

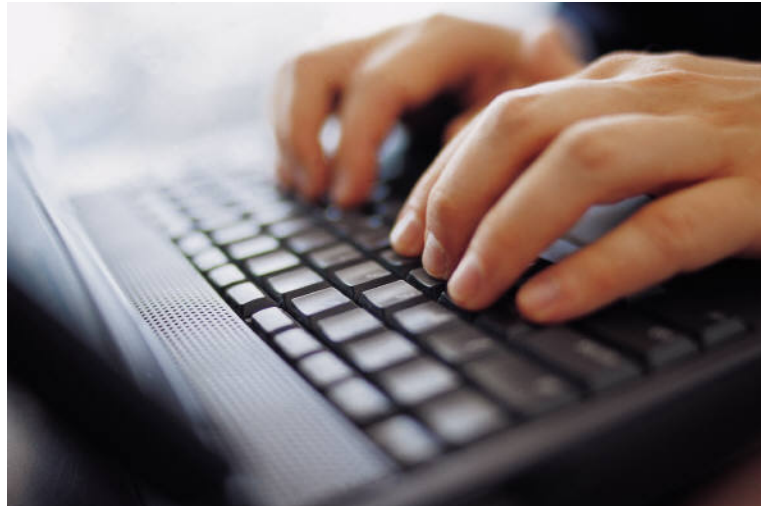
Overview of the AnyLogicInterface

Nathaniel Osgood

11-5-2009



Hands on Model Use Ahead



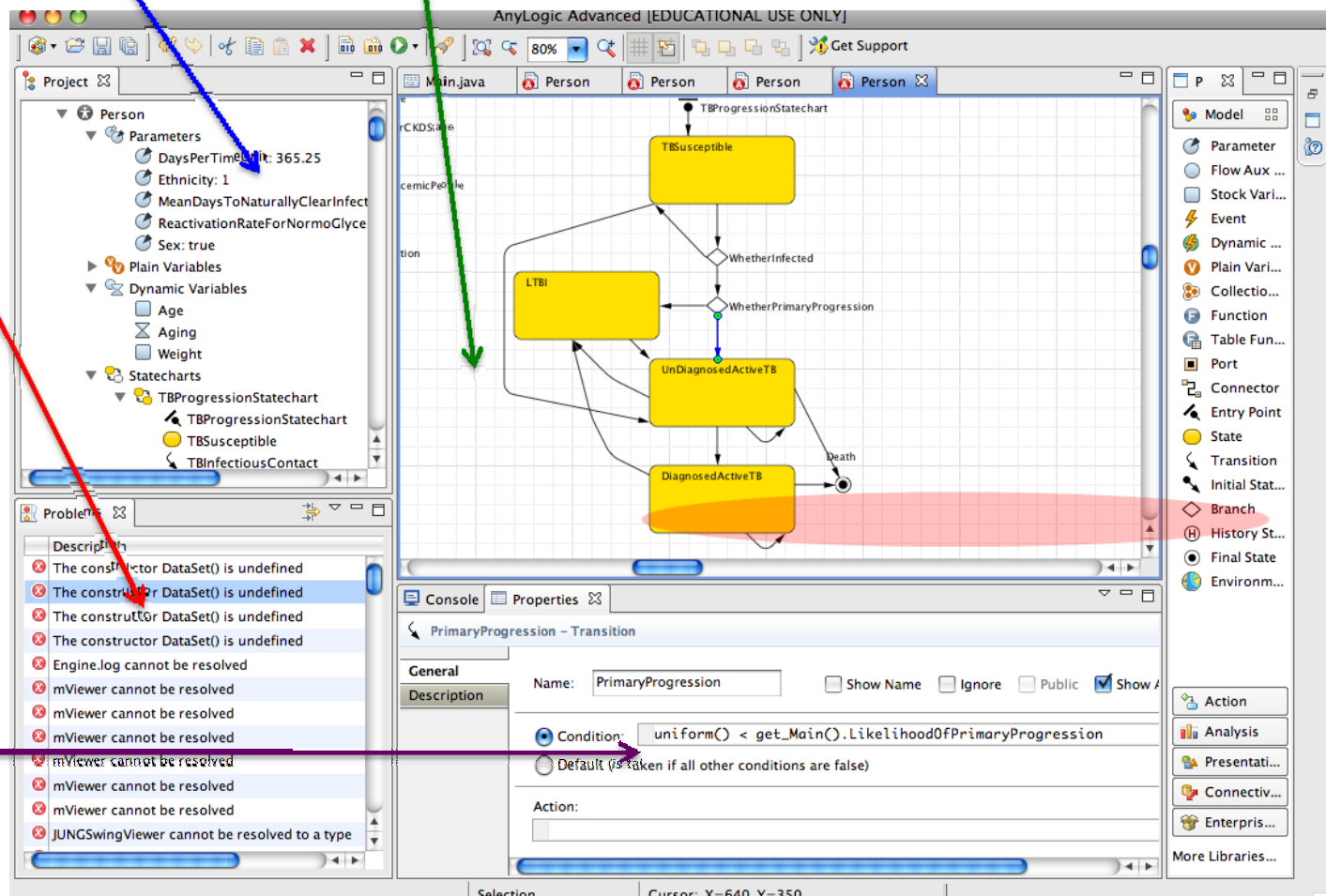
Load model: TBv1.alp

The AnyLogic User Interface

The **Project View** (provides overview of projects & components)
Common Configuration
Palette for currently selected item

Problem area
(indicates problem Building/running Model)

Properties area
(shows info on selected element in project or palette window)



The “Project View” – Hierarchically Shows the Project Components

The screenshot displays the AnyLogic Advanced [EDUCATIONAL USE ONLY] interface. On the left, the 'Project' view shows a hierarchical tree of components. The 'Main' component is highlighted with a red oval, and a red arrow points from it to the central model diagram. The 'Main' component is marked with an asterisk (*), indicating it has been modified since the last save. The central model diagram shows a statechart with states: 'TBSusceptible', 'LTBI', 'UnDiagnosedActiveTB', and 'DiagnoseActiveTB'. Transitions are labeled with conditions like 'WhetherInfected' and 'WhetherPrimaryProgression'. The bottom right panel shows the 'Properties' view for the 'Person' class, with fields for 'Name', 'Agent', and 'Startup Code'.

The “*” means that the model has changed since the last time it was saved.

You should consider saving the model when you see this!

A Critical Distinction: Design vs. Execution time

- The computational elements of Anylogic support both design & execution time presence & behaviour
 - Design time: Constructing the model, running builds
 - Execution time (“Runtime”): Simulating the model
- It is essential to be clear on what behavior & information is associated with which times
- Generally speaking, design-time elements are created to support certain runtime behaviors

The Notion of a “Build”

- A “Compiler” is a tool to convert from a program’s specification (e.g. state charts, Action diagrams, etc.) to a representation that can be executed
- Normally a compiler is applied to each of several components of a program (e.g. classes)
- AnyLogic’s “build” process applies a compiler to the components of the AnyLogic model

The “Problems View”

The screenshot displays the AnyLogic Advanced [EDUCATIONAL USE ONLY] interface. The main workspace shows a statechart model for a person's TB progression. The model includes states: **TBSusceptible**, **LTBI**, **UnDiagnosedActiveTB**, and **DiagnosedActiveTB**. Transitions are labeled with events like **WhetherInfected** and **WhetherPrimaryProgression**. A final state is labeled **Death**.

The left sidebar shows the **Project** tree with the **Person** model selected. The **Statecharts** section lists the **TBProgressionStatechart** and its states.

The bottom-left **Problems** view is highlighted with a red circle and a red arrow pointing to it from the title. It lists several errors:

- The constructor `DataSet()` is undefined
- The constructor `DataSet()` is undefined
- The constructor `DataSet()` is undefined
- The constructor `DataSet()` is undefined
- `Engine.log` cannot be resolved
- `mViewer` cannot be resolved
- `mViewer` cannot be resolved

The bottom-right **Person - Active Object Class** view shows the class configuration, including the **Imports section**, **Extends** field, **Implements** field, and **Additional class code**:

```
public static final int MsgInfectiousTBContact=1;
public static final int MsgForceInitialTBInfection=2;
```

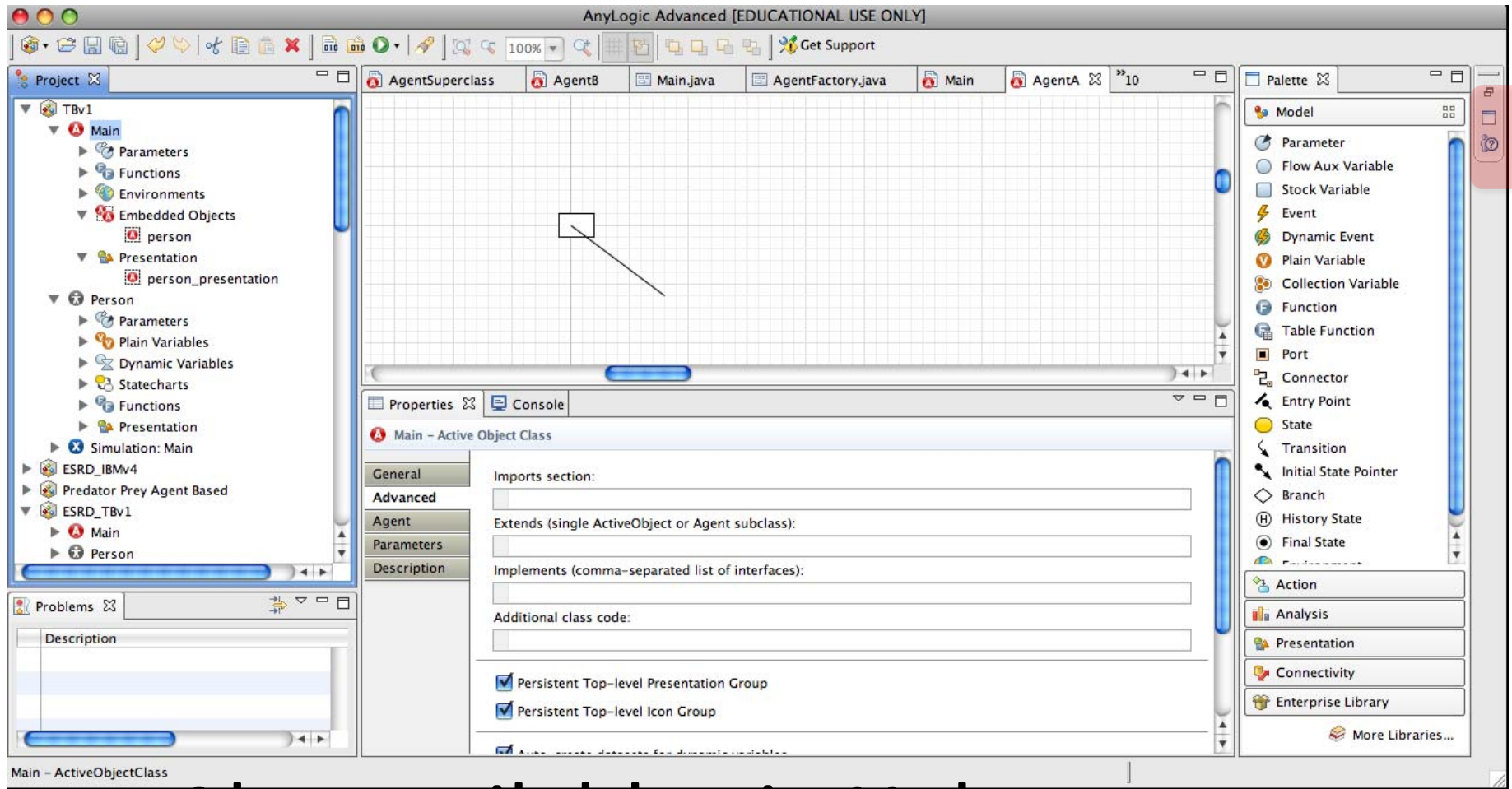
Multiple Tabs (switch among tasks)

