5	34.9	77.00	0.53	0.0	201	2005	81	18.3	26.3	39.9	77.00	0.73	0.0
201	2005	32	14.9	22.6	33.4	92.00	0.20	0.0	201	2005	82	21.0	24.8	39.6	78.00	0.91	0.0
201	2005	33	13.9	24.8	34.9	93.00	0.34	0.0	201	2005	83	17.3	26.3	33.9	73.00	1.09	0.0
201	2005	34	15.7	25.0	34.9	93.00	0.49	0.0	201	2005	84	10.2	20.8	31.7	67.00	1.69	0.4
201	2005	35	15.3	23.3	35.3	94.00	0.72	0.0	201	2005	85	15.2	20.2	34.5	82.00	1.56	0.6
201	2005	36	15.6	21.6	34.7	84.00	0.45	0.0	201	2005	86	15.0	26.3	36.9	74.00	0.66	0.0
201	2005	37	16.6	22.4	35.9	94.00	0.21	0.0	201	2005	87	19.6	26.6	40.5	81.00	1.45	0.0
201	2005	38	15.2	24.3	34.9	74.00	0.27	0.0	201	2005	88	17.4	28.1	41.1	71.00	0.89	0.0
201	2005	39	16.8	22.3	36.5	73.00	0.47	0.0	201	2005	89	19.5	25.4	40.2	81.00	0.99	0.0
201	2005	40	15.3	22.4	35.8	56.00	0.33	0.0	201	2005	90	21.4	29.0	42.3	71.00	0.65	0.0
201	2005	41	12.4	25.4	33.0	62.00	0.45	0.0	201	2005	91	18.1	30.2	39.6	66.00	0.94	0.0
201	2005	42	17.3	21.0	31.3	69.00	0.90	0.0	201	2005	92	13.7	25.4	37.2	79.00	1.16	14.8
201	2005	43	18.4	19.9	32.2	73.00	0.56	0.0	201	2005	93	18.9	24.5	33.0	95.00	1.37	7.0
201	2005	44	18.1	19.4	34.7	73.00	0.34	0.0	201	2005	94	2.4	22.1	27.5	70.00	1.57	0.8
201	2005	45	15.2	23.6	34.3	72.00	0.42	0.0	201	2005	95	7.7	21.4	26.6	86.00	1.38	0.0
201	2005	46	18.4	23.7	35.3	65.00	0.76	0.0	201	2005	96	18.7	22.4	34.5	89.00	1.13	0.0
201	2005	47	18.9	22.4	39.3	77.00	1.13	0.0	201	2005	97	22.0	22.7	39.3	91.00	0.42	0.0
201	2005	48	19.6	23.0	40.2	72.00	0.92	0.0	201	2005	98	19.1	25.4	40.2	74.00	0.55	0.0
201	2005	49	16.7	22.1	40.2	58.00	0.79	0.0	201	2005	99	20.2	28.4	40.5	76.00	1.15	0.0
201	2005	50	16.8	22.7	34.8	68.00	0.27	0.0	201	2005	100	22.0	28.7	41.4	71.00	1.37	0.0

201	2005	101	21.7	28.7	42.0	76.00	0.93	0.0
201	2005	102	22.0	28.4	42.3	76.00	1.25	0.0
201	2005	103	11.4	24.8	33.0	88.00	1.47	2.0
201	2005	104	20.3	23.9	34.8	76.00	0.83	0.0
201	2005	105	18.3	26.6	38.4	77.00	0.58	0.0
201	2005	106	13.7	26.6	38.1	87.00	0.81	1.2
201	2005	107	17.0	27.8	39.3	80.00	0.90	0.0
201	2005	108	17.9	26.3	37.8	84.00	1.31	0.0
201	2005	109	14.3	26.3	38.1	84.00	0.73	0.0
201	2005	110	21.3	26.3	37.5	79.00	0.93	0.0
