9. Allegati

Allegato I Codice Acquisizione Datataker per la Caratterizzazione

```
BEGIN "CARATTERIZZAZIONE"
RS1M RA5M
Y10=0.9865,0.9794 '2-
Y11=0.8377,0.9885 '2+
Y12=0.8625,0.97997 '2*
Y13=0.7519,0.9985 '3-
Y14=0.6108,1.0055 '3+
Y15=0.7854,0.9899 '3*
Y16=0.5229,0.9848 '4-
Y17=0.4719,0.9922 '4+
Y18=0.5127,0.9920 '4*
Y19=0.2280,0.9925 '5-
Y20=0.6807,0.9964 '5+
Y21=0.2239,1.0049 '5*
Y22=0.2020,0.9633 '6-
Y23=0.4868,0.9980 '6+
Y24=0.5084,1.0004 '6*
Y25=0.6472,0.9920 '7-
Y26=0.7286,0.9882 '7+
Y27=0.6017,0.9961 '7*
Y28=0.5767,0.9982 '8-
Y29=0.7484,0.9798 '8+
Y30=0.5822,0.9985 '8*
Y31=0.1928,0.9986 '9-
Y32=0.4951,1.0194 '9+
Y33=0.8287,0.9864 '9*


‘TERMOCOPPIE
2*TT(Y12,"Tai_1~degC",AV)
8*TT(Y30,"Tai_2~degC",AV)
4*TT(Y18,"Tai_3~degC",AV)
4+TT(Y17,"Tout~degC",AV)
3+TT(Y14,"Tsi_floor_1~degC",AV)
2-TT(Y10,"Tsi_floor_2~degC",AV)
6*TT(Y24,"Tsi_Floor_3~degC",AV)
7+TT(Y26,"Tse_Floor~degC",AV)
7-TT(Y25,"Tsi_Door~degC",AV)
5*TT(Y21,"Tse_Door~degC",AV)
8-TT(Y28,"Tsi_W_West~degC",AV)
6-TT(Y22,"Tse_W_West~degC",AV)
3*TT(Y15,"Tsi_W_East~degC",AV)
2+TT(Y11,"Tse_W_East~degC",AV)
```

```
12*V(Y43,"HF_Ceiling~W/m2",FM2,AV)
4-TT(Y16,"Tsi_Ceiling~degC",AV)
9*TT(Y33,"Tse_Ceiling~degC",AV)
5-TT(Y19,"Tsi_SG~degC",AV)
3-TT(Y13,"Tse_SG~degC",AV)
5+TT(Y20,"Tsi_EC1~degC",AV)
8+TT(Y29,"Tse_EC1~degC",AV)
7*TT(Y27,"Tsi_EC2~degC",AV)
6+TT(Y23,"Tse_EC2~degC",AV)
'TERMOFLUSSIMETRI
Y8=0,15.1286"W/m2" 'Pavimento
Y40=0,14.6843"W/m2"  'Finestra
Y42=0,15.9236"W/m2"  'Par_est


10V(Y40,"HF_SG~W/m2",FM2,AV)
11V(Y8,"HF_EC1~W/m2",FM2,AV)
12V(Y42,"HF_EC2~W/m2",FM2,AV)
'PIRANOMETRI
Y1=0,66.6667"W/m2" 'LP02-40820 Vetro Statico
Y2=0,81.3008"W/m2" 'LP02-41273 Vetro EC2
Y3=0,66.666"W/m2"  'LP02-40485 Vetro EC1
Y5=0,69.1563"W/m2" 'LP02-40222 esterno cella verticale
Y6=0,77.0416"W/m2" 'SR12-T1-1117  esterno orizzontale

15V(Y1,"Pyra_In_Vert_SG~W/m2",FM2,AV)
15*V(Y3,"Pyra_In_Vert_EC1~W/m2",FM2,AV)
16V(Y2,"Pyra_In_Vert_EC2~W/m2",FM2,AV)
16*V(Y5,"Pyra_Out_Vert~W/m2",FM2,AV)
14V(Y6,"Pyra_Out_Horiz~W/m2",FM2,AV)

END
LOGON
G
```

