Discrete Intra-Agent Dynamics: Statecharts & Messaging

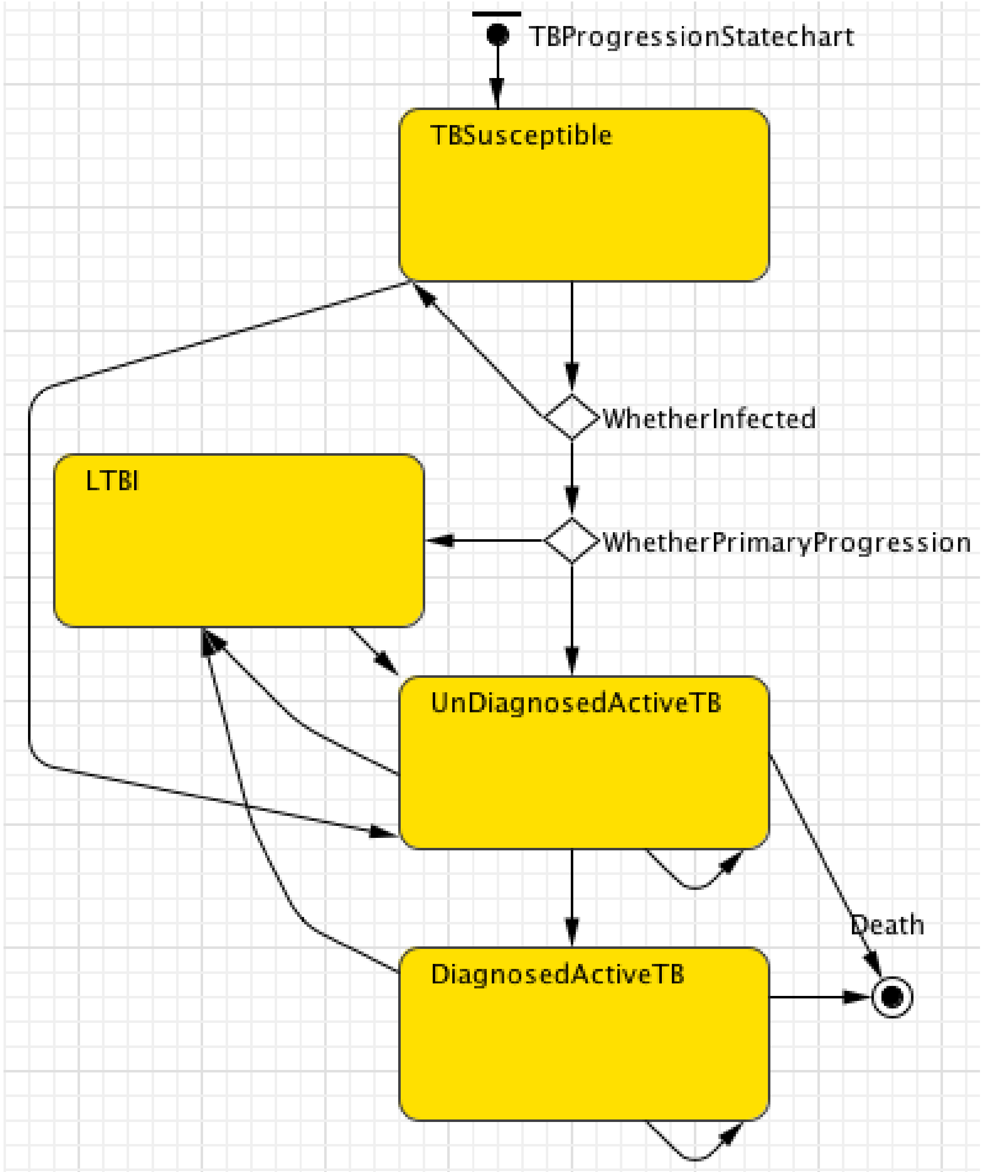
Nathaniel Osgood

February 10, 2011

Discrete Agent Dynamics

- Frequently we can represent agent behaviour using as transitioning among a set of states in a state chart
- For a given simple statechart, the agent is in exactly one state at a time
- Fixed transitions between states define possible evolution
- The transitions between states occur instantaneously, based on some condition