201	2005	111	21.0	26.6	38.4	74.00	0.89	0.0
201	2005	112	18.9	28.7	40.5	81.00	0.66	0.0
201	2005	113	16.6	27.2	39.9	78.00	1.17	0.0
201	2005	114	20.2	26.9	41.1	81.00	0.86	0.0
201	2005	115	19.6	29.0	41.1	77.00	0.81	0.0
201	2005	116	14.4	24.8	40.5	80.00	0.81	25.8
201	2005	117	18.3	26.3	36.3	87.00	1.24	0.0
201	2005	118	19.6	28.7	39.6	76.00	0.75	0.0
201	2005	119	21.4	26.0	39.6	82.00	1.11	0.2
201	2005	120	21.9	23.9	40.2	76.00	1.54	9.0
201	2005	121	21.6	24.2	38.4	92.00	1.33	1.4
201	2005	122	18.9	27.8	39.3	82.00	1.03	0.0
201	2005	123	19.3	29.0	39.3	69.00	1.03	0.0
201	2005	124	20.9	26.6	39.9	76.00	0.88	0.0
201	2005	125	19.4	29.3	39.6	68.00	1.09	0.0
201	2005	126	19.8	28.4	40.5	68.00	1.42	0.0
201	2005	127	18.6	25.7	37.8	71.00	1.05	11.6
201	2005	128	15.9	26.3	36.3	90.00	0.76	0.0
201	2005	129	20.9	27.2	37.2	83.00	1.30	0.0
201	2005	130	21.6	29.0	38.7	76.00	1.56	0.0
201	2005	131	22.2	29.0	39.3	71.00	1.55	0.0
201	2005	132	18.7	27.5	39.3	74.00	1.73	0.0
201	2005	133	18.6	26.6	38.7	84.00	1.22	14.2
201	2005	134	8.7	26.0	33.6	92.00	1.14	2.0
201	2005	135	16.9	26.0	36.3	92.00	0.72	0.0
201	2005	136	16.0	27.5	36.6	83.00	0.54	0.0
201	2005	137	17.3	27.2	38.1	83.00	1.15	0.0
201	2005	138	21.1	26.6	38.4	88.00	0.85	0.2
201	2005	139	15.9	26.3	36.6	85.00	1.17	0.0
201	2005	140	21.8	25.7	39.0	82.00	0.87	5.0
201	2005	141	15.8	25.7	34.8	91.00	1.23	0.2
201	2005	142	14.5	25.7	35.4	90.00	1.00	2.6
201	2005	143	13.9	26.3	36.3	89.00	0.84	1.8
201	2005	144	20.0	26.6	37.5	86.00	0.89	0.0
201	2005	145	21.7	27.8	37.8	83.00	1.35	0.0
201	2005	146	19.7	27.8	37.5	80.00	1.46	0.0
201	2005	147	21.9	28.4	39.3	71.00	1.22	0.0
201	2005	148	14.4	26.3	38.7	74.00	1.14	21.8
201	2005	149	13.8	25.1	34.5	95.00	1.14	31.0
201	2005	150	10.5	26.0	33.0	92.00	0.76	0.2
201	2005	151	18.0	26.6	36.3	89.00	0.60	14.8
201	2005	152	18.1	26.9	34.2	90.00	1.17	0.2
201	2005	153	17.2	27.2	35.1	83.00	1.40	0.0
201	2005	154	16.8	28.1	35.7	80.00	1.40	0.0
201	2005	155	22.1	27.5	37.2	79.00	1.51	0.0
201	2005	156	22.6	28.4	37.2	76.00	1.63	0.0
201	2005	157	19.8	27.5	36.9	78.00	2.00	1.2
201	2005	158	15.7	28.4	35.7	82.00	0.96	1.0