The screenshot displays a software development environment with two main components:

- Diagram View:** A state machine diagram on a grid background. It features two yellow rectangular states: `UnDiagnosedActiveTB` (top) and `DiagnosedActiveTB` (bottom). Transitions are shown as arrows: a curved arrow from `UnDiagnosedActiveTB` to `DiagnosedActiveTB`, a curved arrow from `DiagnosedActiveTB` back to `UnDiagnosedActiveTB`, and a straight arrow from `DiagnosedActiveTB` to a final state (a circle with a dot). A label "Death" is placed near the final state. A red arrow points from the top-left towards the diagram.
- Properties Window:** A panel at the bottom with tabs for "Console", "Properties", and "Person - Active Object Class". The "Person - Active Object Class" tab is selected and highlighted in red. It contains a sidebar with tabs: "General", "Advanced", "Agent", "Parameters", and "Description". The "Advanced" tab is selected. The main area of this tab contains the following sections:
 - Imports section:** An empty text field.
 - Extends (single ActiveObject or Agent subclass):** An empty text field.
 - Implements (comma-separated list of interfaces):** An empty text field.
 - Additional class code:** A text area containing the following code:

```
public static final int MsgInfectiousTBContact=1;  
public static final int MsgForceInitialTBInfection=2;
```

Displaying the Splash Screen (to Access Samples)



Also available via Help menu

Displaying the Splash Screen (to Access Samples)



Running a Model

The screenshot displays the AnyLogic Advanced [EDUCATIONAL USE ONLY] interface. The main workspace shows a statechart diagram for a tuberculosis model. The diagram includes states: **TBSusceptible**, **LTBI**, **UnDiagnosedActiveTB**, and **DiagnosedActiveTB**. Transitions are labeled with conditions like **WhetherInfected** and **WhetherPrimaryProgression**. A final state is labeled **Death**.

On the left, the **Project** tree shows the hierarchy: **TBv1*** > **Main** > **Parameters**. The **Parameters** list includes: **DaysFromDiagnosisUntilRecovery**, **DaysUntilDiagnosis** (60), **DiagnosedPerDayTBContactRatePer**, **LikelihoodOfPrimaryProgression** (1), **PerContactTBInfectionProbability** (1), and **UndiagnosedPerDayTBContactRate**.

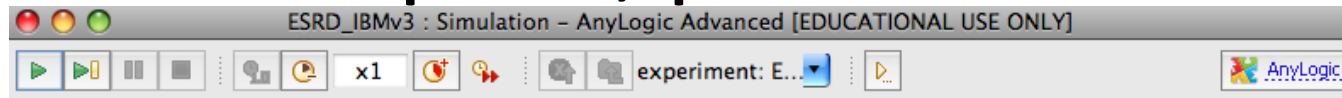
The **Recent Experiment** window lists several simulation models, including **TBv1 / Simulation**, **ESRD_IBMv4 / Simulation**, **CTL State Variable V3 / Simulation**, **Predator Prey Agent Based / Simulation**, **ESRD_TBv1 / Simulation**, **Emergency Department Tulsa / Simulation**, **AnqiModelV1 / Simulation**, **Network Modification of SIR AB / Simulation**, **MalariaV2 / Simulation**, **SIR / Simulation**, and **SIR Agent Based / Simulation**.

The **Properties** window at the bottom right shows the **Person - Active Object Class** settings. The **General** tab is active, showing the **Name** as **Person** and the **Agent** checkbox checked. The **Startup Code** field is empty.

The **Model** toolbar on the right contains icons for **Parameter**, **Flow Aux ...**, **Stock Vari...**, **Event**, **Dynamic ...**, **Plain Vari...**, **Collectio...**, **Function**, **Table Fun...**, **Port**, **Connector**, **Entry Point**, **State**, **Transition**, **Initial Stat...**, **Branch**, **History St...**, **Final State**, and **Environm...**.

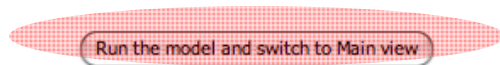
Experiment Set up

(Use to set speed, parameters via UI)