Allegato II Codice Acquisizione Datataker per la Calibrazione

```
BEGIN "CALIBRA"
RS1M RB15M
LOGONB
Y10=0.9865,0.9794 '2-
Y11=0.8377,0.9885 '2+
Y12=0.8625,0.97997 '2*
Y13=0.7519,0.9985 '3-
Y14=0.6108,1.0055 '3+
Y15=0.7854,0.9899 '3*
Y16=0.5229,0.9848 '4-
Y17=0.4719,0.9922 '4+
Y18=0.5127,0.9920 '4*
Y19=0.2280,0.9925 '5-
Y20=0.6807,0.9964 '5+
Y21=0.2239,1.0049 '5*
Y22=0.2020,0.9633 '6-
Y23=0.4868,0.9980 '6+
Y24=0.5084,1.0004 '6*
Y25=0.6472,0.9920 '7-
Y26=0.7286,0.9882 '7+
Y27=0.6017,0.9961 '7*
Y28=0.5767,0.9982 '8-
Y29=0.7484,0.9798 '8+
Y30=0.5822,0.9985 '8*
Y31=0.1928,0.9986 '9-
Y32=0.4951,1.0194 '9+
Y33=0.8287,0.9864 '9*

‘TERMOCOPPIE
2*TT(Y12,"Tai_1~degC",AV)
8*TT(Y30,"Tai_2~degC",AV)
4*TT(Y18,"Tai_3~degC",AV)
4+TT(Y17,"Tout~degC",AV)
7+TT(Y26,"Tse_Floor~degC",AV)
3+TT(Y14,"Tsi_floor_1~degC",AV)
2-TT(Y10,"Tsi_floor_2~degC",AV)
6*TT(Y24,"Tsi_Floor_3~degC",AV)
7-TT(Y25,"Tsi_North~degC",AV)
8-TT(Y28,"Tsi_W_West_1~degC",AV)
9-TT(Y31,"Tsi_West_2~degC",AV)
6-TT(Y22,"Tse_W_West~degC",AV)
3*TT(Y15,"Tsi_W_East_2~degC",AV)
9+TT(Y32,"Tsi_East_1~degC",AV)
2+TT(Y11,"Tse_W_East~degC",AV)
5*TT(Y21,"Tsi_Ceiling_1~degC",AV)
4-TT(Y16,"Tsi_Ceiling_2~degC",AV)
9*TT(Y33,"Tse_Ceiling~degC",AV)
```

```
5-TT(Y19,"Tsi_SG~degC",AV)
3-TT(Y13,"Tse_SG~degC",AV)
5+TT(Y20,"Tsi_EC1~degC",AV)
8+TT(Y29,"Tse_EC1~degC",AV)
7*TT(Y27,"Tsi_EC2~degC",AV)
6+TT(Y23,"Tse_EC2~degC",AV)

'TERMOFLUSSIMETRI
Y8=0,15.1286"W/m2" 'Pavimento
Y9=0,15.4799"W/m2" 'Porta
Y40=0,14.6843"W/m2"  'Finestra
Y41=0,15.8479"W/m2"  'Par_Ovest
Y42=0,15.9236"W/m2"  'Par_est
Y43=0,15.4083"W/m2"  'Soff
10V(Y40,"HF_SG~W/m2",FM2,AV)
11V(Y8,"HF_EC1~W/m2",FM2,AV)
10*V(Y41,"HF_EC2~W/m2",FM2,AV)
12*V(Y43,"HF_Floor~W/m2",FM2,AV)
11*V(Y9,"HF_North~W/m2",FM2,AV)
12V(Y42,"HF_East~W/m2",FM2,AV)
'PIRANOMETRI
Y1=0,66.6667"W/m2" 'LP02-40820 Vetro Statico
Y2=0,81.3008"W/m2" 'LP02-41273 Vetro EC2
Y3=0,66.666"W/m2"  'LP02-40485 Vetro EC1
Y5=0,69.1563"W/m2" 'LP02-40222 esterno cella verticale
Y6=0,77.0416"W/m2" 'SR12-T1-1117  esterno orizzontale
Y7=0,62.5"W/m2" 'vicino finestra
Y8=0,63.0517"W/m2" 'centro cella

13V(Y7,"Pyra_A~W/m2",FM2,AV)
13*V(Y8,"Pyra_B~W/m2",FM2,AV)
15V(Y1,"Pyra_In_Vert_SG~W/m2",FM2,AV)
15*V(Y3,"Pyra_In_Vert_EC1~W/m2",FM2,AV)
16V(Y2,"Pyra_In_Vert_EC2~W/m2",FM2,AV)
16*V(Y5,"Pyra_Out_Vert~W/m2",FM2,AV)
14V(Y6,"Pyra_Out_Horiz~W/m2",FM2,AV)
END
```