201	2005	159	8.8	27.2	33.3	88.00	0.63	0.6
201	2005	160	11.6	27.2	34.8	88.00	0.62	0.6
201	2005	161	12.5	26.0	35.4	83.00	1.03	4.6
201	2005	162	18.1	26.9	36.3	81.00	1.47	0.0
201	2005	163	16.1	28.1	36.3	81.00	1.71	0.0
201	2005	164	17.4	28.1	36.9	81.00	1.44	0.0
201	2005	165	11.1	28.4	34.5	77.00	1.72	0.0
201	2005	166	17.7	27.2	36.0	75.00	1.61	0.0
201	2005	167	14.4	28.1	36.6	76.00	2.17	0.0
201	2005	168	19.4	26.9	37.5	85.00	1.89	2.8
201	2005	169	17.9	26.3	37.2	93.00	1.41	3.8
201	2005	170	14.9	26.3	36.0	87.00	0.98	0.8
201	2005	171	20.1	26.3	36.3	92.00	1.03	4.4
201	2005	172	17.6	25.4	36.9	91.00	0.98	46.0
201	2005	173	8.8	24.8	32.0	90.00	1.36	23.4
201	2005	174	11.2	25.4	35.4	85.00	0.36	16.6
201	2005	175	11.9	24.8	32.7	94.00	1.15	12.2
201	2005	176	11.3	25.7	31.4	88.00	1.24	0.0
201	2005	177	17.7	26.9	34.8	84.00	1.18	0.0
201	2005	178	18.5	27.5	35.1	78.00	1.55	0.0
201	2005	179	11.9	28.4	33.6	79.00	1.50	0.0
201	2005	180	19.3	26.3	34.8	80.00	1.19	1.8
201	2005	181	13.1	26.6	34.5	83.00	1.39	4.8
201	2005	182	14.0	26.9	34.5	79.00	0.91	0.0
201	2005	183	13.2	26.6	34.2	83.00	0.91	0.4
201	2005	184	11.9	26.0	34.5	90.00	0.77	0.6
201	2005	185	14.9	26.3	34.2	88.00	0.61	0.0
201	2005	186	10.0	25.4	33.0	91.00	0.83	24.2
201	2005	187	17.7	25.1	35.7	92.00	0.57	0.2
201	2005	188	18.3	26.0	36.0	89.00	0.88	0.0
201	2005	189	14.9	26.3	35.7	78.00	1.18	0.6
201	2005	190	14.2	27.5	34.5	76.00	1.05	0.0
201	2005	191	19.8	26.0	36.3	91.00	1.33	0.0
201	2005	192	16.8	26.3	36.3	77.00	0.74	0.0
201	2005	193	8.4	25.7	32.3	85.00	0.91	10.4
201	2005	194	9.0	24.2	33.3	92.00	0.92	11.0
201	2005	195	17.5	24.5	34.8	92.00	0.73	0.0
201	2005	196	16.6	25.7	36.6	97.00	0.53	6.4
201	2005	197	23.3	25.7	37.8	88.00	0.78	0.0
201	2005	198	24.2	27.8	38.4	82.00	0.55	0.0
201	2005	199	20.7	27.8	37.2	78.00	0.75	0.0
201	2005	200	15.9	26.9	37.5	85.00	1.01	3.0
201	2005	201	18.8	25.7	36.9	90.00	1.00	2.4
201	2005	202	11.9	26.3	33.0	86.00	1.12	4.8
201	2005	203	13.4	26.3	34.2	90.00	1.30	5.0
201	2005	204	10.4	26.0	33.0	89.00	1.53	9.4
201	2005	205	13.8	25.4	32.7	89.00	2.04	4.0
201	2005	206	5.8	25.4	29.3	90.00	2.13	4.2