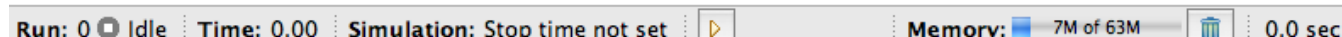


ESRD_IBMv3

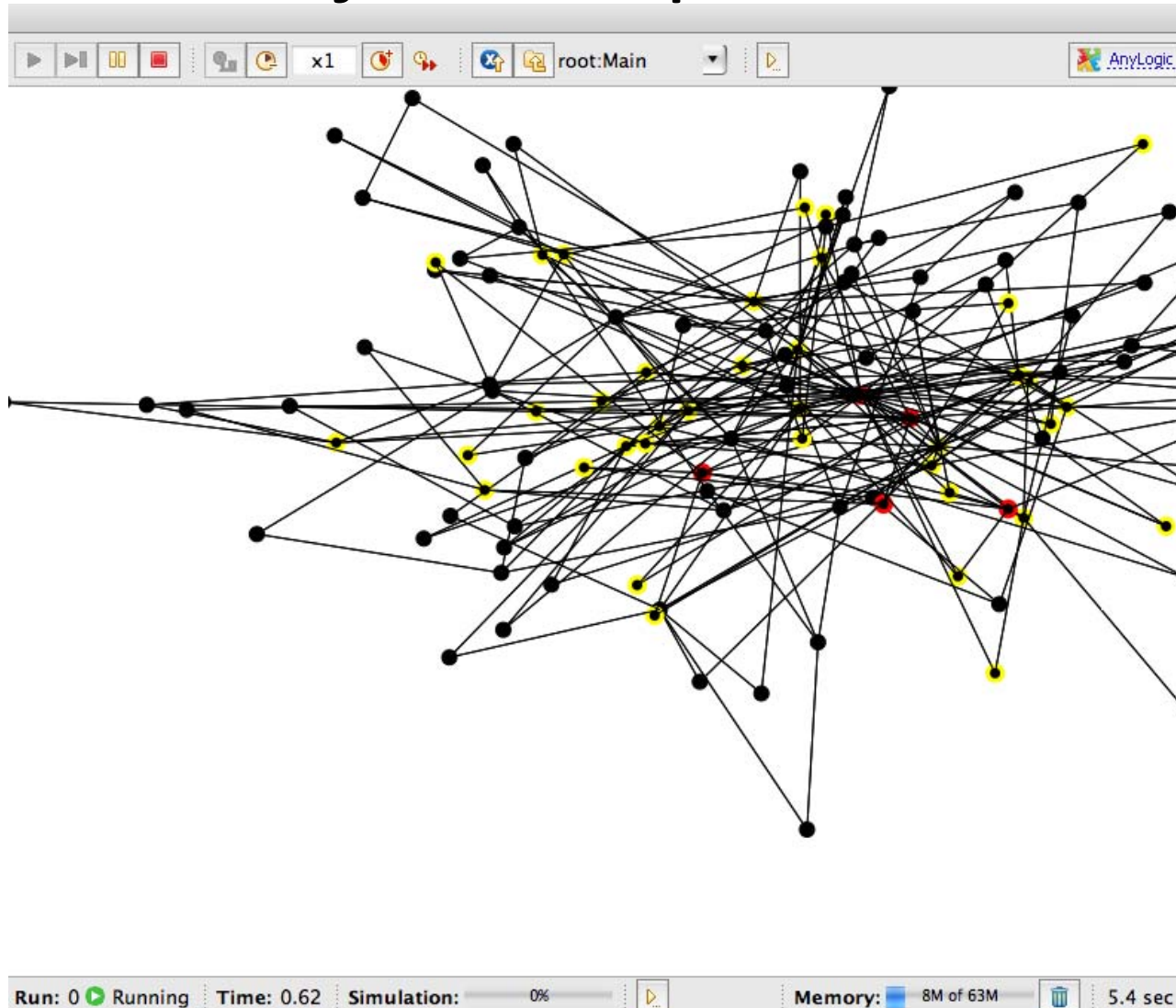
Experiment setup page



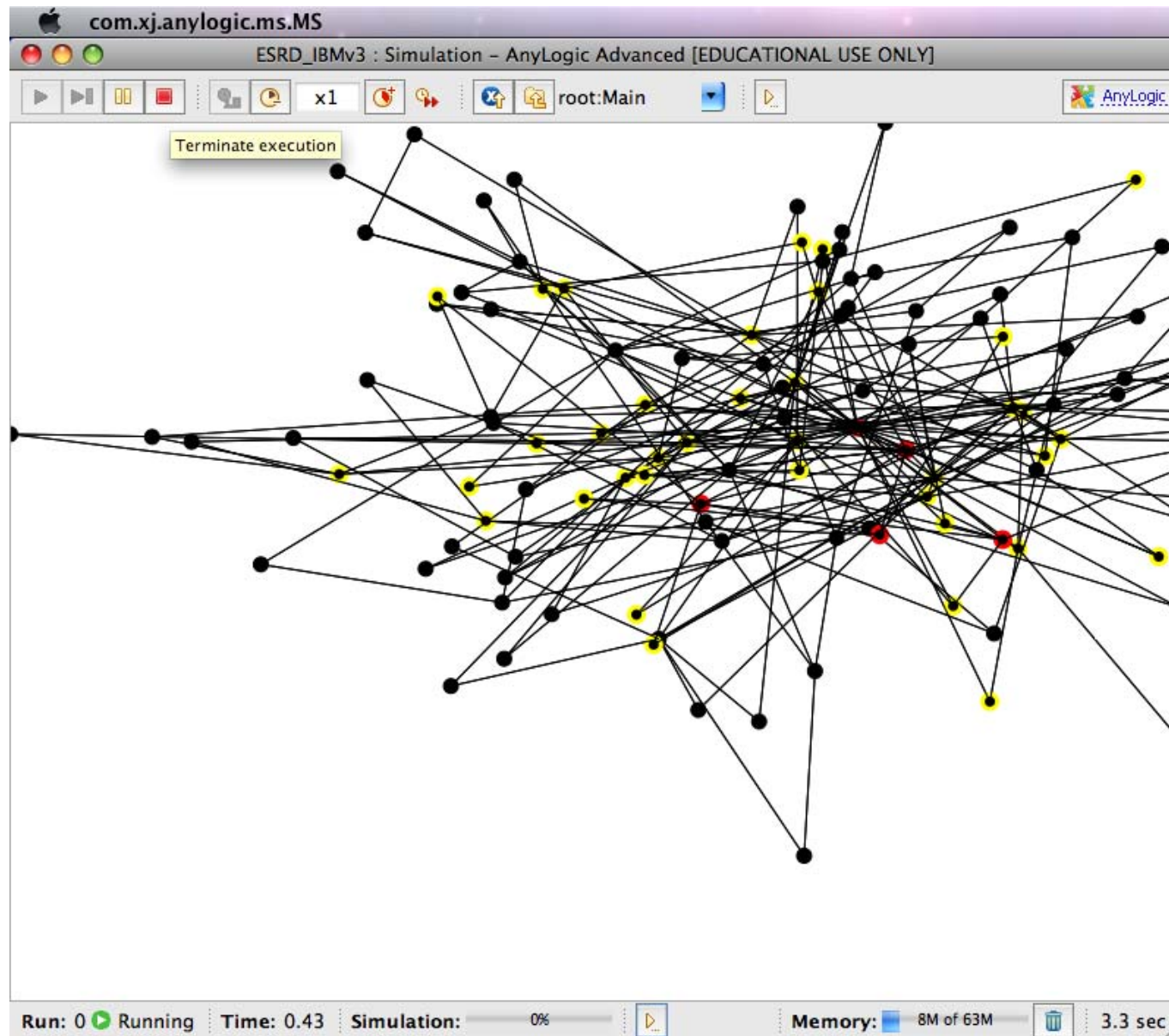
Press this button to switch to the model presentation display



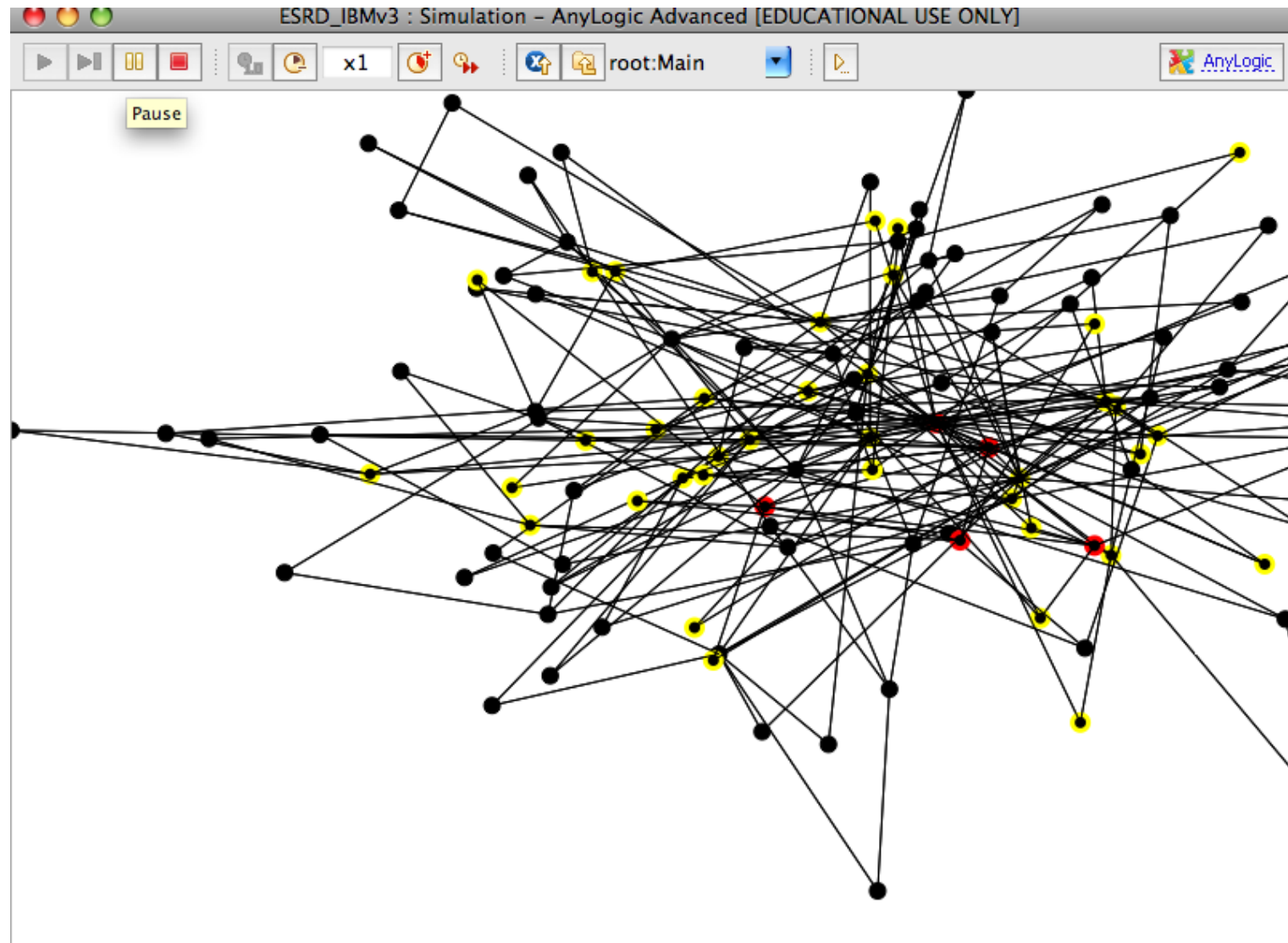
Presentation of the Model Main Object in Operation