Allegato III Codice per l'identificazione dei parametri concentrati tramite CTSM-R

```
library(ctsmr)
files <- dir("functions", full.names=TRUE)
for(i in 1:length(files)) source(files[i])
X <- read.csv("input/dati_new.csv",sep=',')
names(X)<-c("t","yTi","Ta","Psw","Pse","Ph","vetri")
X$t<-X$t/60
X$timedate <- asP("2019-01-01 00:00:00)") + X$t *3600
X$Ph<-X$Ph/1000
X$Psw<-X$Psw/1000
X$Pse<-X$Pse/1000

 gg<-14
 time_step<-1
 star<-(gg*time_step*24)
 finish<-star+(time_step*24*7*3)
 gg_test<-31+11
 test_start<-(gg_test*time_step*24)
 test_end<-test_start+(time_step*24*7*3)

  #estimation subset
 Xest<-subset(X,t>star & t<finish)
 #test subset
 Xtest<-subset(X,t>test_start & t<test_end)

 X<-Xest
#creazione modello
model<-ctsm()
model$addSystem(dTi~1/(Ri*Ci)*(Te-
Ti)*dt+1/Ci*Aw*Psw*dt+1/Ci*Ph*dt+exp(p11)*dw1)
 model$addSystem(dTe~1/(Ri*Ce)*(Ti-Te)*dt+1/(Ri*Ce)*(Ta-
Te)*dt+1/Ce*Ae*Pse*dt+exp(p22)*dw2)
#Input
 model$addInput(Ta,Psw,Pse,Ph)
#output
model$addObs(yTi~Ti)
model$setVariance(yTi~  exp(e11))
#set initial value of states for the first time point
Ti<-X$yTi[1]
Te<-(Ti+X$Ta[1])
#set initial value
Rie0<-25.95 #K/kW
Ci0<-0.0174 #kWh/K
Ce0<-0.037 #kWh/K
Aw0<-3.16 #m2
Ae0<-16.13 #m2
kk<-3 #factor
model$setParameter(Ti0=c(init=Ti,Ti-0.5,Ti+0.5))
model$setParameter(Te0=c(init=Te,Te/kk,Te+kk))
```

```
model$setParameter(Ri=c(init=Rie0,Rie0/kk,Rie0*kk),
                     Ci=c(init=Ci0,Ci0/kk,Ci0*kk),
                     Ce=c(init=Ce0,Ce0/kk,Ce0*kk),
                     Aw=c(init=Aw0,Aw0/5,Aw0*kk),
                    Ae=c(init=Ae0,2,Ae0*kk))
model$setParameter( p11 = c(init=1 ,lb=-50 ,ub=10) )
model$setParameter( p22 = c(init=1 ,lb=-50 ,ub=10))
model$setParameter( e11 = c(init=-1 ,lb=-50 ,ub=10))

#fit the model
fit<-model$estimate(data=X)
analyzeFit(fit,  plotACF=TRUE,  newdev=TRUE,  tPer=c("2019-01-22","2019-01-
29"))

#prediction value with TEST dataframe
test<-predict(fit,n.ahead=1,newdata=Xtest)
X<-Xtest
 ## Calculate the residuals
 X$residuals <- X$yTi - test$output$pred$yTi
 X$yTiHat <-test$output$pred$yTi

#plot output/input/residual
xwin<-c(X$timedate[10080],X$timedate[10080])
 yTT<-c(-10,60)
 xsum<-c(X$timedate[20160],X$timedate[20160])
 yPP<-c(-2,2)

 dev.new()
  plotTSBeg(5)
  gridSeq <- seq(asP("2019-01-01"),by="days",len=365)
  plot(X$timedate,X$residuals,type="n",xlab="",ylab="Residual[°C]",
yaxt="n")
  axis(2,pretty(scalerange(X$residuals,0.8)))
  abline(v=gridSeq,h=0,col="grey85",lty=3)
  lines(X$timedate,X$residuals)
  text(X$timedate[5020],1.3,label="Heating",cex=1.4,font=2)
  text(X$timedate[15020],1.3,label="Free running",cex=1.4,font=2)
  text(X$timedate[25020],1.3,label="Cooling",cex=1.4,font=2)
  lines(xwin, yPP,col=c(2),lty=5)
  lines(xsum, yPP,col=c(2),lty=5)
   ##

plot(X$timedate,X$yTi,type="n",xlab="",ylab="Ti°C]",yaxt="n",ylim=range(X
[,c("yTi","yTiHat")])) #
  axis(2,pretty(scalerange(X[c("yTi","yTiHat")],0.2)))
  abline(v=gridSeq,h=10,col="grey85",lty=3)
  lines(X$timedate,X$yTi)
  lines(X$timedate,X$yTiHat,col=2,lty=5)

legend(X$timedate[26000],13,c("Measured","Predicted"),lty=1,col=1:2,box.c
ol="white")#,bg="grey95")
  lines(xwin, yTT,col=c(2),lty=5)
```

```
lines(xsum, yTT,col=c(2),lty=5)
 plot(X$timedate,X$Psw,type="n",xlab="",ylab="Psw [kW]",yaxt="n")
axis(2,pretty(scalerange(X$Psw,0.2)))
abline(v=gridSeq,h=0,col="grey85",lty=3)
lines(X$timedate,X$Psw)
lines(xwin, yPP,col=c(2),lty=5)
lines(xsum, yPP,col=c(2),lty=5)
##
plot(X$timedate,X$Pse,type="n",xlab="",ylab="Pse[kW]",yaxt="n")
axis(2,pretty(scalerange(X$Pse,0.2)))
abline(v=gridSeq,h=0,col="grey85",lty=3)
lines(X$timedate,X$Pse)
lines(xwin, yPP,col=c(2),lty=5)
lines(xsum, yPP,col=c(2),lty=5)
plot(X$timedate,X$Ph,type="n",xlab="",ylab="Ph [kW]",yaxt="n")
axis(2,pretty(scalerange(X$Ph,0.2)))
abline(v=gridSeq,h=0,col="grey85",lty=3)
lines(X$timedate,X$Ph)
lines(xwin, yPP,col=c(2),lty=5)
lines(xsum, yPP,col=c(2),lty=5)
##
plotTSXAxis(X$timedate,format="%Y-%m-%d")
```

