Model-Driven Design using XJ Technologies AnyLogic

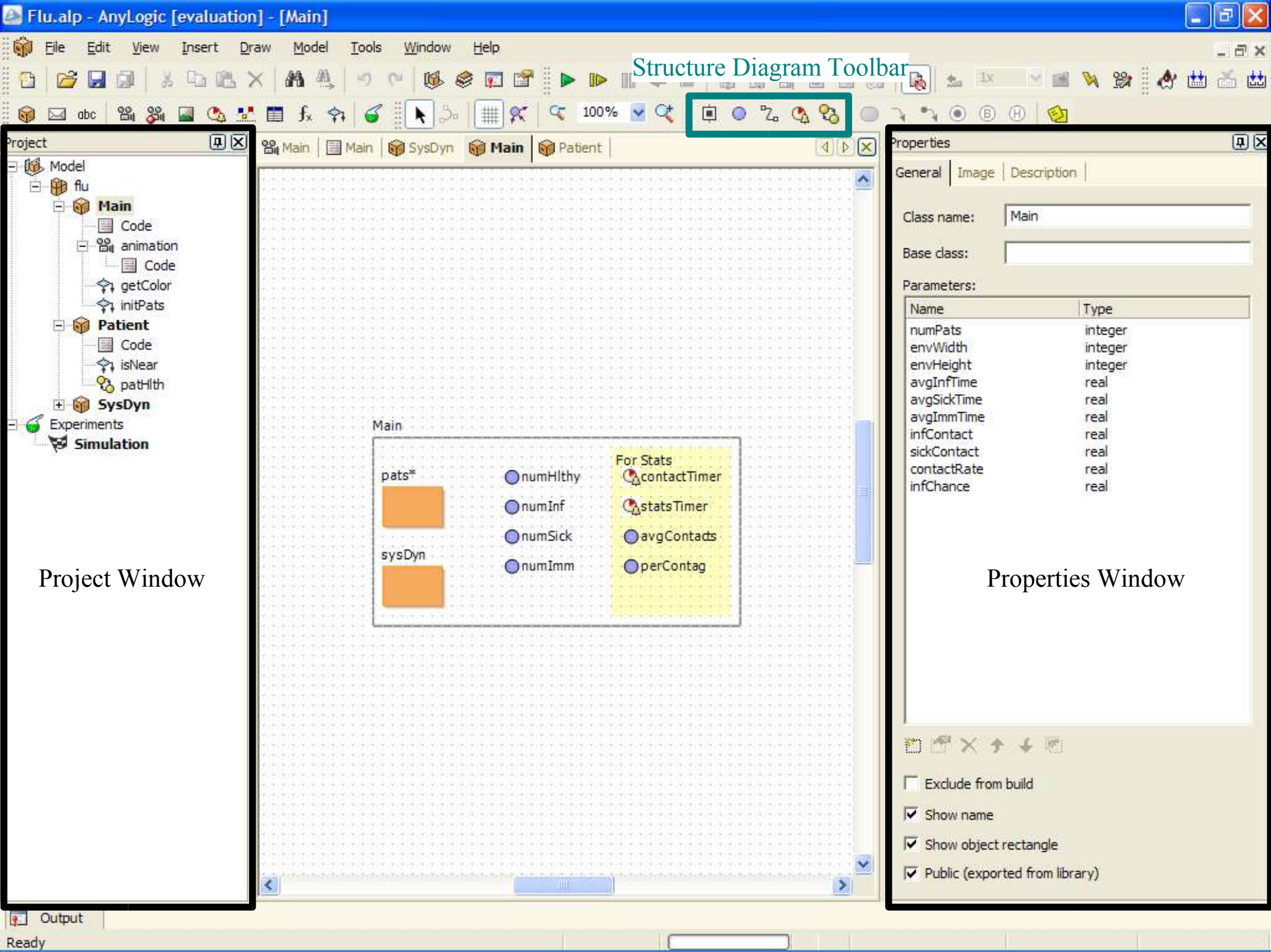
Riandi Wiguna
MSDL
September, 2005

Overview

1. Introduction to AnyLogic
2. Basic Usage of AnyLogic
3. AnyLogic Features
4. Example: Answering Machine
5. Personal Experiences using AnyLogic
6. Conclusion