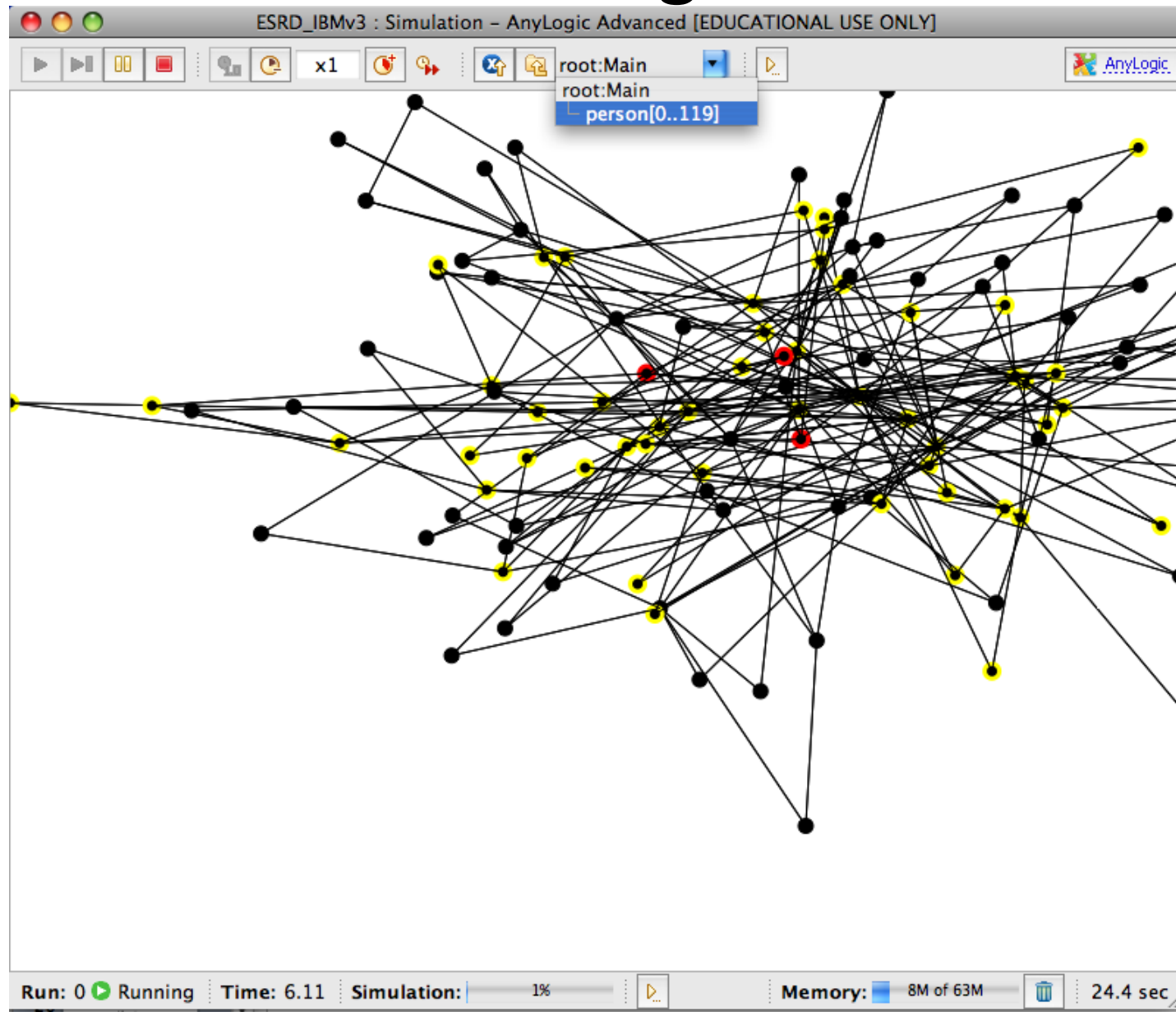
Terminating Model Execution



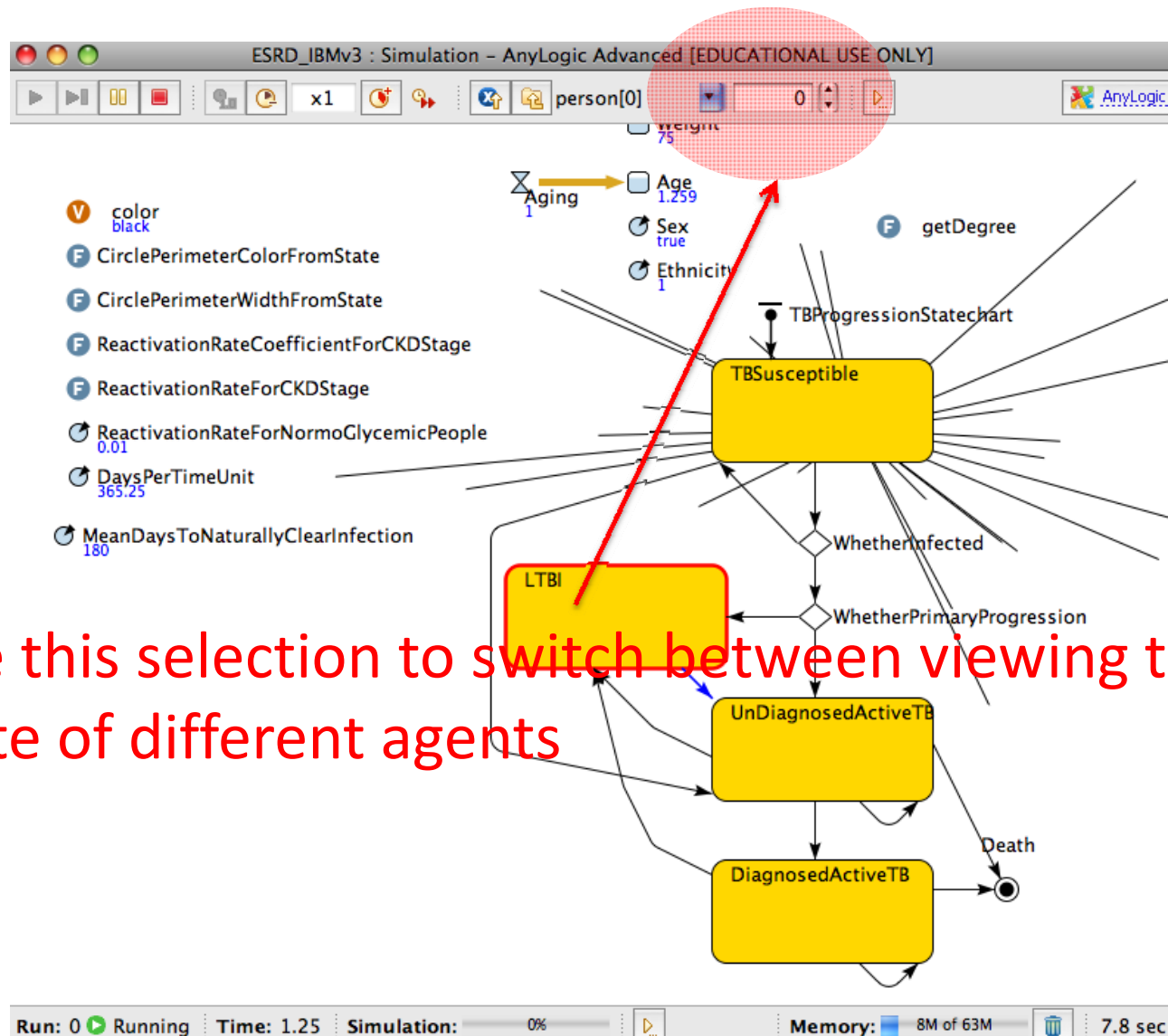
Pausing the Model



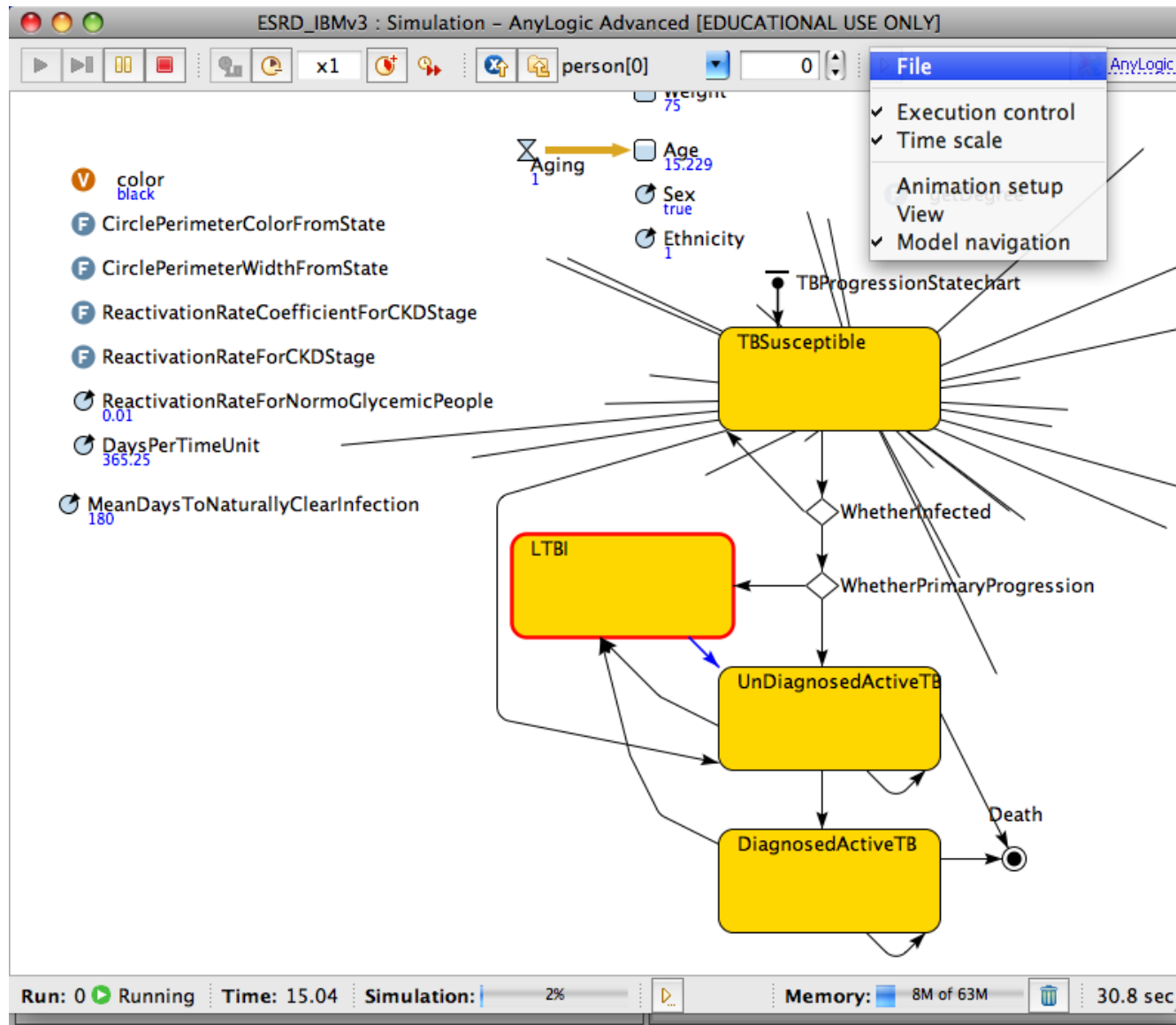
Drill Down from the Model to Particular Agents