Allegato IV Codice per l'implementazione del Controllo 1 (EMS)

```
EnergyManagementSystem:Sensor,
    INC,                     !- Name
    SOUTH WALL,               !- Output:Variable or Output:Meter Index Key
Name
    Surface Outside Face Incident Solar Radiation Rate per Area ;   !-
Output:Variable
EnergyManagementSystem:Actuator,
    EC1,                     !- Name
    EC 1,                    !- Actuated Component Unique Name
    Surface,                 !- Actuated Component Type
   Construction State;      !- Actuated Component Control Type
EnergyManagementSystem:Actuator,
    EC2,                     !- Name
    EC 2,                    !- Actuated Component Unique Name
    Surface,                 !- Actuated Component Type
    Construction State;      !- Actuated Component Control Type
EnergyManagementSystem:Program,
    CONTROL_RAD,             !- Name
    IF (Minute == 60 || Minute == 15 || Minute == 30 || Minute == 45),  !-
Program Line 1
    IF INC <=100,            !- Program Line 2
    SET EC1=HALIO73,         !- A4
    SET EC2=HALIO73,         !- A5
    ELSEIF INC<=200,         !- A6
    SET EC1=HALIO60,         !- A7
    SET EC2=HALIO60,         !- A8
    ELSEIF INC<=300,         !- A9
    SET EC1=HALIO46,         !- A10
    SET EC2=HALIO46,         !- A11
    ELSEIF INC<=400,         !- A12
    SET EC1=HALIO28,         !- A13
    SET EC2=HALIO28,         !- A14
    ELSE,                    !- A15
    SET EC1=HALIO5,          !- A16
    SET EC2=HALIO5,          !- A17
    ENDIF,                   !- A18
    ENDIF;                   !- A19
```

Allegato V Codice per l'implementazione del Controllo 2RB (EMS)

```
EnergyManagementSystem:Sensor,
    Tair,                  !- Name
    Thermal Zone 1,       !- Output:Variable or Output:Meter Index Key Name
    Zone Mean Air Temperature ;  !- Output:Variable or Output:Meter Name
EnergyManagementSystem:Actuator,
    EC1,                      !- Name
    EC 1,                     !- Actuated Component Unique Name
    Surface,                  !- Actuated Component Type
    Construction State;       !- Actuated Component Control Type
EnergyManagementSystem:Actuator,
    EC2,                      !- Name
    EC 2,                     !- Actuated Component Unique Name
    Surface,                  !- Actuated Component Type
    Construction State;       !- Actuated Component Control Type
EnergyManagementSystem:Program,
    CONTROL_TEMP,             !- Name
    SET Too=@Round(Tair),     !- Program Line 1
    IF Minute == 15 || Minute == 30 || Minute == 45 || Minute == 60,  !-
Program Line 2
    IF Too>=26,               !- A4
    SET EC1=HALIO5,           !- A5
    SET EC2=HALIO5,           !- A6
    ELSEIF Too>24,            !- A7
    SET EC1=HALIO28,          !- A8
    SET EC2=HALIO28,          !- A9
    ELSEIF Too>22,            !- A10
    SET EC1=HALIO46,          !- A11
    SET EC2=HALIO46,          !- A12
    ELSEIF Too>20,            !- A13
    SET EC1=HALIO60,          !- A14
    SET EC2=HALIO60,          !- A15
    ELSE,                     !- A16
    SET EC1=HALIO73,          !- A17
    SET EC2=HALIO73,          !- A18
    ENDIF,                    !- A19
    ENDIF;                    !- A20
EnergyManagementSystem:ProgramCallingManager,
    CALLING,                  !- Name
    BeginTimestepBeforePredictor,  !- EnergyPlus Model Calling Point
    CONTROL_TEMP;              !- Program Name 1
```