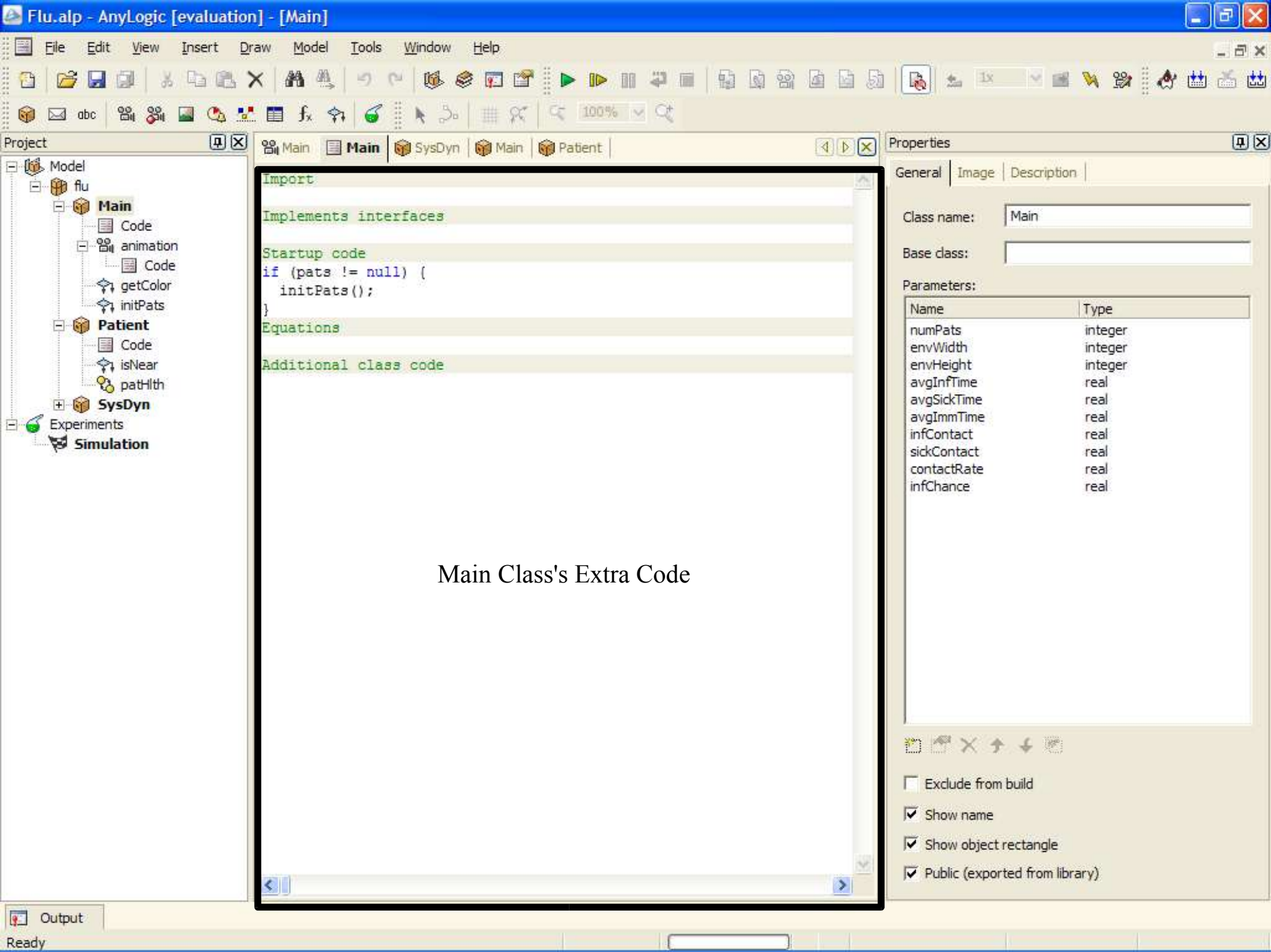
Introduction to AnyLogic

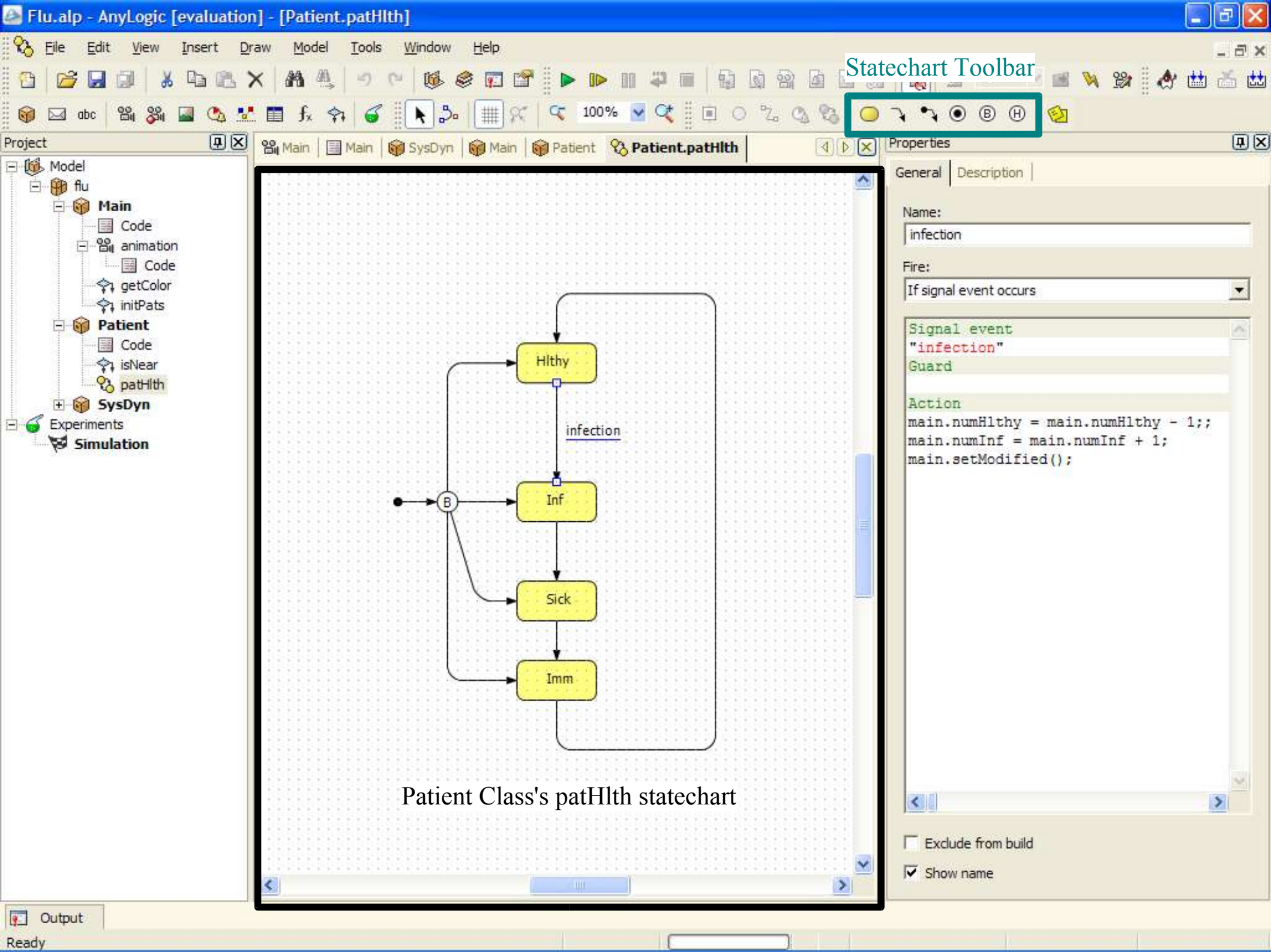
- AnyLogic is software for model-based design
- Users define systems with
  - AnyLogic structure diagrams
  - AnyLogic statecharts
  - mathematical equations
  - Java code
- Users can use AnyLogic to create
  - animations of their running systems
  - graphical user interfaces for their systems
  - live-updating charts that graph data as their systems run