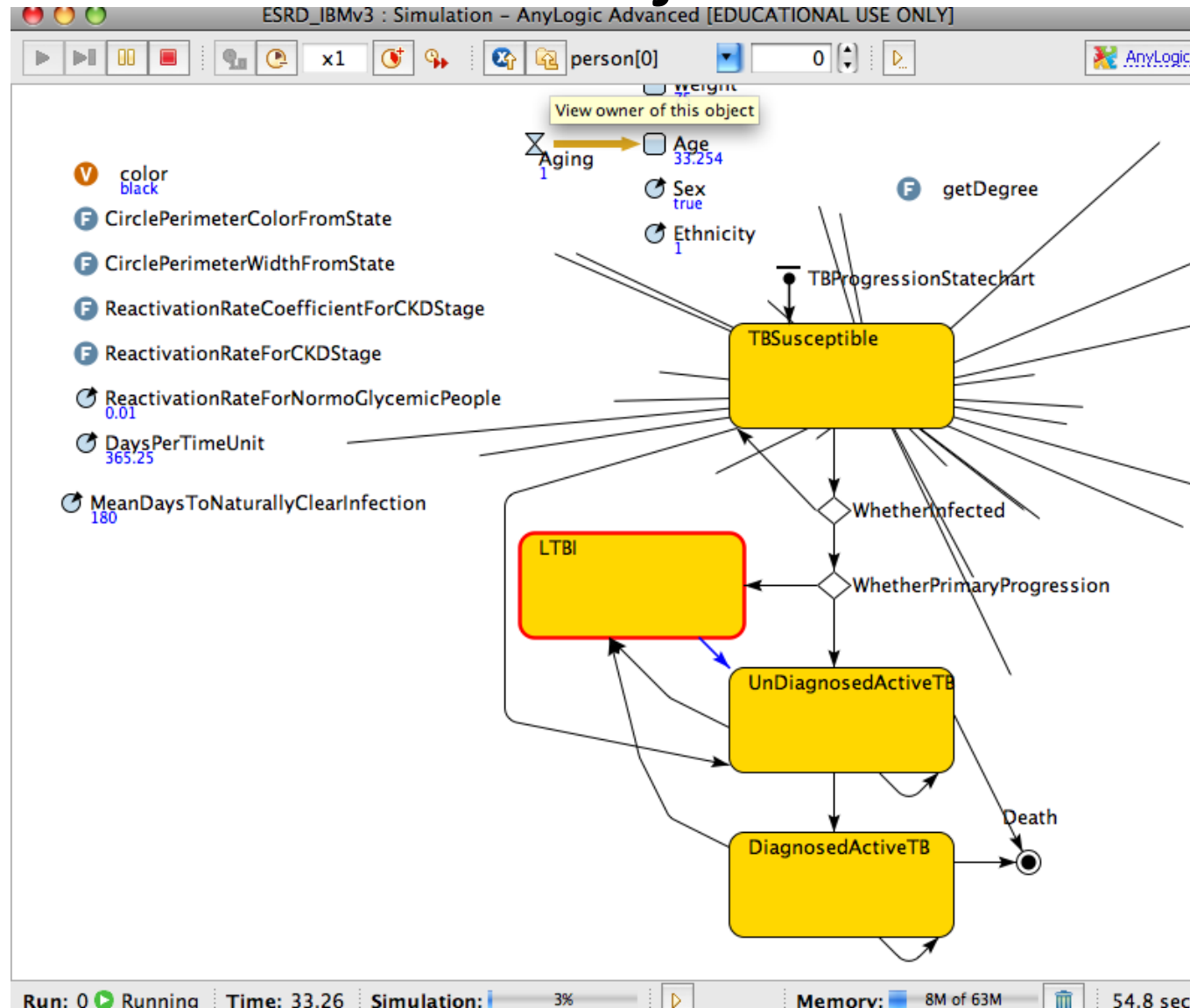
View of Agent State



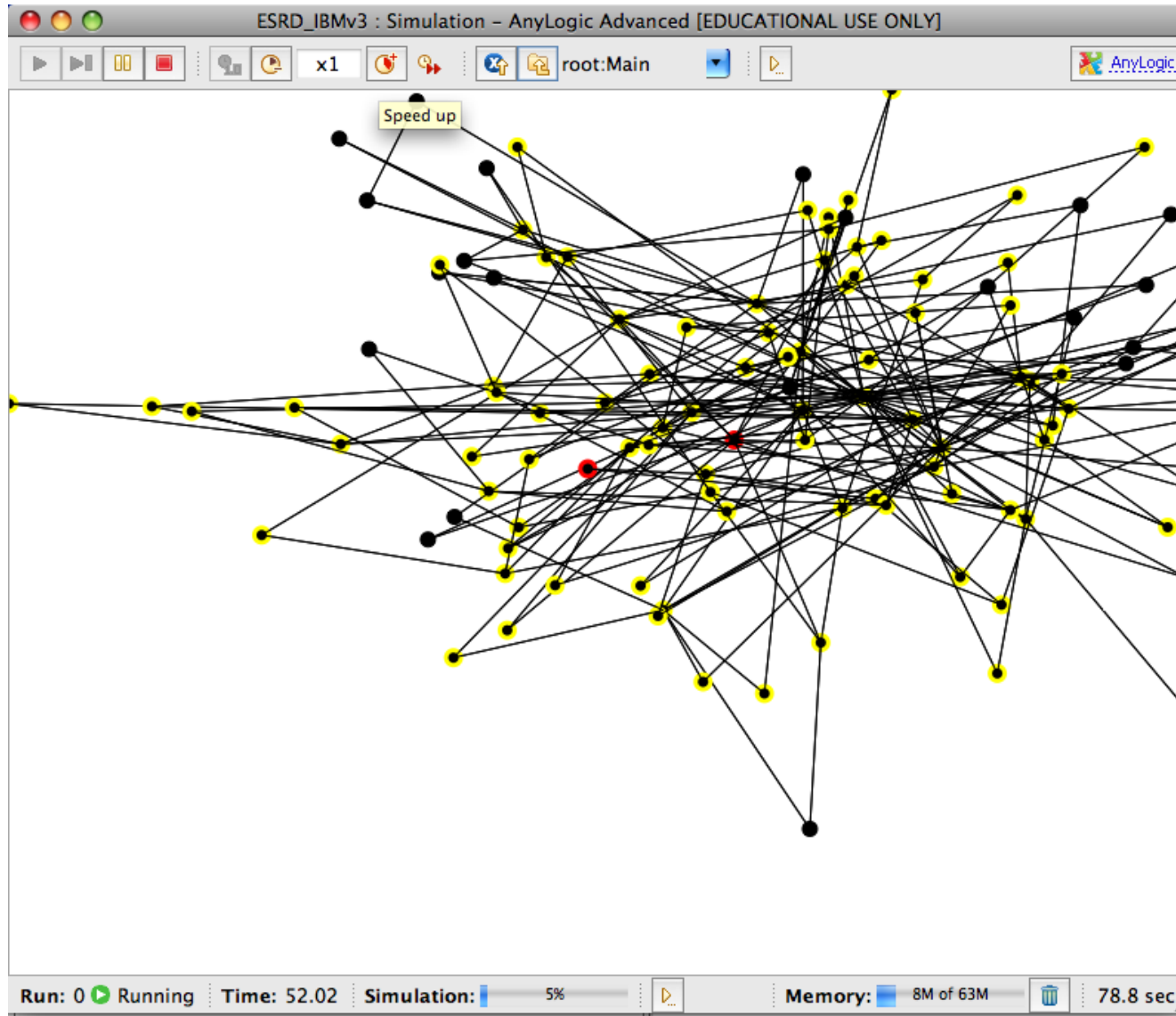
Customizing the Model Running User Interface