201	2005	207	8.2	25.4	29.9	92.00	1.31	1.6
201	2005	208	16.0	26.0	34.2	78.00	1.66	0.2
201	2005	209	18.2	26.6	35.4	85.00	1.58	0.0
201	2005	210	18.4	26.0	35.4	89.00	1.18	0.0
201	2005	211	11.8	27.2	33.0	83.00	1.85	0.0
201	2005	212	6.3	26.9	30.5	84.00	2.33	0.0
201	2005	213	13.2	26.6	32.7	79.00	2.70	0.0
201	2005	214	11.3	25.4	33.0	91.00	1.83	1.4
201	2005	215	17.9	27.2	36.3	82.00	1.31	0.6
201	2005	216	15.9	26.6	34.8	77.00	1.77	0.4
201	2005	217	10.6	27.8	31.7	79.00	2.34	0.0
201	2005	218	15.7	26.6	35.1	83.00	2.21	0.0
201	2005	219	11.0	25.7	34.5	87.00	2.29	4.6
201	2005	220	15.4	26.3	34.2	90.00	1.44	0.2
201	2005	221	9.8	26.0	31.4	90.00	1.82	0.4
201	2005	222	11.3	26.6	33.9	94.00	1.33	1.0
201	2005	223	12.1	26.9	33.3	90.00	1.22	0.0
201	2005	224	8.6	25.1	31.4	95.00	1.99	4.0
201	2005	225	18.6	25.7	35.1	92.00	1.55	0.6
201	2005	226	14.3	26.3	33.0	91.00	1.63	0.0
201	2005	227	13.5	26.6	33.9	87.00	1.60	0.0
201	2005	228	11.1	25.7	32.7	91.00	1.06	11.2
201	2005	229	7.1	24.2	31.7	94.00	1.30	16.0
201	2005	230	10.2	25.1	34.2	95.00	1.14	11.8
201	2005	231	13.8	24.5	33.9	97.00	1.06	10.0
201	2005	232	13.1	26.6	34.5	93.00	0.83	0.0
201	2005	233	14.5	26.6	36.0	84.00	1.13	2.6
201	2005	234	12.9	25.4	33.3	95.00	1.17	1.4

201	2005	235	17.9	25.4	35.1	91.00	0.65	6.0
201	2005	236	19.5	24.8	36.6	94.00	0.67	0.0
201	2005	237	18.7	26.0	36.3	95.00	0.63	0.0
201	2005	238	12.2	26.0	33.9	86.00	0.69	0.0
201	2005	239	17.3	24.2	36.3	89.00	0.60	19.4
201	2005	240	15.3	24.8	34.8	92.00	0.87	2.0
201	2005	241	11.3	25.4	33.6	94.00	0.74	0.4
201	2005	242	13.8	26.0	34.5	93.00	0.70	13.8
201	2005	243	13.1	23.9	33.0	95.00	0.82	38.2
201	2005	244	20.4	25.4	35.1	93.00	0.89	0.0
201	2005	245	9.9	26.3	33.3	90.00	0.71	2.2
201	2005	246	11.5	26.0	33.3	93.00	0.81	28.0
201	2005	247	14.2	26.3	34.8	91.00	0.59	0.2
201	2005	248	16.2	26.3	34.2	89.00	0.54	1.8
201	2005	249	16.1	25.7	33.3	91.00	0.65	16.2
201	2005	250	15.7	25.1	33.9	93.00	0.65	11.6
201	2005	251	11.0	25.4	35.7	92.00	0.90	2.8
201	2005	252	6.4	25.4	30.5	94.00	0.79	35.8
201	2005	253	5.4	25.1	31.4	95.00	0.55	47.4
201	2005	254	14.8	25.1	31.7	94.00	0.52	0.6