Allegato VI Codice per l'implementazione del Controllo 2 MB

MATLAB

```
clear all
close all
% Initialize flags
retVal    = 0;
flaWri    = 0;
flaRea    = 0;
simTimWri = 0; %prima 0
simTimRea = 0;
delTim    = 1*60;  % time step 1 min
%parameters
g=[0.2076,0.2746,0.3422,0.4051,0.468];
Ts=1/60;
index=5;

%%parameters R
Ae=3.2;
Aw=7.9e-1;
Cw=1.6e-2;
Ci=4.8e-2;
Ri=14.2;

%matrix
A=[-(1/(Ri*Ci)),1/(Ri*Ci);1/(Ri*Cw),-(2/(Ri*Cw))];
B=[0, Aw/Ci,0,1/Ci;1/(Ri*Cw),0,Ae/Cw,0];

%%%loading meteo data anno
dati=load('anno_rev02.mat');
dati=dati.data;
dati(:,2:4)=dati(:,2:4)./1000;
dati(:,4)=0;

%BCVTB input - output
output= [1 0 2 0 0];
u= [20 0 20]; % Tair time
% Add path to BCVTB matlab libraries
addpath( strcat(getenv('BCVTB_HOME'), '/lib/matlab'));
%addpath('C:\Users\isaia\Documents\BCVTB\lib\matlab');
% Establish the socket connection
sockfd = establishClientSocket('socket.cfg');
if sockfd < 0
  fprintf('Error: Failed to obtain socket file descriptor. sockfd=%d.\n',
...
          sockfd);
  exit;
end
% Loop for simulation time steps.
simulate=true;
while (simulate)
```

```
  % Assign values to be exchanged.
  try
    [retVal, flaRea, simTimRea, u ] = ...
        exchangeDoublesWithSocket(sockfd,  flaWri,  length(u),  simTimWri,
...
                                    output);
  catch ME1
    % exchangeDoublesWithSocket had an error. Terminate the connection
    processError(ME1, sockfd, -1);
    simulate=false;
  end
    % Check return flags
  if (flaRea == 1) % End of simulation
    disp('Matlab  received  end  of  simulation  flag  from  BCVTB.  Exit
simulation.');
    closeIPC(sockfd);
    simulate=false;
  end
  if (retVal < 0) % Error during data exchange
    exception = MException('BCVTB:RuntimeError', ...
                           'exchangeDoublesWithSocket returned value %d',
...
                           retVal);
    processError(exception, sockfd, -1);
    simulate=false;
  end
  if (flaRea > 1) % BCVTB requests termination due to an error.
    exception = MException('BCVTB:RuntimeError', ...
                           ['BCVTB requested MATLAB to terminate by sending
%d\n', ...
                            'Exit simulation.\n'], retVal);
    processError(exception, sockfd, -1);
    simulate=false;
  end
   if (simulate)
    time_pred=15; %predizione ogni 15 minuti
    simtime=u(2); %tempo in secondi
   Ten=u(1); %temperatura interna E+
    Te0=(u(1)+u(3))/2; %Te media tra Tinterna e Tout
    j=simtime/60+2; %tempo in minuti +2 per la sincronizzazione con E+
   if (mod(j,15)==0)&& j~=0 && j<(525585)
     output(1)=index; %actual index

     X0=[Ten;Te0];
  %%STIMA CON LE MATRICI
     for i=0:time_pred-2
     %free running
   I=dati(j+i+1,:)';
    I(4)=0;
    I(2)=I(2)*g(index);

    dT0=A*X0+B*I;
```

```
    X0=X0+dT0*Ts;
    TTT(i+2,1)=X0(1);
    TTT(i+2,2)=X0(2);
 end
   To=X0(1);
   Ttest=round(To);

%%CONTROLLO TEMPERATURA
           if Ttest>=26
           index=1; %Halio 5
            elseif Ttest>24
           index=2;  %Halio 28
            elseif Ttest>22
            index=3; %Halio 46
            elseif Ttest>20
            index=4;  %Halio 60
           else
           index=5;  %Halio 73
        end
end
    end

  % Advance simulation time
    simTimWri = simTimWri + delTim;
end
```

-BCVTB

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE BCVTB-variables SYSTEM "variables.dtd">
<BCVTB-variables>
 <variable source="Ptolemy">
    <EnergyPlus variable="HALIO"/>
  </variable>


  <variable source="EnergyPlus">
    <EnergyPlus name="Thermal Zone 1" type="Zone Mean Air Temperature"/>
  </variable>


  <variable source="EnergyPlus">
    <EnergyPlus   name="ENVIRONMENT"   type="Site   Outdoor   Air   Drybulb
Temperature"/>
  </variable>
```

-**EnergyPlus**