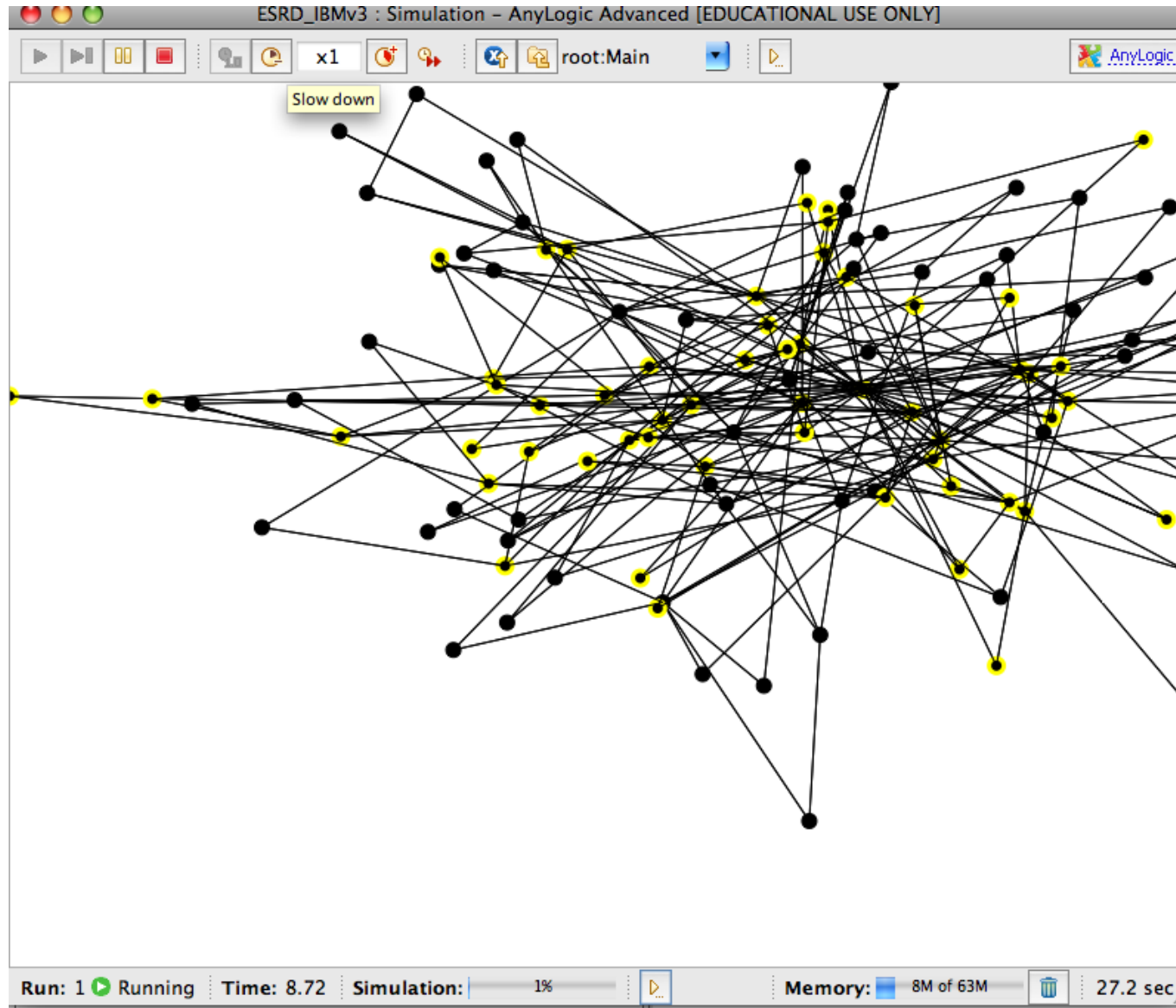
Switching Back to View the Main Object



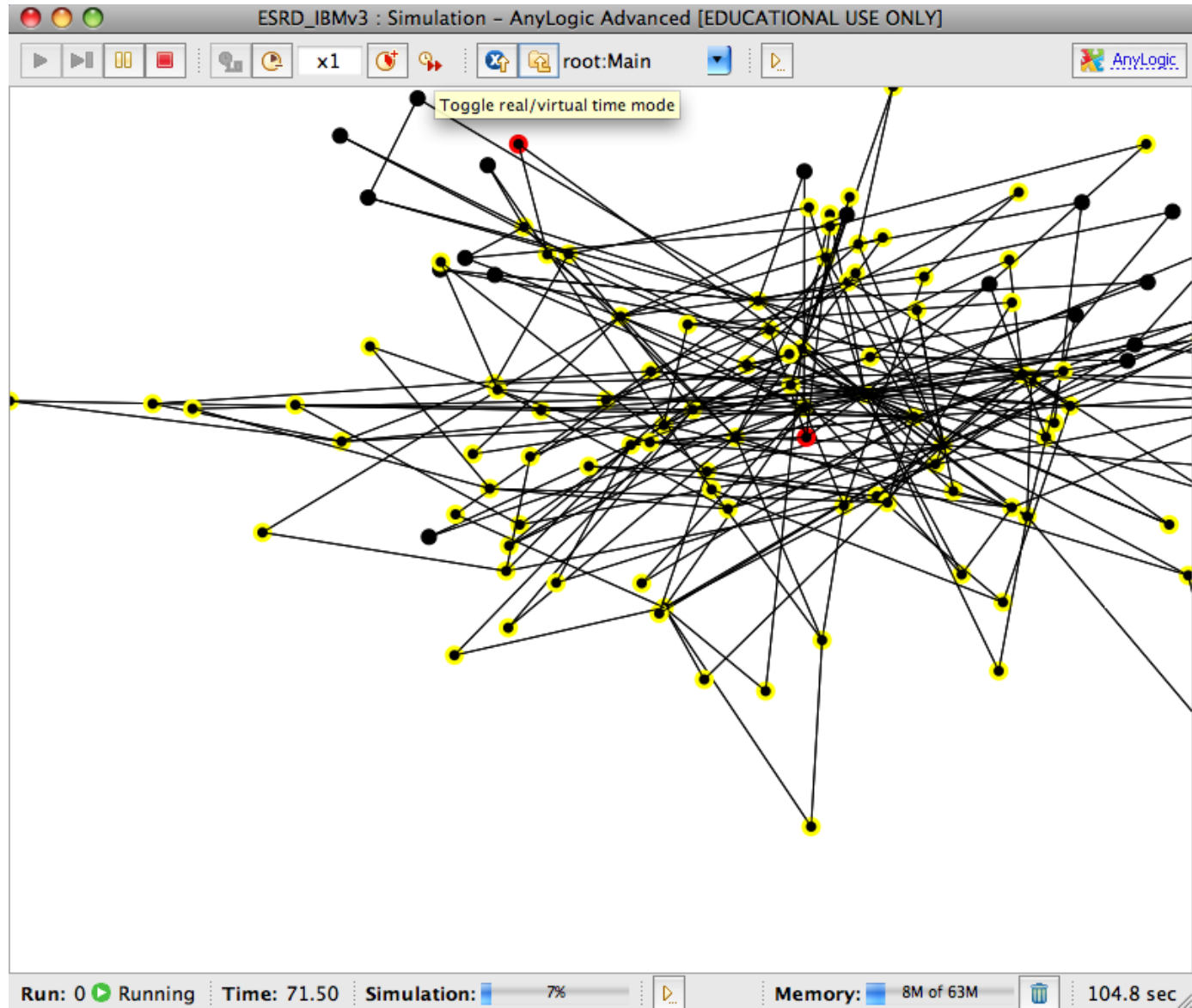
Controlling Simulation Speed (Speeding up)



Controlling Simulation Speed (Slowing Down)

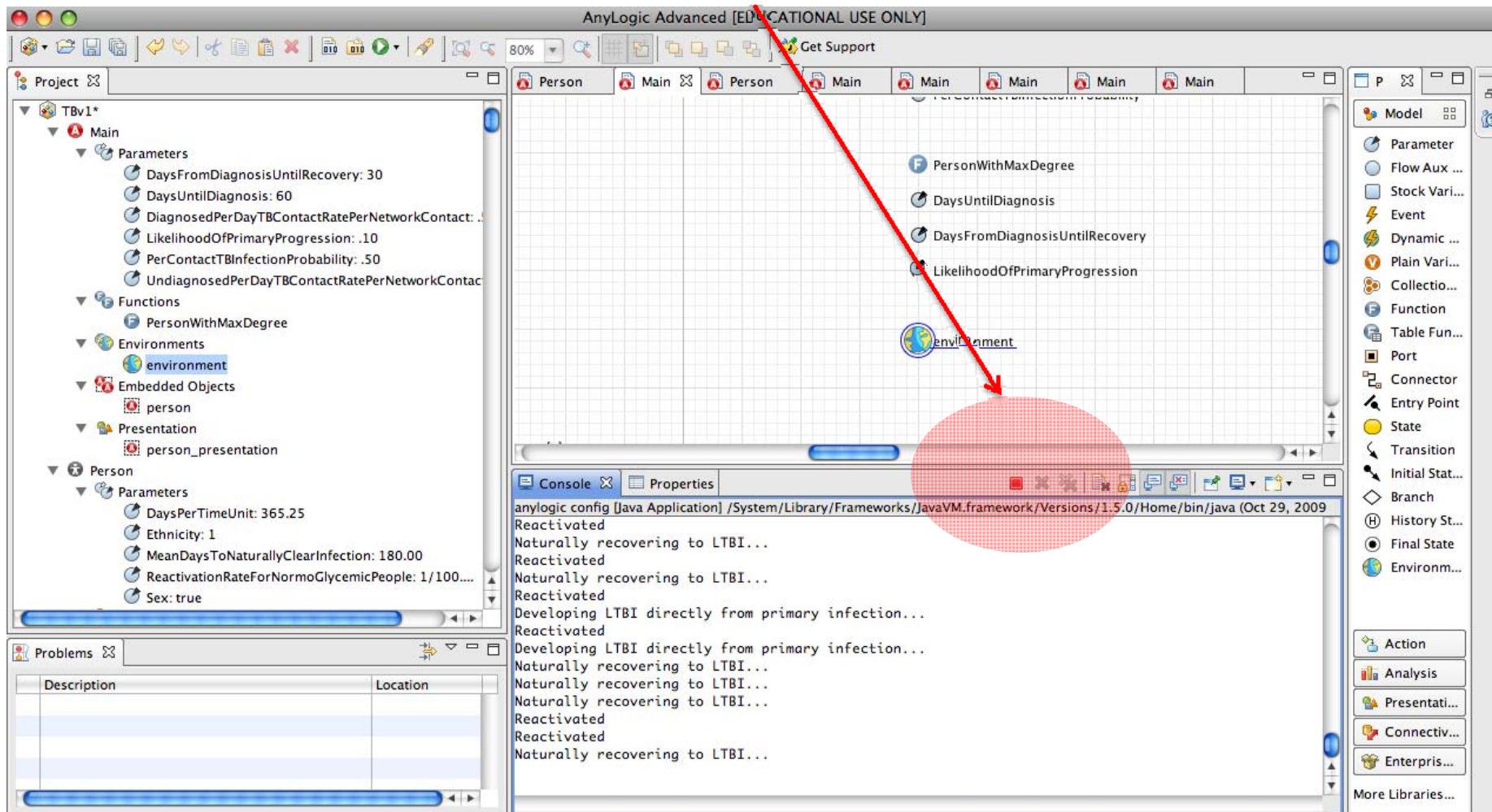


Toggling between Maximum and a Throttled Speed



Another Way to Terminate a Simulation

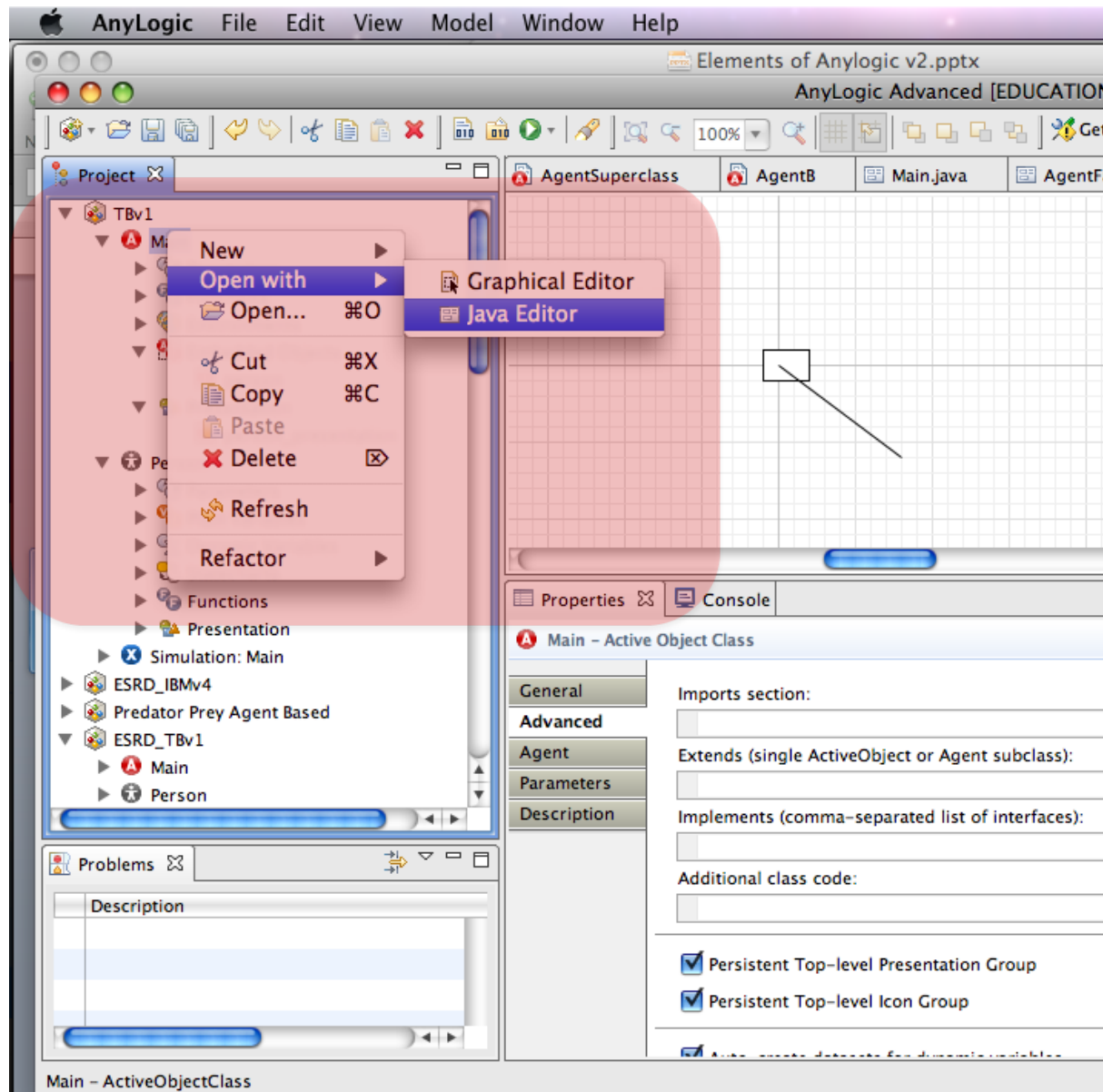
Use this Console “stop” button to terminate the simulation



Inspecting the Java code

- As a step towards creating an executable representation of the code, AnyLogic creates a Java representation
 - If you want to see the Java code for a model, you will need to do a “build”
- Sometimes it can be helpful to look at this Java code
 - To find errors about which AnyLogic may be complaining
 - Advanced: To see how things are being accomplished or “work”

Requesting Viewing of Java Code



AnyLogic: Above & Below the “Hood”

- One of AnyLogic’s greatest strengths is the presence of diverse & powerful *declarative* mechanisms for building models
 - These let you focus on the “what” you are modeling, rather than “how” it will be implemented
 - AnyLogic will take care of figuring out the “*how*”
 - This is in contrast to writing code in a general purpose computer language, which requires specifying more of the *how*
- For Anylogic, declarative mechanisms include statecharts, stock & flow diagrams, “action” flow charts & process maps
- Other familiar declarative mechanisms include spreadsheet formulas &vensim stock & flow diagrams.
- For most interactions with AnyLogic, you will be able to specify your intentions using these declarative mechanisms
- On occasion, you will need to write & look at Java code

The Notion of a Code “Library”

- A “library” lets third parties (e.g. xjtek) share compiled code they have developed with others
- The classes built into our AnyLogic projects (e.g. Agent, ActiveObject, NetworkResourcePool, etc.) are contained in the library
- The available libraries that come with AnyLogic & Java have many additional components that can offer tremendous additional functionality
 - By tapping into this functionality, we can avoid having to write code ourselves
- To use a library, you need to know what is in it!