201	2005	255	15.3	25.7	33.9	89.00	0.72	23.2
201	2005	256	6.1	25.1	31.4	93.00	0.95	5.6
201	2005	257	6.7	25.4	28.7	87.00	1.83	6.2
201	2005	258	17.1	24.5	33.6	93.00	1.70	0.2
201	2005	259	16.9	25.7	35.7	92.00	0.50	0.0
201	2005	260	17.8	26.0	35.4	93.00	0.56	0.0
201	2005	261	6.3	25.4	31.4	94.00	0.94	4.6
201	2005	262	7.9	24.5	31.4	94.00	1.28	14.4
201	2005	263	17.4	25.4	33.6	93.00	0.97	12.4
201	2005	264	9.2	25.4	31.4	95.00	0.89	0.2
201	2005	265	15.0	25.1	35.4	95.00	0.66	0.0
201	2005	266	13.6	25.1	35.1	96.00	0.50	0.0
201	2005	267	16.2	25.1	35.7	94.00	0.51	0.4
201	2005	268	12.1	24.5	35.7	94.00	0.46	42.4
201	2005	269	14.1	26.0	34.5	96.00	0.55	0.0
201	2005	270	8.3	26.0	32.7	94.00	0.87	9.2
201	2005	271	9.9	26.0	32.7	92.00	1.45	1.4
201	2005	272	13.4	25.7	34.2	93.00	0.95	1.8
201	2005	273	10.3	26.0	33.9	93.00	0.75	1.6
201	2005	274	20.1	25.4	35.4	92.00	0.75	0.0
201	2005	275	20.4	25.7	35.7	87.00	0.47	0.0
201	2005	276	20.2	25.7	35.4	91.00	0.70	0.0
201	2005	277	20.2	25.1	35.1	87.00	0.71	0.0
201	2005	278	16.6	25.4	34.8	90.00	0.85	0.0
201	2005	279	17.2	25.1	34.5	85.00	0.77	0.0
201	2005	280	19.2	25.1	34.8	90.00	0.71	0.0
201	2005	281	7.5	25.1	30.8	82.00	1.05	3.2
201	2005	282	17.0	24.2	33.3	69.00	1.50	0.0
201	2005	283	18.9	23.0	33.9	68.00	1.23	0.0
201	2005	284	17.1	23.6	34.8	80.00	1.15	0.0
201	2005	285	15.1	24.8	33.9	86.00	1.18	2.6
201	2005	286	16.4	26.3	35.1	87.00	0.86	0.0
201	2005	287	18.9	24.5	34.8	87.00	0.90	0.0
201	2005	288	18.4	23.6	35.1	79.00	1.01	0.0
201	2005	289	19.4	22.7	34.2	81.00	0.98	0.0
201	2005	290	16.1	22.7	34.5	81.00	1.12	0.0
201	2005	291	18.1	23.6	34.5	82.00	0.87	0.0
201	2005	292	16.7	23.6	34.5	78.00	1.28	0.0
201	2005	293	16.4	24.2	34.8	79.00	1.49	0.0
201	2005	294	17.9	22.1	34.2	89.00	1.34	0.0
201	2005	295	18.2	22.4	34.2	76.00	1.45	0.0
201	2005	296	14.8	23.3	33.6	77.00	1.62	0.0
201	2005	297	11.8	24.8	33.0	75.00	1.59	0.0
201	2005	298	13.7	23.6	33.0	82.00	1.29	0.0
201	2005	299	14.8	24.5	33.9	77.00	1.10	0.0
201	2005	300	17.1	23.3	34.8	77.00	0.89	0.0
201	2005	301	16.9	23.3	35.1	84.00	1.04	0.0
201	2005	302	16.6	22.7	35.4	86.00	0.89	0.0
201	2005	303	17.1	22.4	33.9	87.00	1.08	0.0