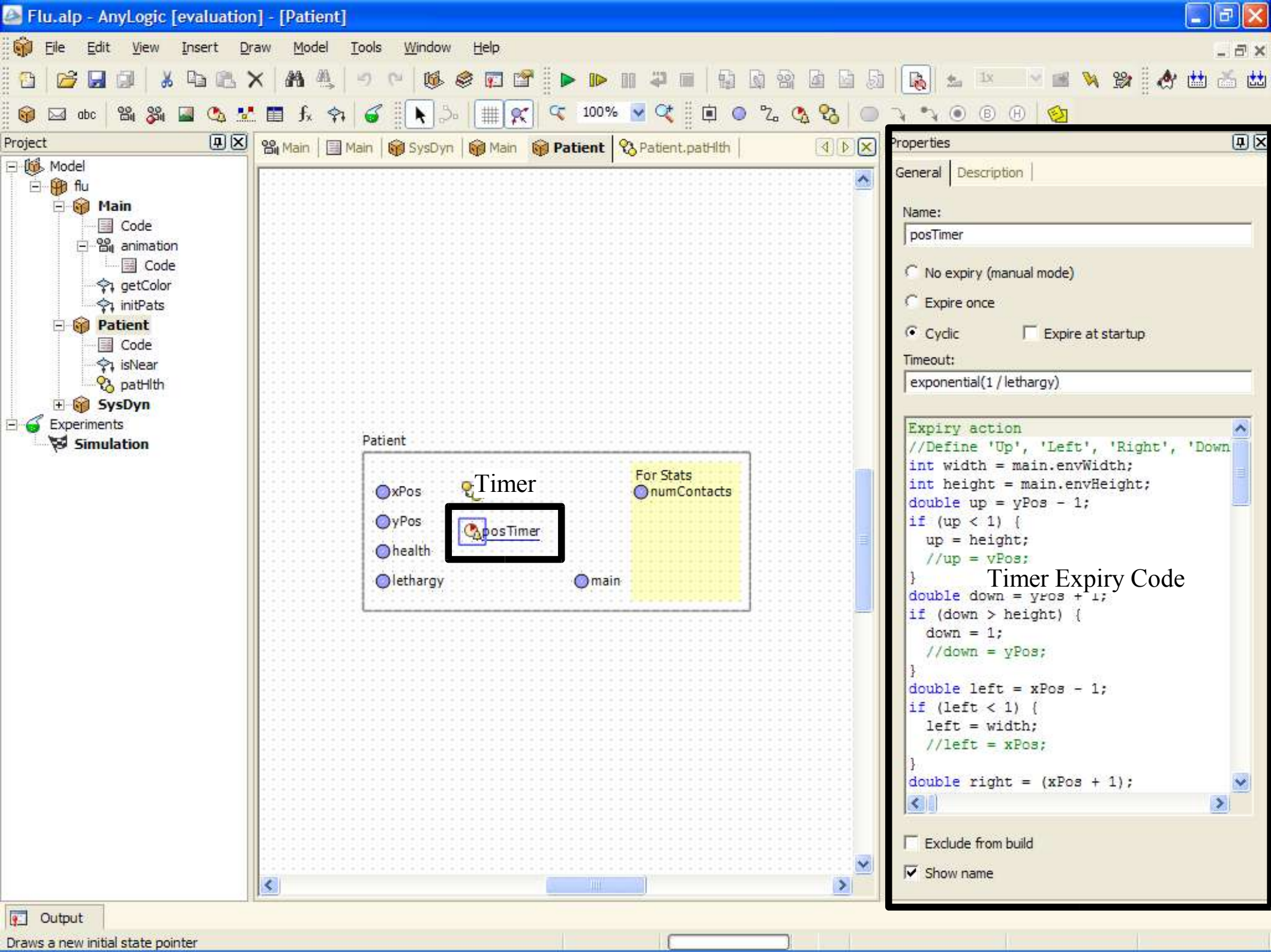
Project Window

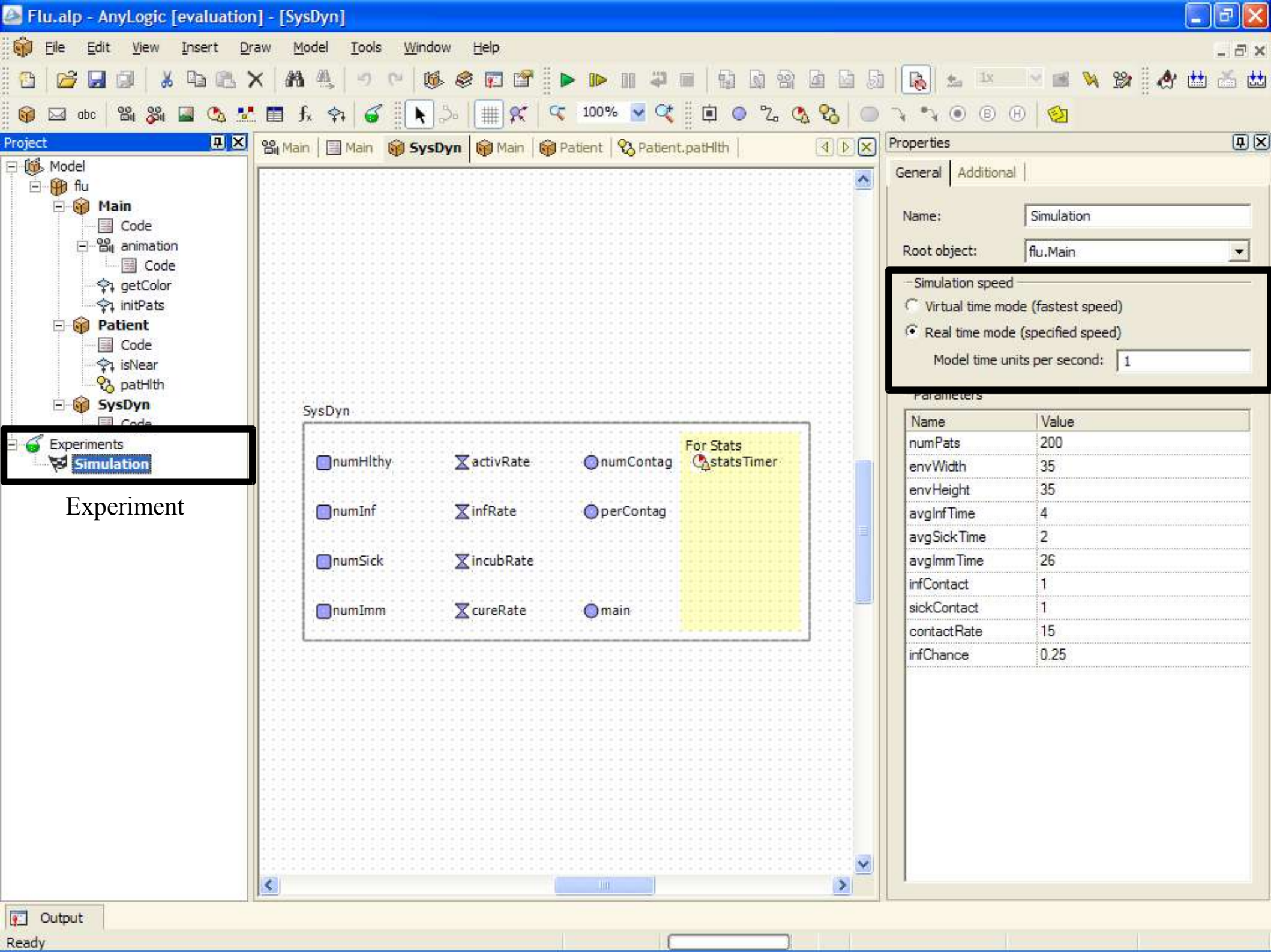
Properties Window


Flu.alp - AnyLogic [evaluation] - [Main]
File Edit View Insert Draw Model Tools Window Help
Project
Model
flu
Main
Code
animation
Code
getColor
initPats
Patient
Code
isNear
patHlth
SysDyn
Experiments
Simulation
Main Main SysDyn Main Patient
Import
Implements interfaces
Startup code
if (pats != null) {
initPats();
}
Equations
Additional class code
Main Class's Extra Code
Properties
General Image Description
Class name: Main
Base class:
Parameters:
Name Type
numPats integer
envWidth integer
envHeight integer
avgInfTime real
avgSickTime real
avgImmTime real
infContact real
sickContact real
contactRate real
infChance real
Exclude from build
Show name
Show object rectangle
Public (exported from library)
Output
Ready