```
ExternalInterface,
    PtolemyServer;           !- Name of External Interface

ExternalInterface:Variable,
    HALIO,                   !- Name
    1;                       !- Initial Value
EnergyManagementSystem:Program,
    CONTROL_BCVTB,           !- Name
    IF HALIO==1,             !- Program Line 1
    SET EC2= HALIO5,         !- Program Line 2
    SET EC1= HALIO5,         !- A4
    SET g=0.087,             !- A5
    ELSEIF HALIO==2,         !- A6
    SET EC2= HALIO28,        !- A7
    SET EC1= HALIO28,        !- A8
    SET g=0.124,             !- A9
    ELSEIF HALIO==3,         !- A10
    SET EC2= HALIO46,        !- A11
    SET EC1= HALIO46,        !- A12
    SET g=0.185,             !- A13
    ELSEIF HALIO==4,         !- A14
    SET EC2= HALIO60,        !- A15
    SET EC1= HALIO60,        !- A16
    SET g=0.284,             !- A17
    ELSE,                    !- A18
    SET EC2= HALIO73,        !- A19
    SET EC1= HALIO73,        !- A20
    SET g=0.468,             !- A21
    ENDIF,                   !- A22
```

Allegato VIII Codice per l'implementazione del Controllo 3 (BCVTB)

- **MATLAB**

```
clear all
close all
% Initialize flags
retVal    = 0;
flaWri    = 0;
flaRea    = 0;
simTimWri = 0; %prima 0
simTimRea = 0;
delTim    = 1*60;  % time step 1 min
%parameters
g=[0.2076,0.2746,0.3422,0.4051,0.468];
Ts=1/60;
index=5;

%%parameters R
Ae=3.2;
Aw=7.9e-1;
Cw=1.6e-2;
Ci=4.8e-2;
Ri=14.2;

%matrix
A=[-(1/(Ri*Ci)),1/(Ri*Ci);1/(Ri*Cw),-(2/(Ri*Cw))];
B=[0, Aw/Ci,0,1/Ci;1/(Ri*Cw),0,Ae/Cw,0];

%%%loading meteo data anno
dati=load('anno_rev02.mat');
dati=dati.data;
dati(:,2:4)=dati(:,2:4)./1000;
dati(:,4)=0;

%BCVTB input - output
output= [1 0 2 0 0];
u= [20 0 20]; % Tair time
% Add path to BCVTB matlab libraries
addpath( strcat(getenv('BCVTB_HOME'), '/lib/matlab'));
%addpath('C:\Users\isaia\Documents\BCVTB\lib\matlab');
% Establish the socket connection
sockfd = establishClientSocket('socket.cfg');
if sockfd < 0
  fprintf('Error: Failed to obtain socket file descriptor. sockfd=%d.\n',
...
          sockfd);
  exit;
end
% Loop for simulation time steps.
simulate=true;
while (simulate)
```

```
  % Assign values to be exchanged.
  try
    [retVal, flaRea, simTimRea, u ] = ...
        exchangeDoublesWithSocket(sockfd,  flaWri,  length(u),  simTimWri,
...
                                    output);
  catch ME1
    % exchangeDoublesWithSocket had an error. Terminate the connection
    processError(ME1, sockfd, -1);
    simulate=false;
  end
    % Check return flags
  if (flaRea == 1) % End of simulation
    disp('Matlab  received  end  of  simulation  flag  from  BCVTB.  Exit
simulation.');
    closeIPC(sockfd);
    simulate=false;
  end
  if (retVal < 0) % Error during data exchange
    exception = MException('BCVTB:RuntimeError', ...
                           'exchangeDoublesWithSocket returned value %d',
...
                           retVal);
    processError(exception, sockfd, -1);
    simulate=false;
  end
  if (flaRea > 1) % BCVTB requests termination due to an error.
    exception = MException('BCVTB:RuntimeError', ...
                           ['BCVTB requested MATLAB to terminate by sending
%d\n', ...
                            'Exit simulation.\n'], retVal);
    processError(exception, sockfd, -1);
    simulate=false;
  end
if (simulate)

    time_pred=15; %predizione ogni 15 minuti
    simtime=u(2); %tempo in secondi
      Ten=u(1); %temperatura interna E+
    Te0=(u(1)+u(3))/2; %Te media tra Tinterna e Tout

    j=simtime/60+1; %tempo in minuti +1 per la sincronizzazione con E+


%
   if (mod(j,15)==0 )&&  j<(525585) )
   if dati(j,2)==0
   id=5;
   end

     output(1)=id; %actual index
```

```
     for idx=1:5

      X0=[Ten;Te0];