Getting to the AnyLogic Help

- Choose “Help”/”Help Contents”

Getting Information on the Anylogic (Java) Libraries

The screenshot shows a web browser window titled "Help - AnyLogic Advanced". The address bar displays "http://127.0.0.1:63191/help/index.jsp". The browser's search bar contains "Google". Below the browser window, there is a search bar with the text "Search:" and a "GO" button. To the right of the search bar, it says "Search scope: All topics".

On the left side of the page, there is a "Contents" panel with a list of topics:

- Enterprise Library Tutorial
- Enterprise Library Reference Guide
- AnyLogic Help
- System Dynamics Tutorial
- Agent-Based Modeling Tutorial
- API Reference

The "API Reference" item is highlighted with a red background.

The main content area is titled "Using AnyLogic Help System". It contains the following text:

Browse topics in the **Contents** frame on the left. Click on a topic to have it displayed. Use the **Back** and **Forward** buttons to navigate within the history of viewed topics.

Style conventions

To make things easy to follow, there are a number of formatting conventions and images used throughout the book:

- Bold** – Used for the names of UI elements such as menus, buttons, field labels, palettes, and view titles.
- Italic* – Used for emphasizing new terms.
- `Courier` – Used for code examples, references to class and function names.
-  – "How to" scenario.
-  – Reference to another help topic.
-  – The feature is available in **AnyLogic Professional edition** only.

Printing multiple help topics

You can now [print multiple topics](#) in the help window with a single action. The new print drop-down button above the table of contents allows you to print a complete topic sub-tree at any level.

Searching

To quickly locate topics on a particular subject in the documentation, enter a query in the **Search** field. Use the **Search** frame to display the **Search** view. After you run a search and find a topic you were looking for, click **Show in Table of Contents** button to match the navigation tree with the current topic.

Finding out Information Interfaces for Library Elements 1

The screenshot shows a web browser window titled "Help - AnyLogic Advanced". The address bar displays "http://127.0.0.1:63191/help/index.jsp". The browser has several tabs open, including "Wikipedia", "Save to Delicious", "My Delicious", "CMPT 858", "CMPT 371", "Env Canada Sask Weather", "Weather: Saskatoon", "Env Canada PA Weather", and "The Pali Tex...h dictionary". The search bar contains "GO" and "Search scope: All topics".

The left sidebar shows the "Contents" menu with the following items:

- AnyLogic Help
- System Dynamics Tutorial
- Agent-Based Modeling Tutorial
- API Reference
 - com.xj.anylogic.engine
 - AbstractShapeGISMap
 - ActiveObject
 - ActiveObjectArrayList
 - ActiveObjectCollection
 - ActiveObjectIntegrationManager
 - ActiveObjectList
 - Agent**
 - CustomDistribution
 - Dimension
 - DynamicEvent
 - Engine
 - Environment
 - Environment.AgentCollection
 - Event
 - EventCondition
 - EventOriginator
 - EventRate
 - EventTimeout
 - Experiment
 - ExperimentCompareRuns
 - ExperimentOptimization
 - ExperimentParamVariation
 - ExperimentSimulation

The main content area displays the API Reference for the `com.xj.anylogic.engine` package. The navigation tabs are: Overview, Package, **Class**, Use Tree, Deprecated, Index, Help. The selected tab is "Class". The page shows the class `com.xj.anylogic.engine.Agent` which extends `ActiveObject`. The class hierarchy is shown as:

```
java.lang.Object
├── com.xj.anylogic.engine.Presentable
│   └── com.xj.anylogic.engine.Utilities
│       └── com.xj.anylogic.engine.ActiveObject
│           └── com.xj.anylogic.engine.Agent
```

All Implemented Interfaces:
`com.xj.anylogic.engine.internal.Child`, `java.io.Serializable`

public class Agent
extends `ActiveObject`

A subclass of `ActiveObject` designed to support agent based modeling, in particular:

- time (continuous or discrete)
- space (continuous or discrete) and spacial animation
- connections between agents, networks (e.g. social) and their visualization
- communication - message passing and broadcasting

A user-defined agent class should be a subclass of `Agent` in order to use those features.

If your model is agent based, but none of the above features are required, it is recommended to use regular `ActiveObject` as a base class for your agents, and not this class: `Agent` requires 36+ bytes more memory than `ActiveObject`.

Finding out Information Interfaces for Library Elements 2

Help - AnyLogic Advanced

http://127.0.0.1:63191/help/index.jsp

Wikipedia Save to Delicious My Delicious CMPT 858 CMPT 371 Env Canada Sask Weather Weather: Saskatoon Env Canada PA Weather The Pali Tex...h dictionary

Help - AnyLogic Advanced

Search: **GO** Search scope: All topics

Contents

- AnyLogic Help
- System Dynamics Tutorial
- Agent-Based Modeling Tutorial
- API Reference
 - com.xj.anylogic.engine
 - AbstractShapeGISMap
 - ActiveObject
 - ActiveObjectArrayList
 - ActiveObjectCollection
 - ActiveObjectIntegrationManager
 - ActiveObjectList
 - Agent**
 - CustomDistribution
 - Dimension
 - DynamicEvent
 - Engine
 - Environment
 - Environment.AgentCollection
 - Event
 - EventCondition
 - EventOriginator
 - EventRate
 - EventTimeout
 - Experiment
 - ExperimentCompareRuns
 - ExperimentOptimization
 - ExperimentParamVariation
 - ExperimentSimulation

Fields inherited from class com.xj.anylogic.engine.Presentable

[ALIGNMENT_CENTER](#), [ALIGNMENT_LEFT](#), [ALIGNMENT_RIGHT](#), [LINE_STYLE_DASHED](#), [LINE_STYLE_DOTTED](#), [LINE_STYLE_SOLID](#), [SHAPE_ARC](#), [SHAPE_BUTTON](#), [SHAPE_CAD](#), [SHAPE_CHART_BAR](#), [SHAPE_CHART_HISTOGRAM](#), [SHAPE_CHART_HISTOGRAM2D](#), [SHAPE_CHART_PIE](#), [SHAPE_CHART_PLOT](#), [SHAPE_CHART_STACK](#), [SHAPE_CHART_TIME_COLOR](#), [SHAPE_CHART_TIME_PLOT](#), [SHAPE_CHART_TIME_STACK](#), [SHAPE_CHECKBOX](#), [SHAPE_COMBOBOX](#), [SHAPE_CURVE](#), [SHAPE_EMBEDDED_OBJECT](#), [SHAPE_FILECHOOSER](#), [SHAPE_GROUP](#), [SHAPE_IMAGE](#), [SHAPE_LINE](#), [SHAPE_LISTBOX](#), [SHAPE_OVAL](#), [SHAPE_PIXEL](#), [SHAPE_POLYLINE](#), [SHAPE_PROGRESSBAR](#), [SHAPE_RADIOBUTTONS](#), [SHAPE_RECTANGLE](#), [SHAPE_ROUNDED_RECTANGLE](#), [SHAPE_SLIDER](#), [SHAPE_TEXT](#), [SHAPE_TEXTFIELD](#)

Constructor Summary

[Agent](#)([Engine](#) engine, [ActiveObject](#) owner, [ActiveObjectCollection](#)<?> collection)

Method Summary

java.lang.String	agentInfo ()
void	connectTo (Agent a) Creates a bi-directional connection between this agent and a given other agent.
void	deliver (java.lang.Object msg, Agent dest) Delivers a message to a given agent immediately during this method call.
void	deliver (java.lang.Object msg, int mode) Delivers a message to an agent or a group of agents, as specified by the mode parameter immediately during this method call.
boolean	disconnectFrom (Agent a)

Using Libraries

- There are two major libraries that can be used “automatically”: Java libraries & AnyLogic libraries
- To use an object in the Java libraries, you will use an “import” statement

Using External Libraries

- There are tremendous numbers of 3rd party libraries available for Java
- The functionality associated with these libraries is incredibly diverse
- Many of these libraries are available for free; others are sold
- It is very easy to make use of the functionality of 3rd party libraries from AnyLogic
 - In order to do this, AnyLogic needs to “know about” the external library.

Adding External Libraries 1

The screenshot displays the AnyLogic Advanced [EDUCATIONAL USE ONLY] interface. The main workspace shows a diagram of a 'Person' entity with attributes like 'color', 'Weight', 'Age', 'Sex', and 'Ethnicity'. The 'Aging' process is highlighted with a yellow arrow. The 'Project' pane on the left shows a hierarchical view of the model structure, including 'Recovered', 'ImmunityWaning', 'InfectionTransmission', 'Functions', 'Presentation', 'Simulation: Main', and 'TBRiskFactors'. The 'Properties' pane at the bottom shows the 'HIV_v3_8Anylogic622 - Model' with a table of dependencies. The 'Console' pane shows error messages: 'The import edu cannot be resolved'.

AnyLogic Advanced [EDUCATIONAL USE ONLY]

Project

- Recovered
- ImmunityWaning
- InfectionTransmission
- Functions
 - getDegree
- Presentation
 - oval
 - line
- Simulation: Main
 - Presentation
- TBRiskFactors
 - Main
 - Person
 - Parameters
 - Plain Variables
 - Dynamic Variables
 - Statecharts
 - Functions
 - Presentation
 - Simulation: Main
- CTL State Variable V4
- HIV_v3_8Anylogic622*
 - Main
 - Person

Person

color

CirclePerimeterColorFromState

CirclePerimeterWidthFromState

Weight

Age

Sex

Ethnicity

Aging

Console

Properties

HIV_v3_8Anylogic622 - Model

General

Dependencies

Description

AnyLogic libraries required to build the model:

Name	Version	Location

Add...

Remove

Jar files and class folders required to build the model:

Location
jung-1.7.4.jar

Add...