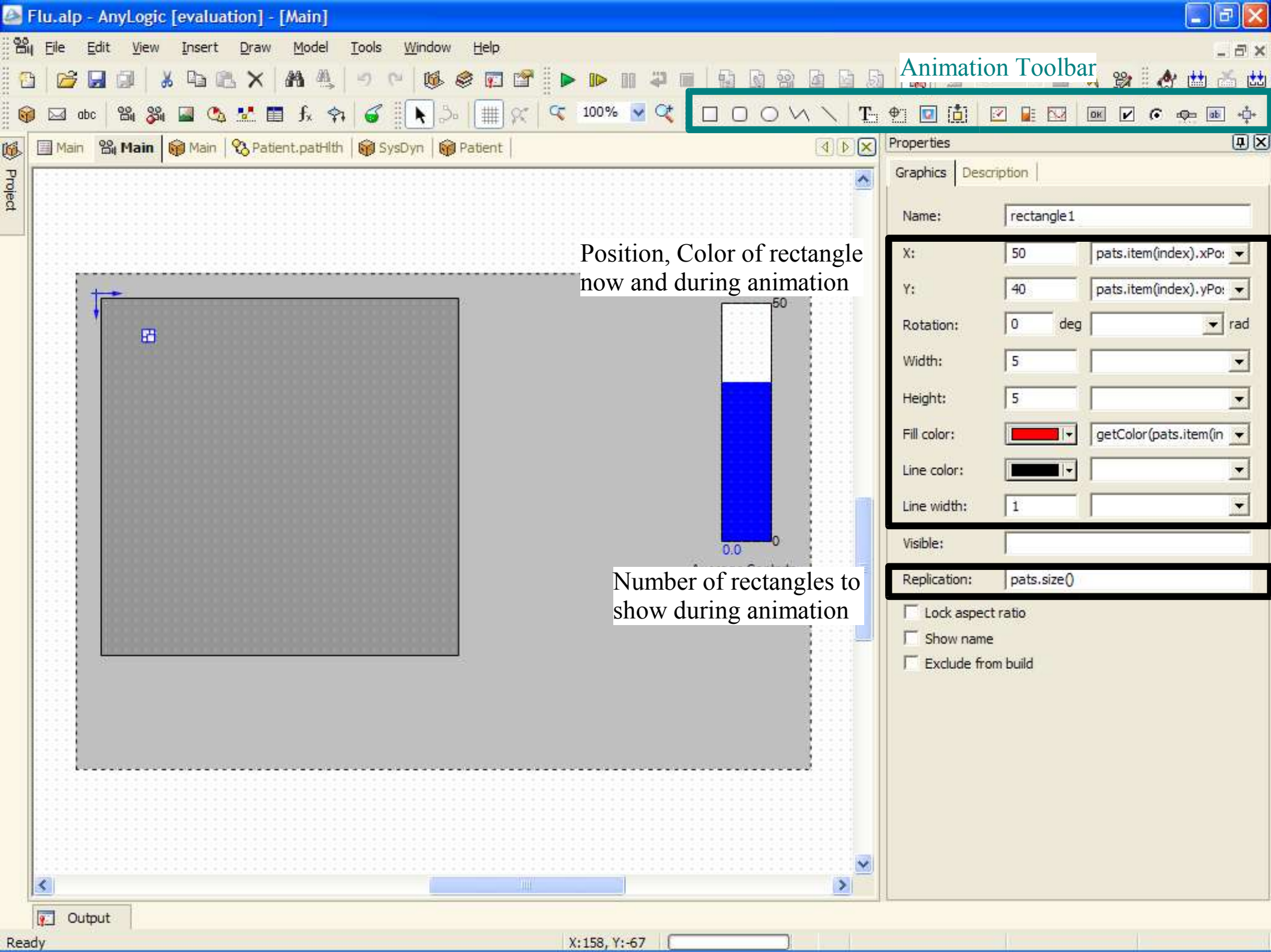


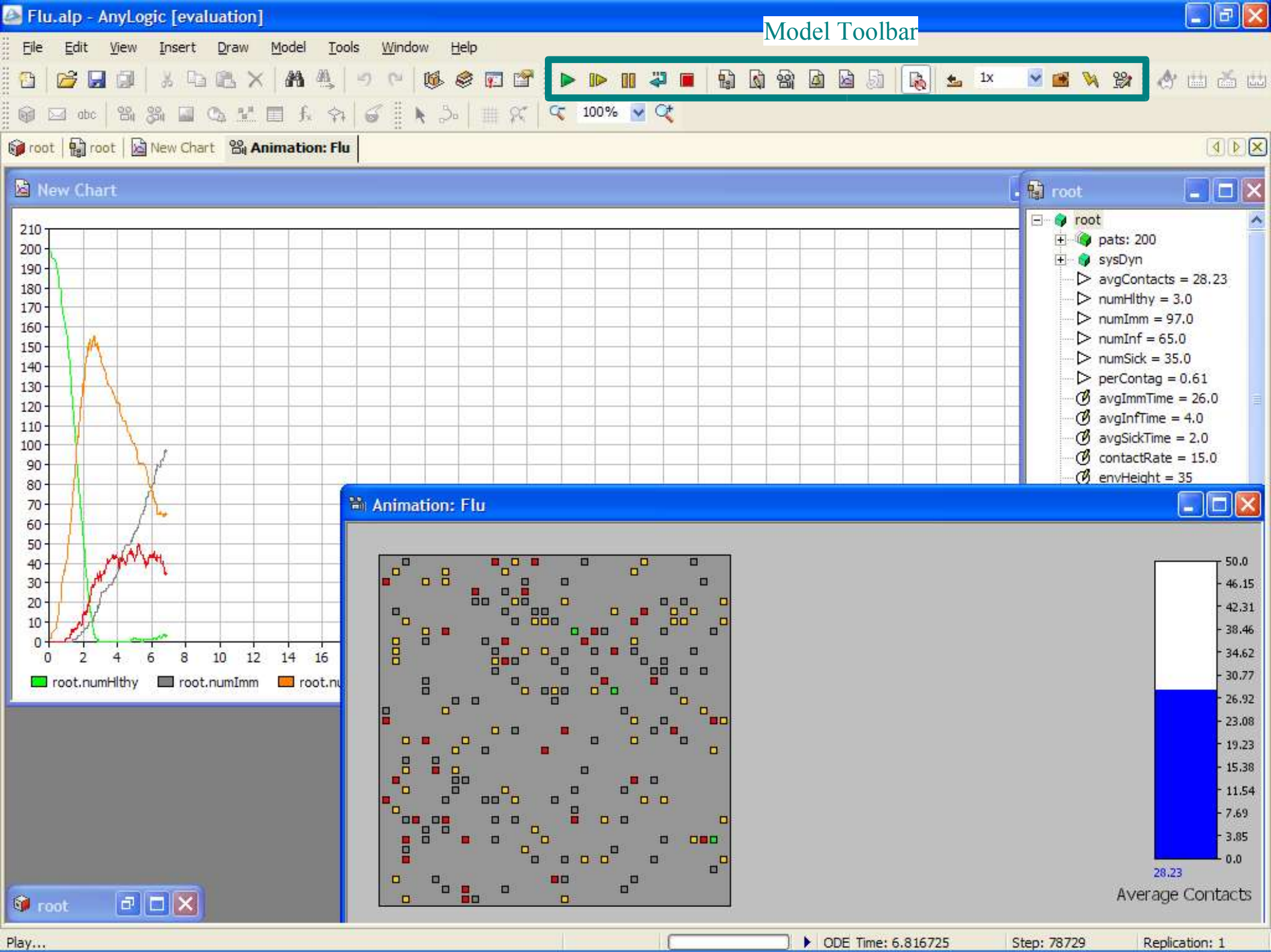
Patient Class's patHlth statechart


Flu.alp - AnyLogic [evaluation] - [Patient]
File Edit View Insert Draw Model Tools Window Help
Project
Model
flu
Main
Code
animation
Code
getColor
initPats
Patient
Code
isNear
patHlth
SysDyn
Experiments
Simulation
Main Main SysDyn Main Patient Patient.patHlth
Patient
xPos
yPos
health
lethargy
Timer
posTimer
main
For Stats
numContacts
Properties
General Description
Name:
posTimer
No expiry (manual mode)
Expire once
Cyclic
Expire at startup
Timeout:
exponential(1 / lethargy)
Expiry action
//Define 'Up', 'Left', 'Right', 'Down
int width = main.envWidth;
int height = main.envHeight;
double up = yPos - 1;
if (up < 1) {
up = height;
//up = yPos;
}
double down = yPos + 1;
if (down > height) {
down = 1;
//down = yPos;
}
double left = xPos - 1;
if (left < 1) {
left = width;
//left = xPos;
}
double right = (xPos + 1);
Timer Expiry Code
Exclude from build
Show name
Output
Draws a new initial state pointer


Flu.alp - AnyLogic [evaluation] - [SysDyn]
File Edit View Insert Draw Model Tools Window Help
Project
Model
flu
Main
Code
animation
Code
getColor
initPats
Patient
Code
isNear
patHlth
SysDyn
Code
Experiments
Simulation
Main Main SysDyn Main Patient Patient.patHlth
SysDyn
numHlthy
activRate
numContag
For Stats
statsTimer
numInf
infRate
perContag
numSick
incubRate
numImm
cureRate
main
Properties
General Additional
Name: Simulation
Root object: flu.Main
Simulation speed
Virtual time mode (fastest speed)
Real time mode (specified speed)
Model time units per second: 1
Parameters
Name Value
numPats 200
envWidth 35
envHeight 35
avgInfTime 4
avgSickTime 2
avgImmTime 26
infContact 1
sickContact 1
contactRate 15
infChance 0.25
Output
Ready

Experiment


Flu.alp - AnyLogic [evaluation] - [Main]
File Edit View Insert Draw Model Tools Window Help
Animation Toolbar
Main Main Main Patient.patHlth SysDyn Patient
Project
Position, Color of rectangle now and during animation
50
0
0.0
Number of rectangles to show during animation
Properties
Graphics Description
Name: rectangle1
X: 50 pats.item(index).xPo
Y: 40 pats.item(index).yPo
Rotation: 0 deg rad
Width: 5
Height: 5
Fill color: getColor(pats.item(in
Line color:
Line width: 1
Visible:
Replication: pats.size()
Lock aspect ratio
Show name
Exclude from build
Output
Ready
X:158, Y:-67


Flu.alp - AnyLogic [evaluation]
Model Toolbar
File Edit View Insert Draw Model Tools Window Help
root root New Chart Animation: Flu
New Chart
210 200 190 180 170 160 150 140 130 120 110 100 90 80 70 60 50 40 30 20 10 0
0 2 4 6 8 10 12 14 16
root.numHlthy root.numImm root.nu
root
root
pats: 200
sysDyn
avgContacts = 28.23
numHlthy = 3.0
numImm = 97.0
numInf = 65.0
numSick = 35.0
perContag = 0.61
avgImmTime = 26.0
avgInfTime = 4.0
avgSickTime = 2.0
contactRate = 15.0
envHeight = 35
Animation: Flu
50.0 46.15 42.31 38.46 34.62 30.77 26.92 23.08 19.23 15.38 11.54 7.69 3.85 0.0
28.23
Average Contacts
root
Play...
ODE Time: 6.816725
Step: 78729
Replication: 1