201	2005	304	16.5	20.5	33.0	84.00	1.19	0.0
201	2005	305	15.4	20.8	32.7	76.00	1.32	0.0
201	2005	306	14.6	22.7	35.4	76.00	1.17	0.0
201	2005	307	11.1	25.4	33.0	80.00	1.12	0.0
201	2005	308	15.8	23.9	36.6	86.00	0.77	1.4
201	2005	309	12.6	24.8	34.5	94.00	1.10	1.6
201	2005	310	14.7	26.0	36.3	85.00	0.59	0.0
201	2005	311	8.7	25.1	35.4	87.00	0.69	25.6
201	2005	312	12.4	24.8	34.8	89.00	0.79	4.0
201	2005	313	14.6	24.2	34.8	94.00	0.80	3.8
201	2005	314	14.7	24.5	35.7	91.00	0.47	0.0
201	2005	315	15.3	25.1	36.3	90.00	0.53	0.0
201	2005	316	13.0	25.4	35.7	86.00	0.65	11.8
201	2005	317	12.9	25.7	35.4	89.00	0.86	0.0
201	2005	318	11.6	25.7	34.2	89.00	0.78	0.0
201	2005	319	15.7	25.4	34.8	90.00	0.58	0.0
201	2005	320	15.4	23.3	34.5	84.00	1.01	0.0
201	2005	321	14.7	22.4	33.9	87.00	1.16	0.0
201	2005	322	7.6	22.7	30.8	84.00	1.10	0.0
201	2005	323	12.9	21.8	30.8	78.00	1.40	0.0
201	2005	324	16.0	18.1	29.9	78.00	1.60	0.0
201	2005	325	15.8	17.5	29.6	84.00	1.23	0.0
201	2005	326	16.5	16.6	29.0	81.00	1.44	0.0
201	2005	327	13.9	16.6	28.7	85.00	1.35	0.0
201	2005	328	15.9	17.2	30.2	77.00	1.32	0.0
201	2005	329	10.1	18.1	29.6	77.00	1.43	0.0
201	2005	330	10.9	22.4	30.8	79.00	0.81	0.0
201	2005	331	15.1	19.9	33.3	82.00	0.79	0.0
201	2005	332	14.5	20.5	34.2	86.00	0.88	0.0
201	2005	333	14.3	20.8	34.2	83.00	1.06	0.0
201	2005	334	14.4	20.8	33.9	85.00	0.96	0.0
201	2005	335	15.3	19.9	33.9	87.00	1.04	0.0
201	2005	336	14.8	21.1	34.2	84.00	0.90	0.0
201	2005	337	14.8	21.8	35.1	83.00	0.96	0.0
201	2005	338	11.7	22.1	34.2	86.00	0.85	0.0
201	2005	339	9.9	20.2	31.4	84.00	1.02	0.0
201	2005	340	14.1	21.8	30.8	72.00	1.12	0.0
201	2005	341	15.2	18.1	29.6	70.00	1.36	0.0
201	2005	342	15.0	17.2	29.6	76.00	1.25	0.0
201	2005	343	15.1	17.5	31.4	76.00	0.94	0.0
201	2005	344	12.8	19.0	32.3	81.00	0.66	0.0
201	2005	345	13.3	19.9	33.3	84.00	0.69	0.0
201	2005	346	14.6	19.9	31.4	73.00	0.89	0.0
201	2005	347	13.6	17.2	29.6	81.00	1.07	0.0
201	2005	348	14.5	16.6	30.2	78.00	1.29	0.0
201	2005	349	12.5	16.6	28.1	75.00	1.58	0.0
201	2005	350	14.3	16.0	27.8	69.00	1.31	0.0
201	2005	351	14.5	16.6	28.1	76.00	1.31	0.0