%%%STIMA CON LE MATRICI

 for i=0:time_pred-2
     %free running

    I=dati(j+i+1,:)';
    I(4)=0;
    I(2)=I(2)*g(idx);

    dT0=A*X0+B*I;
    X0=X0+dT0*Ts;
    TTT(i+2,1)=X0(1);
    TTT(i+2,2)=X0(2);

 end

   TT(idx)=X0(1);

 end

   if min(TT)>26
       %%cooling --> devo scegliere lo stato
       %%del vetro con Ti più vicina a 26
       [y,id]=min(TT);
       id=max(id); %nel caso ci siano 2 o più id scelgo quello più chiaro
       To=TT(id);
   elseif max(TT)<20
       %%heating--> devo scegliere lo stato
       %%del vetro con Ti più vicino a 20
       [y,id]=max(TT);
       id=max(id); %come cooling
       To=TT(id);
   else

   flag=0;
   t=1;
       %free running
         while flag==0
       %for t=1:length(TT)
           if TT(t)<=26 && TT(t)>=20
               id=t;
                    flag=1;
                    else
                    t=t+1;
           end
```

```
       end

   if t>length(TT)
   t=length(TT);
   end

       To=TT(t);
   end
       end

  % Advance simulation time
    simTimWri = simTimWri + delTim;
end
```

-BCVTB

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE BCVTB-variables SYSTEM "variables.dtd">
<BCVTB-variables>
 <variable source="Ptolemy">
    <EnergyPlus variable="HALIO"/>
  </variable>


  <variable source="EnergyPlus">
    <EnergyPlus name="Thermal Zone 1" type="Zone Mean Air Temperature"/>
  </variable>


  <variable source="EnergyPlus">
    <EnergyPlus   name="ENVIRONMENT"   type="Site   Outdoor   Air   Drybulb
Temperature"/>
  </variable>
```

-EnergyPlus

```
ExternalInterface,
    PtolemyServer;           !- Name of External Interface

ExternalInterface:Variable,
    HALIO,                   !- Name
    1;                       !- Initial Value
EnergyManagementSystem:Program,
    CONTROL_BCVTB,           !- Name
    IF HALIO==1,             !- Program Line 1
    SET EC2= HALIO5,         !- Program Line 2
    SET EC1= HALIO5,         !- A4
    SET g=0.087,             !- A5
    ELSEIF HALIO==2,         !- A6
```

```
SET EC2= HALIO28,        !- A7
SET EC1= HALIO28,        !- A8
SET g=0.124,             !- A9
ELSEIF HALIO==3,         !- A10
SET EC2= HALIO46,        !- A11
SET EC1= HALIO46,        !- A12
SET g=0.185,             !- A13
ELSEIF HALIO==4,         !- A14
SET EC2= HALIO60,        !- A15
SET EC1= HALIO60,        !- A16
SET g=0.284,             !- A17
ELSE,                    !- A18
SET EC2= HALIO73,        !- A19
SET EC1= HALIO73,        !- A20
SET g=0.468,             !- A21
ENDIF,                   !- A22
```

Allegato IX Codice per l'implementazione del Controllo 3bis (BCVTB)