Introduction to AnyLogic

Demonstration

Basic Usage of AnyLogic

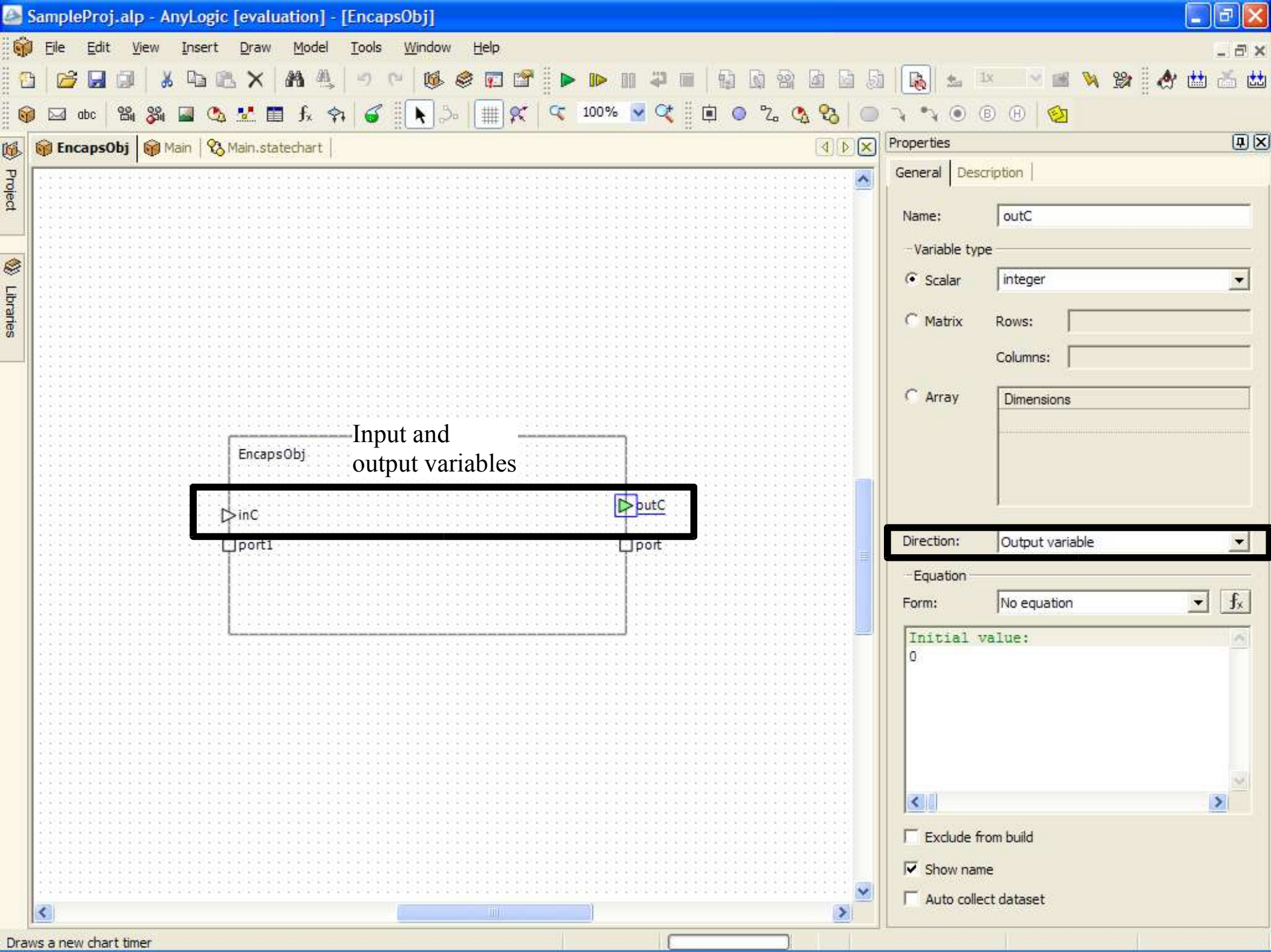
1. Create structure diagrams for classes
2. (Optional) Create statecharts for classes
3. (Optional) Create animations for classes
4. Create an experiment
5. Build project and run experiment

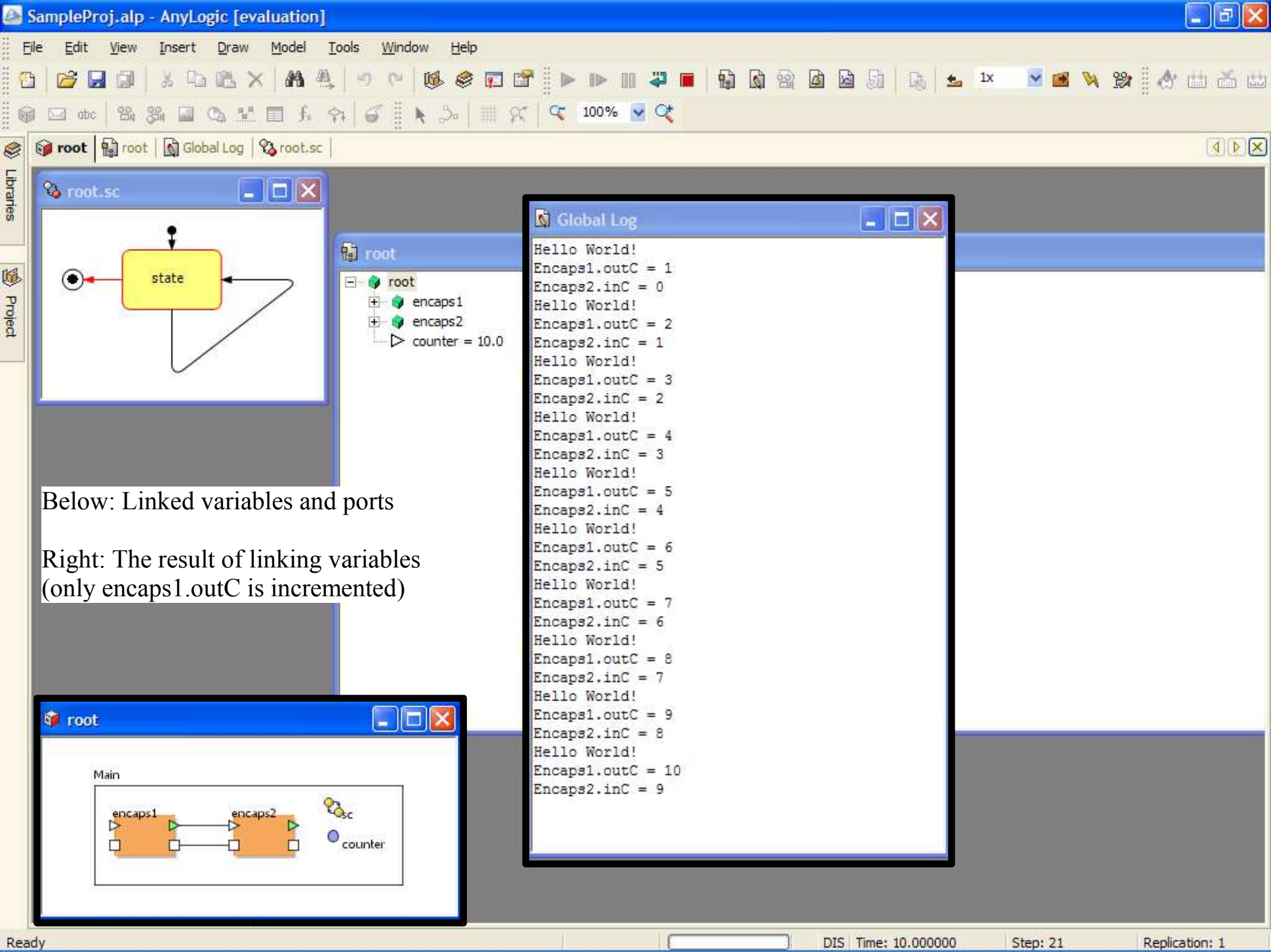
AnyLogic Features

- Manuals, Tutorials, and Examples
  - General AnyLogic User Manual
  - Library usage tutorials
  - Agent-based modeling tutorial
  - System Dynamics tutorial
  - Class Reference API
  - Examples
    - Agent-based modeling
    - Business
    - Ecosystem Dynamics
    - Pedestrian Dynamics
    - Traffic
    - etc.

AnyLogic Features

- Structure Diagrams
  - Variables can be real, integer, or boolean; Parameters can be of any Java class
  - Parameter propagation, variable linking
  - Ports send message objects between active objects
  - An object's setModified() function specifies that it has been altered
- Simulations
  - Virtual Time Mode, Real Time Mode (Choice of mode may result in different output)
  - 3D animation available


SampleProj.alp - AnyLogic [evaluation] - [EncapsObj]
File Edit View Insert Draw Model Tools Window Help
EncapsObj
Main
Main.statechart
Project
Libraries
Input and
output variables
EncapsObj
inC
outC
port1
port
Properties
General
Description
Name:
outC
Variable type
Scalar
integer
Matrix
Rows:
Columns:
Array
Dimensions
Direction:
Output variable
Equation
Form:
No equation
Initial value:
0
Exclude from build
Show name
Auto collect dataset
Draws a new chart timer