201	2005	352	14.7	13.3	27.2	66.00	1.34	0.0
201	2005	353	14.8	12.4	27.2	77.00	1.29	0.0
201	2005	354	14.5	14.5	29.0	77.00	1.03	0.0
201	2005	355	9.1	17.2	26.0	68.00	0.93	1.2
201	2005	356	11.4	16.3	24.8	65.00	0.99	0.0
201	2005	357	12.5	16.3	26.6	64.00	0.83	0.0
201	2005	358	11.4	16.3	29.6	51.00	0.31	0.0
201	2005	359	9.9	21.8	31.7	71.00	0.28	0.0
201	2005	360	9.1	22.7	31.4	87.00	0.28	0.0
201	2005	361	12.2	21.8	32.3	86.00	0.53	0.0
201	2005	362	14.7	20.2	31.7	84.00	0.51	0.0
201	2005	363	15.2	17.8	33.0	85.00	0.80	0.0
201	2005	364	15.3	17.5	33.0	83.00	0.75	0.0
201	2005	365	15.2	18.1	34.2	78.00	0.66	0.0

ภาคผนวกที่ 5 ตัวอย่าง output file แสดงผลลัพธ์จากการจำลองสถานการณ์

TIME	et	p	l	wp	ws	wc	lw
0.0000	3.4384	0.25000	0.75827	0.0000	77.866	90.710	0.0000
7.0000	3.0727	0.25952	1.1915	0.0000	149.70	172.57	0.0000
14.000	4.1927	0.27456	1.5852	0.0000	226.21	259.05	0.0000
21.000	3.9869	0.29877	1.9560	0.0000	306.56	349.36	0.0000
28.000	2.2955	0.33582	2.3338	0.0000	395.71	449.12	0.0000
35.000	3.7127	0.41140	2.7334	0.0000	497.32	562.38	0.0000
42.000	5.1069	0.48480	3.1501	0.0000	610.58	688.18	0.0000
49.000	4.8098	0.56009	3.6486	0.0000	755.07	848.15	0.0000
56.000	4.1698	0.63723	3.7982	0.0000	911.79	1021.1	0.0000
63.000	3.1869	0.71731	2.9389	114.58	955.10	1183.4	0.0000
70.000	4.2069	0.79703	1.9392	247.86	955.10	1316.7	0.0000
77.000	2.2255	0.87352	1.2143	344.52	955.10	1413.3	0.0000
84.000	2.5403	0.94545	0.76744	404.10	955.10	1472.9	0.0000
89.000	2.5614	1.0014	0.55097	432.97	955.10	1501.8	0.0000

Termination: maturation

กิจกรรมที่เกี่ยวข้องกับการนำผลจากโครงการไปใช้ประโยชน์

ได้จัดฝึกอบรมเชิงปฏิบัติการ การใช้แบบจำลองที่เรียบง่าย ให้กับบุคลากรของสถาบันวิจัยพืชไร่ กระทรวงเกษตรและสหกรณ์ ซึ่งเป็นผู้ที่มีประสบการณ์ในการใช้แบบจำลองร่วมกับระบบสารสนเทศและระบบประมวลผลอื่น ๆ ในการปฏิบัติการ

การฝึกอบรมแบ่งเป็นสองระยะ ระยะแรก วันที่ 26-27 เมษายน 2550 เน้นที่หลักการ วิธีการสร้าง คุณสมบัติ และข้อจำกัดของแบบจำลองที่เรียบง่าย ระยะที่สอง วันที่ 28-29 พฤษภาคม 2550 เป็นการแลกเปลี่ยนความเห็นและปรึกษาถึงแนวทางการปรับปรุงและขยายการใช้แบบจำลองดังกล่าว