Remove

The import edu cannot be resolved

The import edu cannot be resolved

The import edu cannot be resolved

The import edu cannot be resolved

Action

Analysis

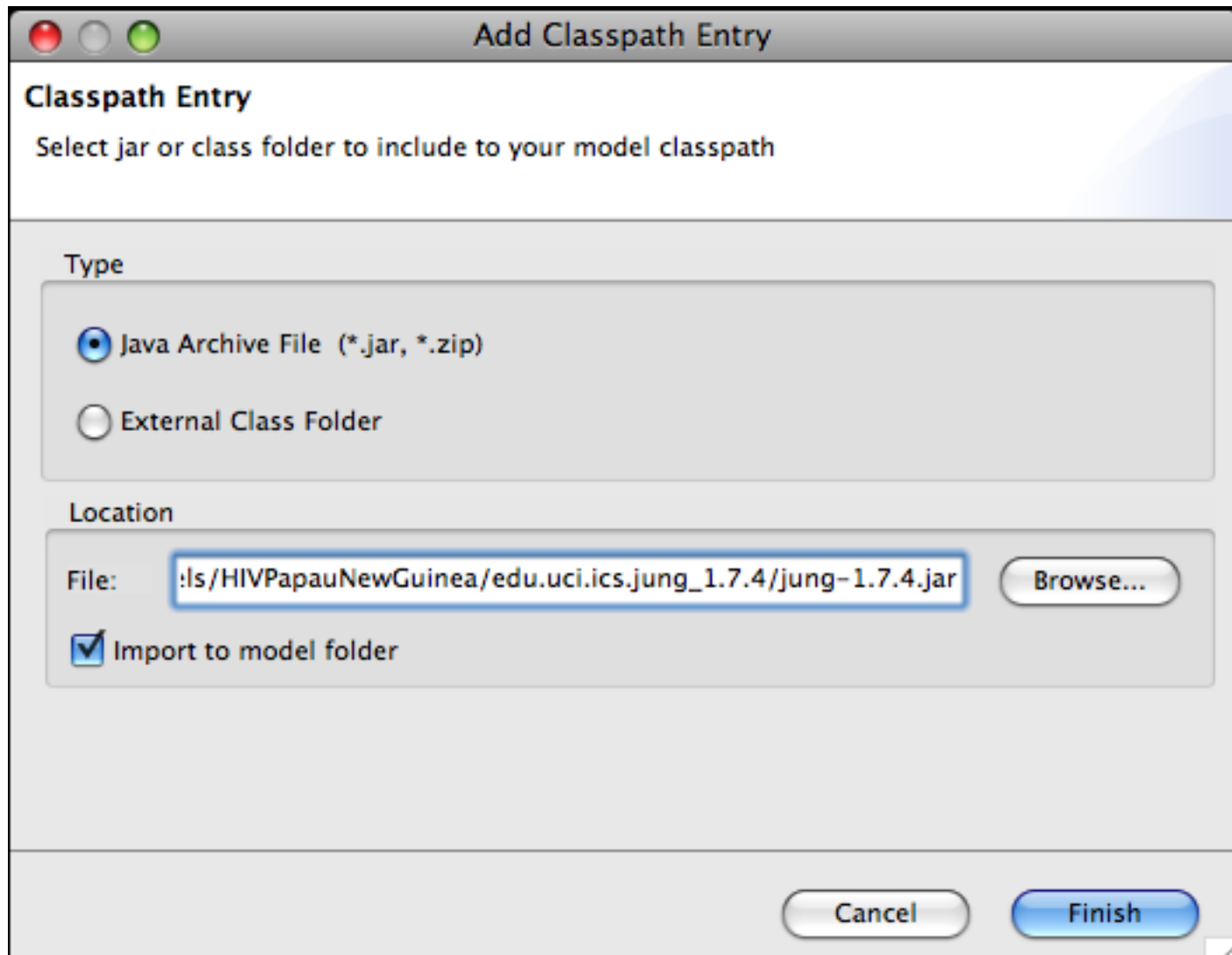
Presentati...

Connectiv...

Enterpris...

More Libraries...

Adding External Libraries 2



Recording of Results

- A frequent modeler need is to record some components of model state over time
 - State variables (e.g. stocks)
 - States of agents
 - Summaries of model state
 - We informally term this a “trajectory file”
- In contrast to Vensim (& other system dynamics packages), trajectory recording is not automatic
- AnyLogic does allow for
 - Definition of *DataSets* that record recent values of parameters
 - Statistics summarizing model state
 - Reporting on values of data sets as a graph or table

Statistics

- A population of agents can have associated statistics that calculate values
- Examples of things that can be computed with using AnyLogic's statistics
 - Count of agents in the population for which certain condition ("predicate") evaluates to true
 - Function of the values of some expression over the population
 - Maximum value
 - Minimum value
 - Average value
 - Sum (total) over population
 - Statistics can be defined as properties of the population

The screenshot displays the AnyLogic Advanced software interface, which is used for modeling and simulation. The interface is divided into several panels:

- Project Tree (Left):** Shows the project structure. The 'Main' model is selected, and its 'Parameters' are listed. The 'person' object is highlighted under 'Embedded Objects'.
- Main Workspace (Center):** Displays a diagram of the model. It includes a 'person' object (represented by a globe icon) and a 'personStat' object (represented by a person icon). The 'personStat' object is connected to the 'person' object.
- Properties Panel (Right):** Shows the properties of the selected 'person - Person' object. The 'Statistics' tab is active, and the 'Type' is set to 'Count'. The 'Name' is 'personStat'.

The 'personStat' object is a statistical object that tracks the count of individuals in the 'person' object. The 'person' object is a stateful object that represents an individual in the model.

[illegible]

Example Statistics

The screenshot displays the AnyLogic Advanced software interface, specifically the 'Person' statechart. The main workspace shows a statechart with several states and transitions, including 'TBProgressionStatechart', 'TBSusceptible', 'TBInfectiousContact', 'WhetherInfected', 'TBTransmission', 'WhetherPrimaryProgression', 'PrimaryProgression', 'UnDiagnosedActiveTB', 'NaturalTBRecovery', 'LTBI', 'Reactivation', 'DeathFromUndiagnosedTB', 'Death', 'UndiagnosedTBInfectionContact', 'Diagnosis', 'DiagnosedActiveTB', and 'TreatmentMediatedTBRecovery'. The 'person' state is highlighted.

The 'Properties' window for the 'person - Person' state is open, showing the 'Statistics' tab. The 'Name' field is set to 'CountSusceptible'. The 'Type' is set to 'Count'. The 'Expression' field is empty. The 'Condition' field contains the code: `item.TBProgressionStatechart.isStateActive(Person.TBSusceptible);`. An 'Add Statistics' button is visible below the condition field.

The 'Project' pane on the left shows the project structure, including 'Plain Variables', 'Dynamic Variables' (Age, Aging, Weight), and 'Statecharts' (TBProgressionStatechart). The 'Problems' pane at the bottom left is empty.

The 'Model' pane on the right shows a list of model elements, including 'Parameter', 'Flow Aux ...', 'Stock Vari...', 'Event', 'Dynamic ...', 'Plain Vari...', 'Collectio...', 'Function', 'Table Fun...', 'Port', 'Connector', 'Entry Point', 'State', 'Transition', 'Initial Stat...', 'Branch', 'History St...', 'Final State', and 'Environ...'. The 'Action' button is highlighted.

Output: Datasets

The screenshot displays the AnyLogic Advanced [EDUCATIONAL USE ONLY] interface. The main workspace shows a project named 'TBv1*' with a 'Main' agent. The 'Main' agent contains several components:

- Parameters:**
 - DaysFromDiagnosisUntilRecovery: 30
 - DaysUntilDiagnosis: 60
 - DiagnosedPerDayTBContactRatePerNetworkContact: .10
 - LikelihoodOfPrimaryProgression: .10
 - PerContactTBInfectionProbability: .50
 - UndiagnosedPerDayTBContactRatePerNetworkContact: .10
- Functions:**
 - PersonWithMaxDegree
- Environments:**
 - environment
- Embedded Objects:**
 - person
- Analysis Data:**
 - person_presentation
 - TimePlotAgentCount
- Person:**
 - Parameters:
 - DaysPerTimeUnit: 365.25
 - Ethnicity: 1
 - MeanDaysToNaturallyClearInfection: 180.00

The 'dsSusceptibleCount' data set is configured with the following settings:

- Name:** dsSusceptibleCount
- Show Name:** ☒
- Ignore:** ☐
- Public:** ☐
- Show At Runtime:** ☒
- Use time as horizontal axis value:** ☒
- Horizontal axis value:** [Empty text box]
- Vertical axis value:** person.CountSusceptible()
- Keep up to:** 1000 latest samples
- Do not update automatically:** ☒
- Update automatically:** ☐
- Begin at time:** 0.0
- Recurrence time:** 1
- October 29, 2009 2:07:08 AM** (dropdown menu)

The 'Problems' panel at the bottom left shows a table with columns 'Description' and 'Location'.

Description	Location

Datasets

- Datasets store recent values of some quantities from the model
- Datasets can be exported easily using custom code
 - This can simply call the dataset's to string method

Example code

```
FileOutputStream fos = new  
    FileOutputStream(strOutputFilename);  
PrintStream p = new PrintStream(fos);  
p.println(datasetName.toString()); // outputs  
    comma delimited values
```

Dataset Properties

The screenshot displays the AnyLogic Advanced [EDUCATIONAL USE ONLY] interface. The main workspace shows a model diagram with a 'TreatmentQueue' block and a 'datasetAbsolutePrevalence' block. The 'datasetAbsolutePrevalence' block is highlighted, and its properties are shown in the 'Properties' panel on the right. The 'Properties' panel has tabs for 'General' and 'Description'. The 'General' tab is active, showing the following settings:

- Name: datasetAbsolutePrevalence
- ☒ Show Name
- ☐ Ignore
- ☐ Public
- ☒ Show A
- ☒ Use time as horizontal axis value
- Horizontal axis value: [empty field]
- Vertical axis value: $((double) \text{ nInfectious}) / ((double) \text{ TotalPopulation})$
- Keep up to: 5000 latest samples
- ☐ Do not update automatically
- ☒ Update automatically
- Begin at time: 0.0
- Recurrence time: 1
- June 11, 2008 2:54:05 AM

The 'Problems' panel on the left shows several error messages:

- The constructor DataSet() is undefined
- The constructor DataSet() is undefined
- The constructor DataSet() is undefined
- The constructor DataSet() is undefined
- Engine.log cannot be resolved
- mViewer cannot be resolved
- mViewer cannot be resolved

Chart Use of Datasets