SampleProj.alp - AnyLogic [evaluation]
File Edit View Insert Draw Model Tools Window Help
root root Global Log root.sc
Libraries
Project
root.sc
state
root
root
encaps1
encaps2
counter = 10.0
Global Log
Hello World!
Encaps1.outC = 1
Encaps2.inC = 0
Hello World!
Encaps1.outC = 2
Encaps2.inC = 1
Hello World!
Encaps1.outC = 3
Encaps2.inC = 2
Hello World!
Encaps1.outC = 4
Encaps2.inC = 3
Hello World!
Encaps1.outC = 5
Encaps2.inC = 4
Hello World!
Encaps1.outC = 6
Encaps2.inC = 5
Hello World!
Encaps1.outC = 7
Encaps2.inC = 6
Hello World!
Encaps1.outC = 8
Encaps2.inC = 7
Hello World!
Encaps1.outC = 9
Encaps2.inC = 8
Hello World!
Encaps1.outC = 10
Encaps2.inC = 9
Below: Linked variables and ports
Right: The result of linking variables
(only encaps1.outC is incremented)
root
Main
encaps1
encaps2
sc
counter
Ready
DIS Time: 10.000000
Step: 21
Replication: 1

AnyLogic Features

- Statecharts
  - If guard condition is True, transition is taken
    - immediately
    - after specified timeout
    - on events
    - on 'change events' (boolean expression is True)
    - at specified rates
  - Inner transitions taken over outer transitions, unlike Statemate statechart semantics
  - History (deep, shallow), Conditional Branches
  - No orthogonal components
  - Classes may have more than one statechart

AnyLogic Features

- Code
  - Full API included (Help -> Class Reference)
  - Programming code is in Java
  - Array structures are replicated active objects
    - Get Item: arr.item(index)
    - Add Item: setup_arr(obj, index)
    - Delete Item: dispose_arr(obj)
  - Users can sample from probability distributions
    - bernoulli(p, r)
    - binomial(n)
    - exponential(lambda)
    - cauchy(lambda, theta)
    - etc.

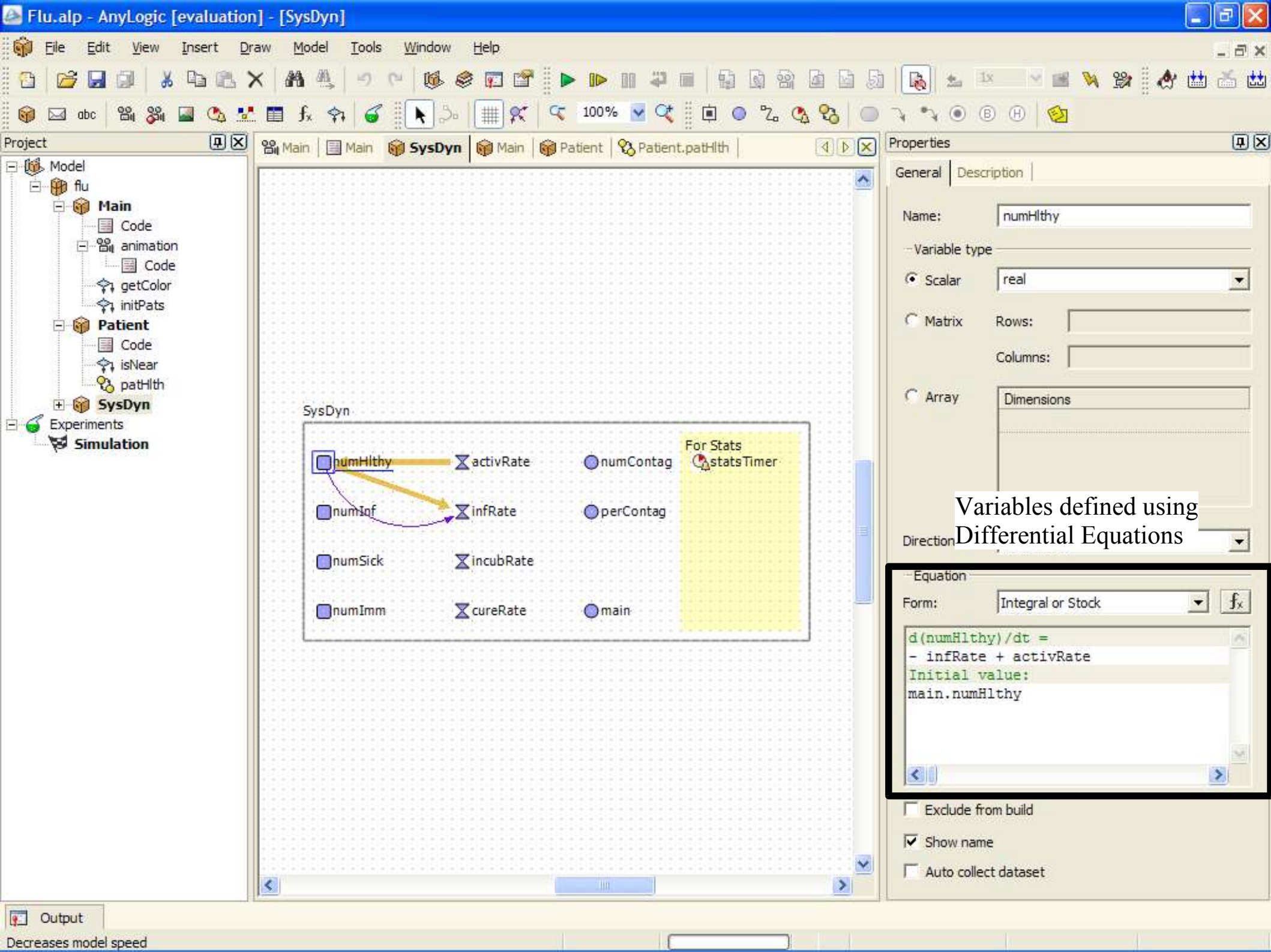
AnyLogic Features

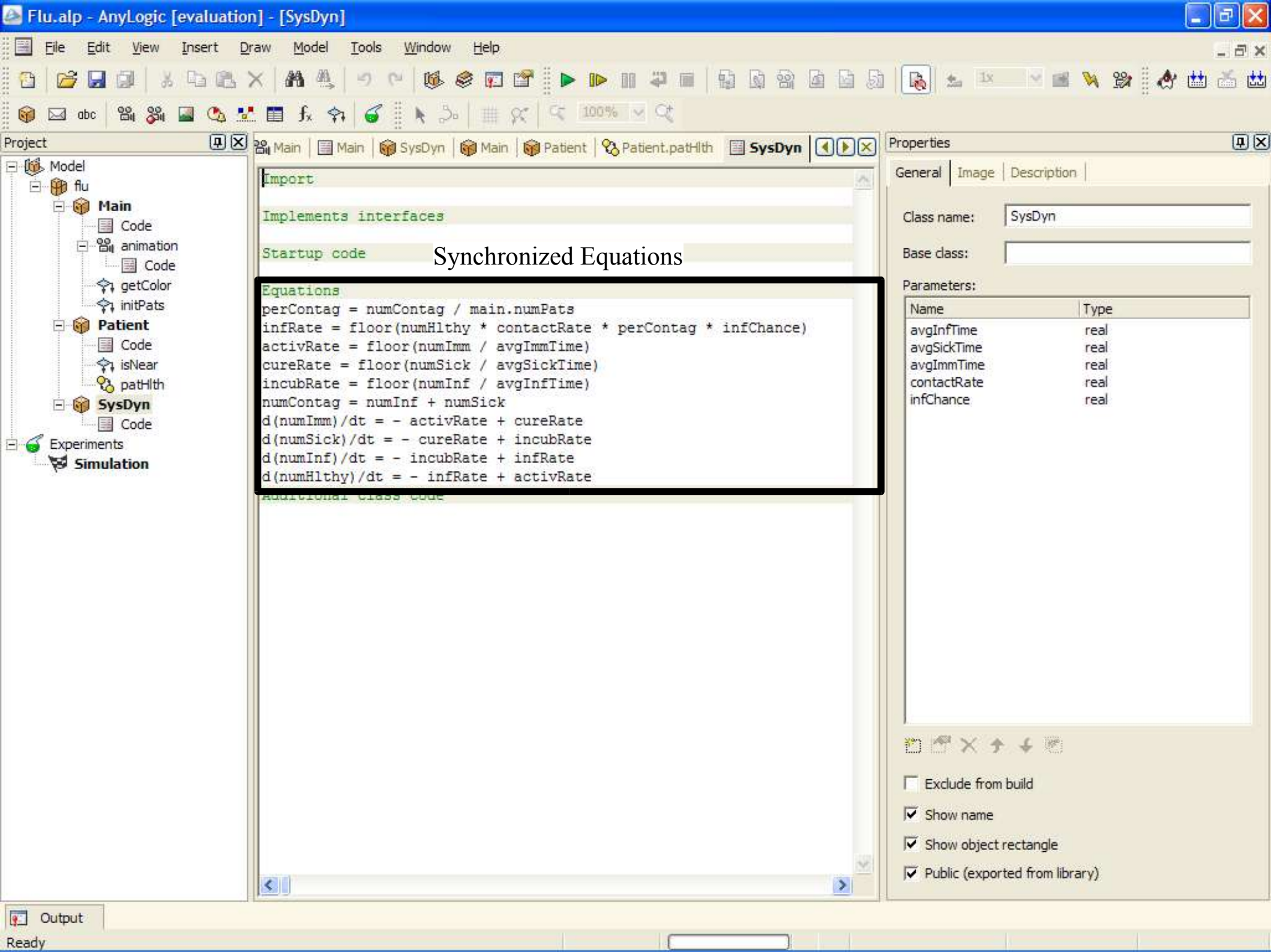
- Users can instantiate encapsulated objects
  - Create Object: setup_myObj(myObj)
  - Destroy Object: dispose_myObj(myObj), and then unregister(myObj)
- Users can control active object creation
  - Constructors
  - onCreate(), before statechart initialized, define in “Additional Class code”
  - “Startup Code”
- Users can control active object destruction
  - cleanup(), define in “Additional class code”
  - onDestroy(), after all structure diagram elements destroyed, destroy manually created encapsulated objects