ตารางเปรียบเทียบวัตถุประสงค์ กิจกรรมที่วางแผนไว้ กิจกรรมที่ได้ดำเนินการ และผลที่ได้รับตลอดโครงการ

วัตถุประสงค์ของโครงการ	กิจกรรมที่วางแผนไว้	กิจกรรมที่ได้ดำเนินการ	ผลที่ได้รับตลอดโครงการ
พัฒนาแบบจำลองข้าวที่เรียบง่าย ทั้งในแง่ของหลักการการใช้งานและความต้องการ input เป็นแบบจำลองแบบที่เปิดให้ผู้ใช้สามารถทำการศึกษาและดัดแปลงได้	เลือกสถานที่ และทำการเก็บข้อมูลสภาพฟ้าอากาศ ดิน การเจริญเติบโตข้าว และอัตราการคายระเหยโดยวิธี Bowen Ratio Energy Balance	ได้เลือกสถานที่และทำการเก็บข้อมูลที่ อ.พยัคฆภูมิ อ.เกษตรวิสัย อ.โพนทราย และ อ. สุวรรณภูมิ โดยอิงกับลักษณะการกระจายตัวฝนและชุดดิน	แบบจำลองข้าวที่เรียบง่าย สามารถใช้เป็นทางเลือกในการวิเคราะห์และวางแผนการการผลิตข้าวในสภาพนาน้ำฝน ซึ่งมีคุณสมบัติโดยสังเขปดังนี้
	พัฒนา และตรวจสอบการทำงานของแบบจำลองที่ได้พัฒนาขึ้นโดยเปรียบเทียบกับข้อมูลภาคสนามและผลของการจำลองโดยการใช้แบบจำลอง ORYZA2000 และ DSSAT	เป็นไปตามแผน ยกเว้นการประเมินแบบจำลอง DSSAT ได้ดำเนินการโดยโครงการกลาง และได้ทดสอบประยุกต์ใช้แบบจำลองที่ได้พัฒนาขึ้นวิเคราะห์ช่องว่างของผลผลิตจากผลกระทบของวัชพืช	ผู้ใช้ทำความเข้าใจได้ง่าย ต้องการ input น้อย และเปิดให้ผู้ใช้ปรับเปลี่ยนได้ สามารถใช้ประเมินผลผลิตข้าวขาวดอกมะลิ 105 ในสภาพทุ่งกุลาซึ่งเป็นนาหว่านอาศัยน้ำฝนได้

(ต่อ) ตารางเปรียบเทียบวัตถุประสงค์ กิจกรรมที่วางแผนไว้ กิจกรรมที่ได้ดำเนินการ และผลที่ได้รับตลอดโครงการ

วัตถุประสงค์ของโครงการ	กิจกรรมที่วางแผนไว้	กิจกรรมที่ได้ดำเนินการ	ผลที่ได้รับตลอดโครงการ
พัฒนาแบบจำลองให้เป็นองค์ประกอบหนึ่ง ของระบบสนับสนุนการวางแผนจัดการทรัพยากรเพื่อการผลิตข้าวในสภาพนาน้ำฝน	สร้างโครงสร้างและรูปแบบการเชื่อมต่อกับข้อมูลเชิงพื้นที่อื่น ๆ	เป็นไปตามแผน	สามารถเชื่อมต่อกับระบบฐานข้อมูลและการประมวลผลอื่น ๆ เพื่อบูรณาการเป็นระบบสนับสนุนการตัดสินใจได้
ศึกษาแนวทางการใช้ประโยชน์จากแบบจำลองที่พัฒนาขึ้นเพื่อใช้สำหรับการผลิตพืชอื่น ๆ ที่มีศักยภาพในสภาพพื้นที่ทุ่งกุลาร้องไห้	ปรึกษาแลกเปลี่ยนข้อมูลและความคิดเห็นกับบุคลากรของหน่วยงานที่เกียวข้อง	ได้แลกเปลี่ยนความคิดเห็นและปรึกษา ถึงแนวทางการปรับปรุง และขยายการใช้แบบจำลองดังกล่าวกับบุคลากรของสถาบันวิจัยพืชไร่ กระทรวงเกษตรและสหกรณ์ ซึ่งเป็นผู้ที่มีประสบการณ์ ในการใช้แบบจำลอง ร่วมกับระบบสารสนเทศ และระบบประมวลผลอื่น ๆ ในการปฏิบัติการ	สามารถปรับเปลี่ยนเพิ่มเติมเพื่อใช้ประเมินผลผลิตพืชอื่น ๆ ได้ง่าย