- **MATLAB**

```
clear all
close all
% Initialize flags
retVal    = 0;
flaWri    = 0;
flaRea    = 0;
simTimWri = 0; %prima 0
simTimRea = 0;
delTim    = 1*60;  % time step 1 min
%parameters
g=[0.2076,0.2746,0.3422,0.4051,0.468];
Ts=1/60;
index=5;

%%parameters R
Ae=3.2;
Aw=7.9e-1;
Cw=1.6e-2;
Ci=4.8e-2;
Ri=14.2;

%matrix
A=[-(1/(Ri*Ci)),1/(Ri*Ci);1/(Ri*Cw),-(2/(Ri*Cw))];
B=[0, Aw/Ci,0,1/Ci;1/(Ri*Cw),0,Ae/Cw,0];

%%%loading meteo data anno
dati=load('anno_rev02.mat');
dati=dati.data;
dati(:,2:4)=dati(:,2:4)./1000;
dati(:,4)=0;

%BCVTB input - output
output= [1 0 2 0 0];
u= [20 0 20]; % Tair time
% Add path to BCVTB matlab libraries
addpath( strcat(getenv('BCVTB_HOME'), '/lib/matlab'));
%addpath('C:\Users\isaia\Documents\BCVTB\lib\matlab');
% Establish the socket connection
sockfd = establishClientSocket('socket.cfg');
if sockfd < 0
  fprintf('Error: Failed to obtain socket file descriptor. sockfd=%d.\n',
...
          sockfd);
  exit;
end
% Loop for simulation time steps.
simulate=true;
while (simulate)
```

```
  % Assign values to be exchanged.
  try
    [retVal, flaRea, simTimRea, u ] = ...
        exchangeDoublesWithSocket(sockfd,  flaWri,  length(u),  simTimWri,
...
                                    output);
  catch ME1
    % exchangeDoublesWithSocket had an error. Terminate the connection
    processError(ME1, sockfd, -1);
    simulate=false;
  end
    % Check return flags
  if (flaRea == 1) % End of simulation
    disp('Matlab  received  end  of  simulation  flag  from  BCVTB.  Exit
simulation.');
    closeIPC(sockfd);
    simulate=false;
  end
  if (retVal < 0) % Error during data exchange
    exception = MException('BCVTB:RuntimeError', ...
                           'exchangeDoublesWithSocket returned value %d',
...
                           retVal);
    processError(exception, sockfd, -1);
    simulate=false;
  end
  if (flaRea > 1) % BCVTB requests termination due to an error.
    exception = MException('BCVTB:RuntimeError', ...
                           ['BCVTB requested MATLAB to terminate by sending
%d\n', ...
                            'Exit simulation.\n'], retVal);
    processError(exception, sockfd, -1);
    simulate=false;
  end
if (simulate)

    time_pred=30; %predizione ogni 30 minuti
    simtime=u(2); %tempo in secondi
      Ten=u(1); %temperatura interna E+
    Te0=(u(1)+u(3))/2; %Te media tra Tinterna e Tout

    j=simtime/60+1; %tempo in minuti +1 per la sincronizzazione con E+


%
   if (mod(j,15)==0) &&  j<(525585))
   if dati(j,2)==0
   id=5;
   end

     output(1)=id; %actual index
```

```
     for idx=1:5

      X0=[Ten;Te0];


%%%STIMA CON LE MATRICI

 for i=0:time_pred-2
     %free running

    I=dati(j+i+1,:)';
    I(4)=0;
    I(2)=I(2)*g(idx);

    dT0=A*X0+B*I;
    X0=X0+dT0*Ts;
    TTT(i+2,1)=X0(1);
    TTT(i+2,2)=X0(2);

 end

   T15(idx)=TTT(15,1);
   T30(idx)=TTT(30,1);
 end

   if min(TT)>26
       %%cooling --> devo scegliere lo stato
       %%del vetro con Ti più vicina a 26
       [y,id]=min(TT);
       id=max(id); %nel caso ci siano 2 o più id scelgo quello più chiaro
       To=TT(id);
   elseif max(TT)<20
       %%heating--> devo scegliere lo stato
       %%del vetro con Ti più vicino a 20
       [y,id]=max(TT);
       id=max(id); %come cooling
       To=TT(id);
   else

%free running
        for t=1:length(T15)
          if T15(t)<=26 && T15(t)>=20
            id=t;

          end

             if T30(id)>26
             id=1;
          elseif T30(id)>25
             id=2;
```

```
            elseif T30(id)>20 && T30(id)<21
            id=4;
            elseif T30(id)<20
            id=5;
            end


   end
       end

  % Advance simulation time
    simTimWri = simTimWri + delTim;
end
```

-BCVTB

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE BCVTB-variables SYSTEM "variables.dtd">
<BCVTB-variables>
 <variable source="Ptolemy">
    <EnergyPlus variable="HALIO"/>
  </variable>


  <variable source="EnergyPlus">
    <EnergyPlus name="Thermal Zone 1" type="Zone Mean Air Temperature"/>
  </variable>


  <variable source="EnergyPlus">
    <EnergyPlus   name="ENVIRONMENT"   type="Site   Outdoor   Air   Drybulb
Temperature"/>
  </variable>
```

-EnergyPlus

```
ExternalInterface,
    PtolemyServer;           !- Name of External Interface

ExternalInterface:Variable,
    HALIO,                   !- Name
    1;                       !- Initial Value
EnergyManagementSystem:Program,
    CONTROL_BCVTB,           !- Name
    IF HALIO==1,             !- Program Line 1
    SET EC2= HALIO5,         !- Program Line 2
    SET EC1= HALIO5,         !- A4
    SET g=0.087,             !- A5
    ELSEIF HALIO==2,         !- A6
```

```
SET EC2= HALIO28,        !- A7
SET EC1= HALIO28,        !- A8
SET g=0.124,             !- A9
ELSEIF HALIO==3,         !- A10
SET EC2= HALIO46,        !- A11
SET EC1= HALIO46,        !- A12
SET g=0.185,             !- A13
ELSEIF HALIO==4,         !- A14
SET EC2= HALIO60,        !- A15
SET EC1= HALIO60,        !- A16
SET g=0.284,             !- A17
ELSE,                    !- A18
SET EC2= HALIO73,        !- A19
SET EC1= HALIO73,        !- A20
SET g=0.468,             !- A21
ENDIF,                   !- A22
```