AnyLogic Features

- Users can observe and control running system
  - Start New Thread: startThread()
  - Get Current Simulation Time: getTime()
- Users can run systems outside of AnyLogic
  - java -classpath xjanylogic5engine.jar; myLib.jar;...myLibN.jar; package.root
  - Use AnyLogic to generate Java applet

- Miscellaneous
  - Automatically handles algebraic loops in structure diagrams
  - Has support for matrices/arrays, lookup tables, enumerations, system dynamics/diff. equations
  - Can generate HTML reports of models

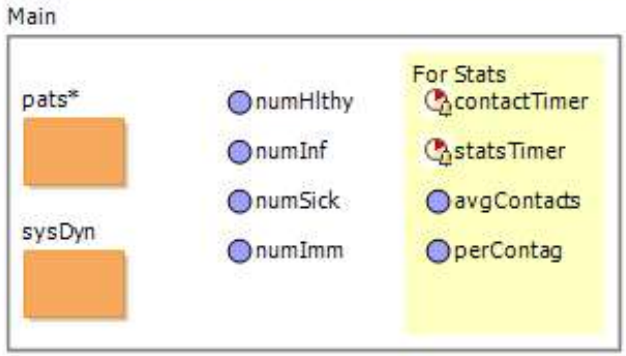

Flu.alp - AnyLogic [evaluation] - [SysDyn]
File Edit View Insert Draw Model Tools Window Help
Project
Model
flu
Main
Code
animation
Code
getColor
initPats
Patient
Code
isNear
patHlth
SysDyn
Experiments
Simulation
Main Main SysDyn Main Patient Patient.patHlth
SysDyn
numHlthy
activRate
numContag
For Stats
statsTimer
numInf
infRate
perContag
numSick
incubRate
numImm
cureRate
main
Properties
General Description
Name: numHlthy
Variable type
Scalar real
Matrix Rows:
Columns:
Array Dimensions
Direction
Variables defined using Differential Equations
Equation
Form: Integral or Stock
d(numHlthy)/dt =
- infRate + activRate
Initial value:
main.numHlthy
Exclude from build
Show name
Auto collect dataset
Output
Decreases model speed


Flu.alp - AnyLogic [evaluation] - [SysDyn]
File Edit View Insert Draw Model Tools Window Help
Project
Model
flu
Main
Code
animation
Code
getColor
initPats
Patient
Code
isNear
patHlth
SysDyn
Code
Experiments
Simulation
Main Main SysDyn Main Patient Patient.patHlth SysDyn
Import
Implements interfaces
Startup code
Synchronized Equations
Equations
perContag = numContag / main.numPats
infRate = floor(numHlthy * contactRate * perContag * infChance)
activRate = floor(numImm / avgImmTime)
cureRate = floor(numSick / avgSickTime)
incubRate = floor(numInf / avgInfTime)
numContag = numInf + numSick
d(numImm)/dt = - activRate + cureRate
d(numSick)/dt = - cureRate + incubRate
d(numInf)/dt = - incubRate + infRate
d(numHlthy)/dt = - infRate + activRate
Properties
General Image Description
Class name: SysDyn
Base class:
Parameters:
Name Type
avgInfTime real
avgSickTime real
avgImmTime real
contactRate real
infChance real
Exclude from build
Show name
Show object rectangle
Public (exported from library)
Output
Ready

Flu - Mozilla Firefox

File Edit View Go Bookmarks Tools Help

file:///C:/Documents%20and%20Settings/Riandi/My%20Documents/My%20Models/Flu/Flu.htm

Picture	
ChartTimer	statsTimer
Timeout	0.25
Expire At Startup	No
Expiry Action	traceln(numHlthy + "\t" + numInf + "\t" + numSick + "\t" + numImm); //newFunc();
ChartTimer	contactTimer
Timeout	1.0
Expire At Startup	No

Expiry Action	int sumContacts = 0;
	for (int x = 0; x < numPats; x++) {
	sumContacts = sumContacts + pats.item(x).numContacts;
	pats.item(x).numContacts = 0;
	}
	avgContacts = ((double) sumContacts) / numPats;
	perContag = ((double) (numInf + numSick)) / numPats;
	double time = getTime();
	/*if ((numHlthy == numPats) && (time >= 8) && (time <= 9)) {
	pats.item(0).health = 1;

Example of Generated Report

Done

Example: Answering Machine

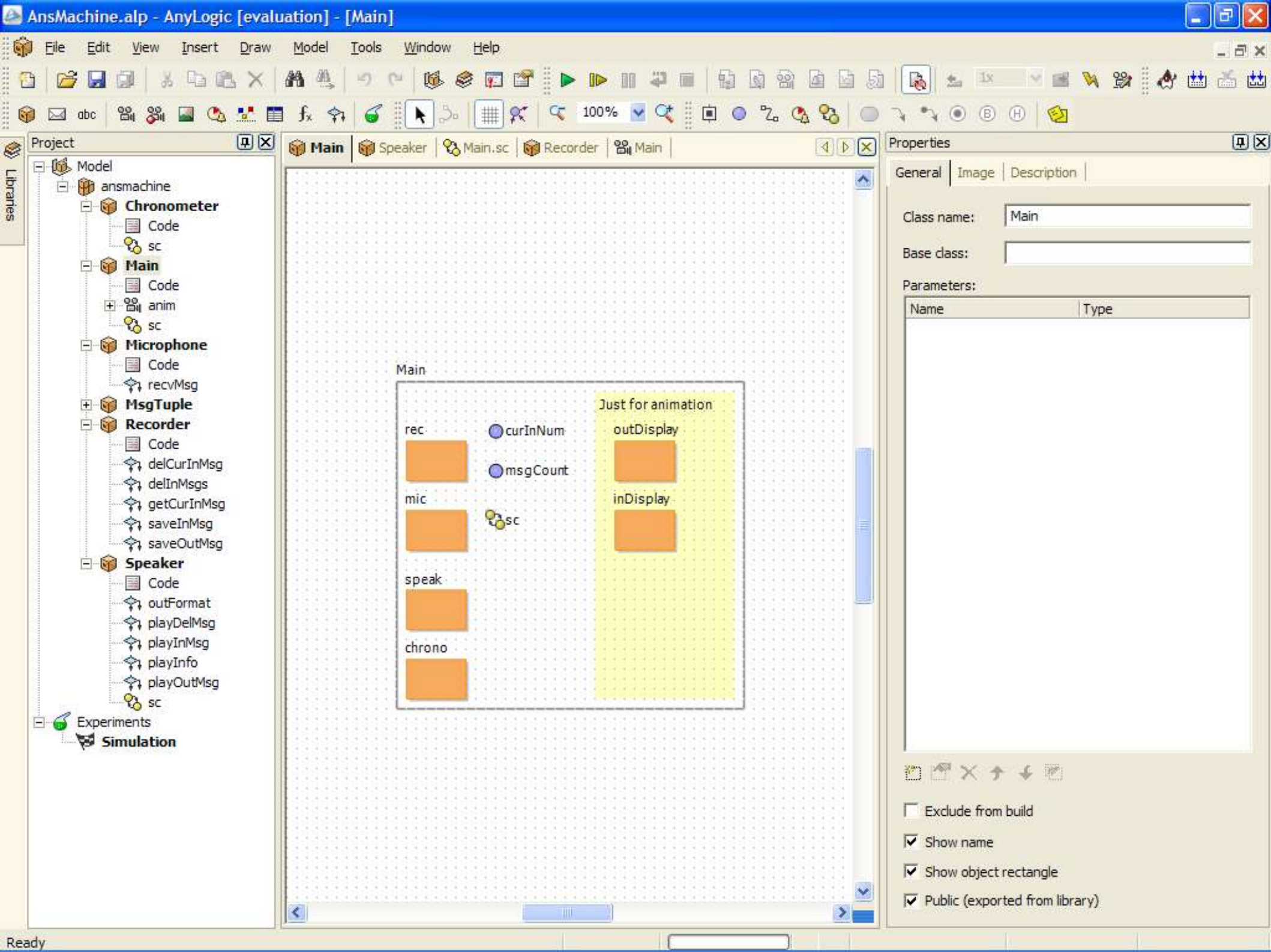
- Use-Cases
  - Record incoming message
  - Play
    - outgoing message
    - currently-selected saved message
    - all saved messages
  - Delete
    - currently-selected saved message
    - all saved messages
  - Create new outgoing message
  - Change current message selection
    - up (+1)
    - down (-1)

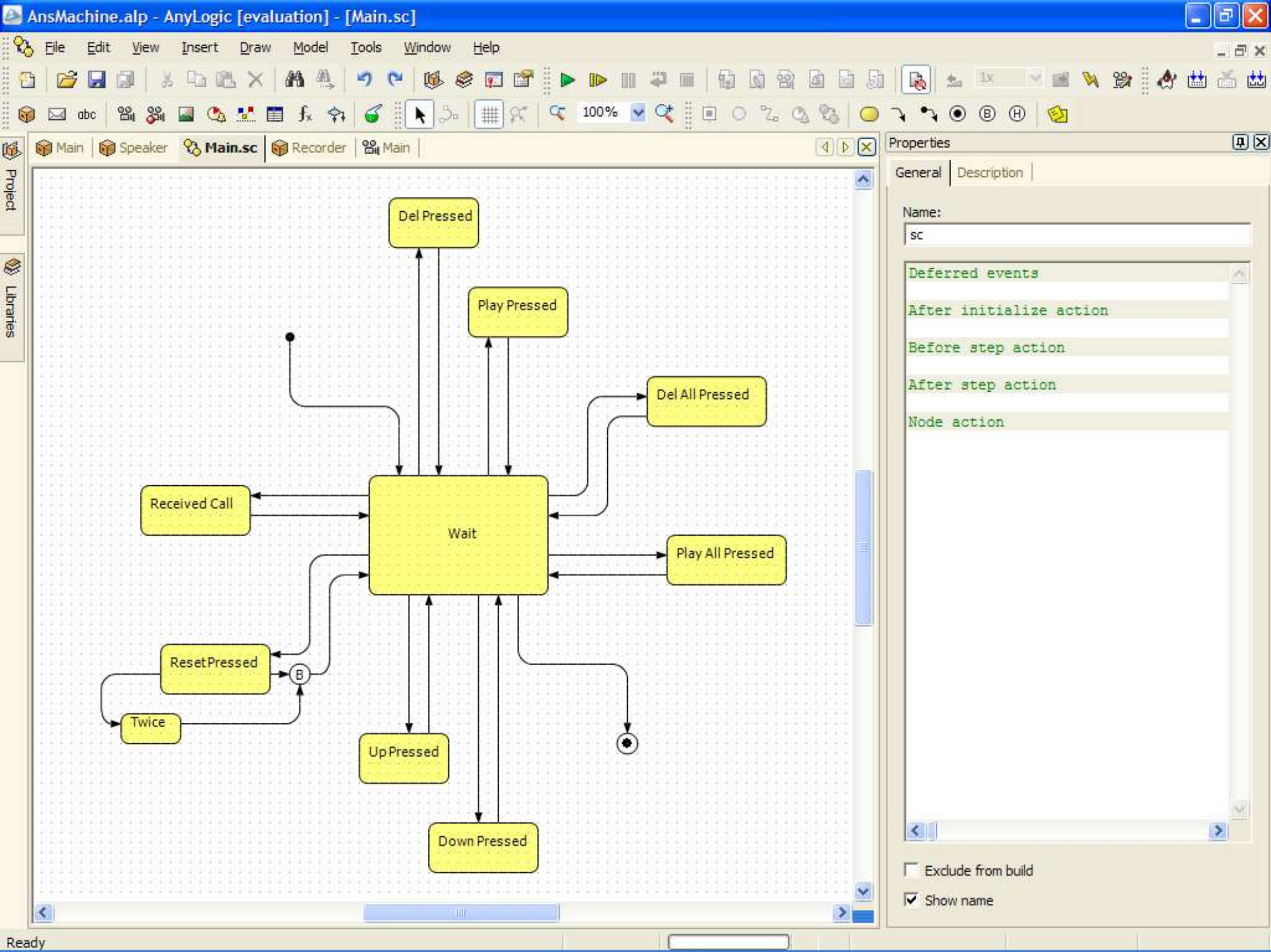
Example: Answering Machine

- Classes with statecharts
  - Chronometer
    - Tracks lengths of messages as user “speaks” into microphone
  - Main
    - Contains one instance each of Chronometer, Microphone, Recorder, Speaker
    - Contains “curInNum”, current message index
    - Contains “msgCount”, number of saved messages
    - Contains animated GUI of answering machine
  - Speaker
    - Plays outgoing message
    - Plays incoming messages
    - Plays informational messages

Example: Answering Machine

- Classes without statecharts
  - Microphone
    - Receives incoming messages
  - MsgTuple
    - Contains string data “text”
    - Contains integer data “length”
  - Recorder
    - Saves message data
    - Discards data if “blankTape” is less than message length


AnsMachine.alp - AnyLogic [evaluation] - [Main]
File Edit View Insert Draw Model Tools Window Help
Project
Model
ansmachine
Chronometer
Code
sc
Main
Code
anim
sc
Microphone
Code
recvMsg
MsgTuple
Recorder
Code
delCurInMsg
delInMsgs
getCurInMsg
saveInMsg
saveOutMsg
Speaker
Code
outFormat
playDelMsg
playInMsg
playInfo
playOutMsg
sc
Experiments
Simulation
Main
Speaker
Main.sc
Recorder
Main
Main
rec
curInNum
msgCount
mic
sc
speak
chrono
Just for animation
outDisplay
inDisplay
Properties
General
Image
Description
Class name:
Main
Base class:
Parameters:
Name
Type
Exclude from build
Show name
Show object rectangle
Public (exported from library)
Libraries
Ready


AnsMachine.alp - AnyLogic [evaluation] - [Main.sc]
File Edit View Insert Draw Model Tools Window Help
Main Speaker Main.sc Recorder Main
Project
Libraries
Del Pressed
Play Pressed
Del All Pressed
Received Call
Wait
Play All Pressed
Reset Pressed
B
Twice
Up Pressed
Down Pressed
Properties
General Description
Name:
sc
Deferred events
After initialize action
Before step action
After step action
Node action
Exclude from build
Show name
Ready

Example: Answering Machine

Demonstration

Personal Experiences using AnyLogic

- Ease of Use
  - Easy to make simple systems; fairly intuitive
  - Slight learning curve for making animations, GUI
  - Switching between Java and Math can be a little awkward
- Extent of Model-Driven Design
  - Functions in “Extra code” sections of classes don't show in Project Window
- Stability of AnyLogic
  - Some crashes during animation runs

Conclusion

- XJ Technologies AnyLogic
  - has
    - plenty of examples in many different kinds of domains
    - the ability to generate Java applets of systems
    - powerful tools for creating system animations/GUI
    - animated statecharts and structure diagrams, “animated” variables
    - 3D animation capabilities
    - the ability to interface with databases
  - but lacks
    - the ability to run active objects as different threads
    - language agnosticism
    - statechart support for orthogonal components
    - more complete implementation of statechart formalism (for analysis purposes)

References

XJ Technologies. “User's Manual.” 2005.

XJ Technologies. “Agent-Based Modeling Tutorial.” 2005.

XJ Technologies. “AnyLogic 5.3 Class Reference.”