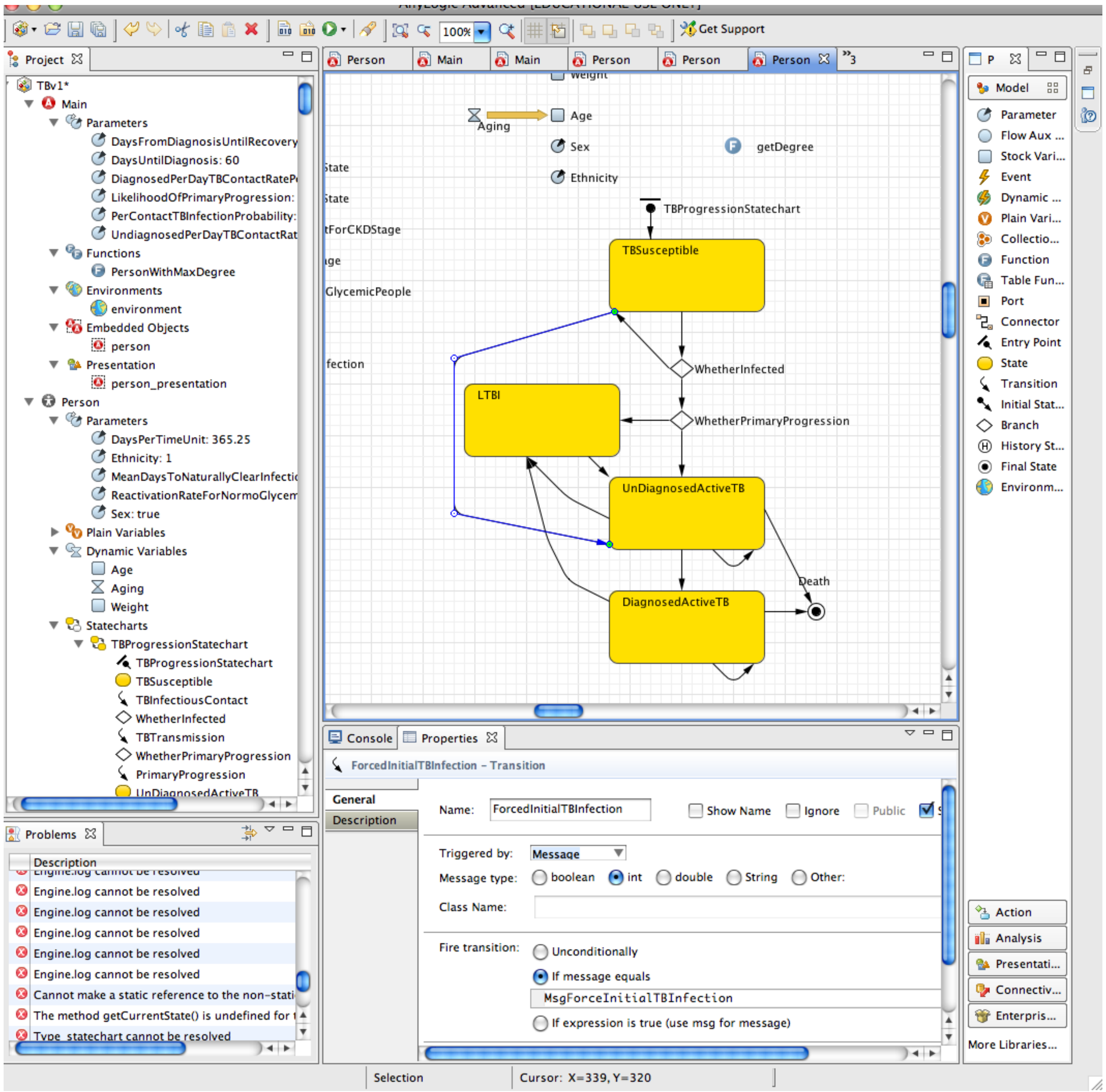
Example State Transition Diagram



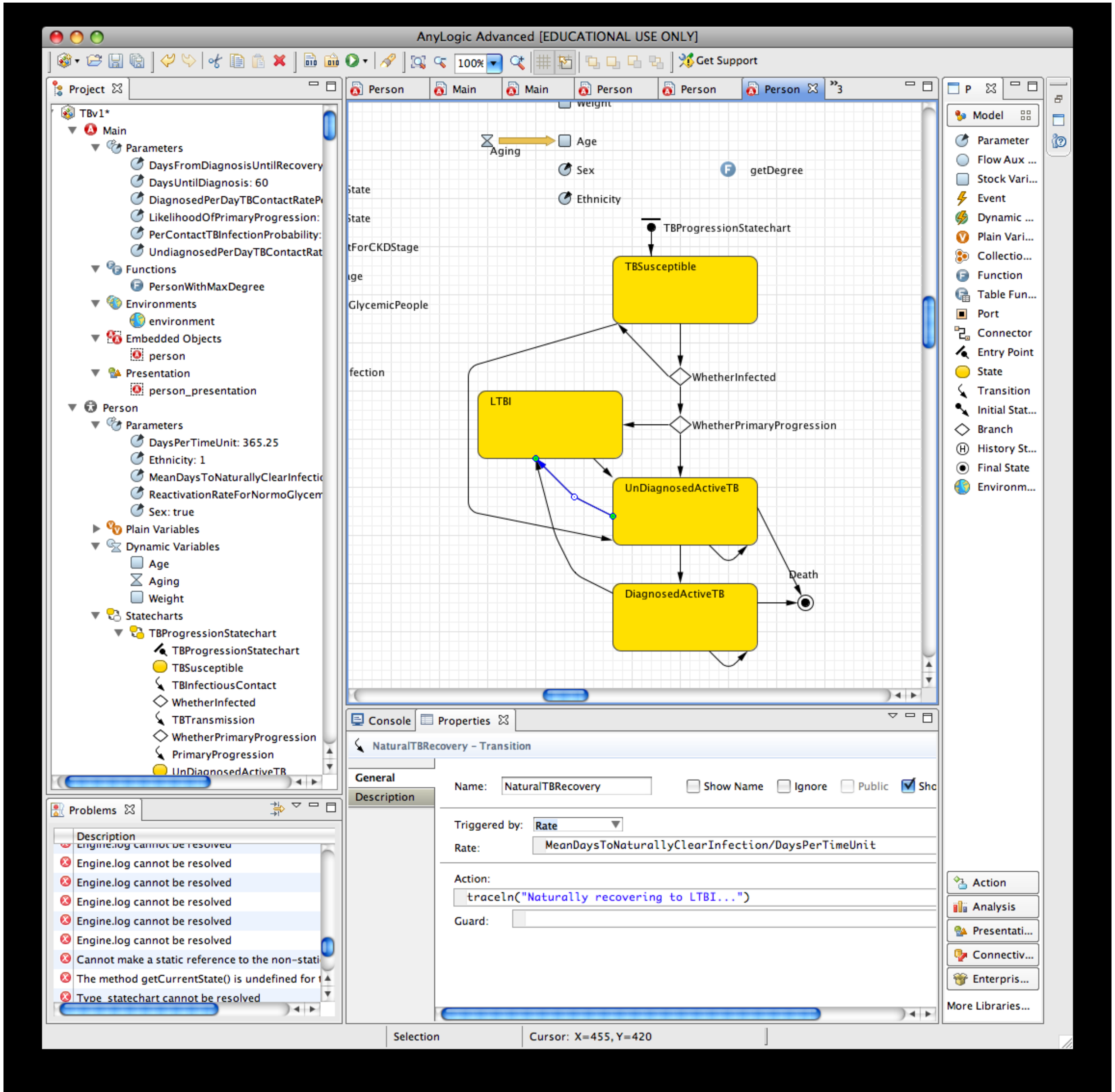
Discrete Agent Dynamics: Transitions

- Many transition conditions are possible
  - Timeout: Spending some period of time in the state
  - Fixed rate: Leave state with some fixed change per unit time
    - This is similar to “first order interarrival time”, and is conceptually linked to the operation of first-order delays in stock & flow diagrams
  - Variable rate: If desired, we can change the rate over time – *but Anylogic only “notices” changes when eg agent re-enters the state*
  - Message received: We can transition when a message (any message or particular type of message) is received
  - Predicate: Only transition when condition becomes true
- These transitions can be conditionally “routed” via branches
  - Conditions can determine to what destination state a particular transition will travel

Transition Type: Message Triggered



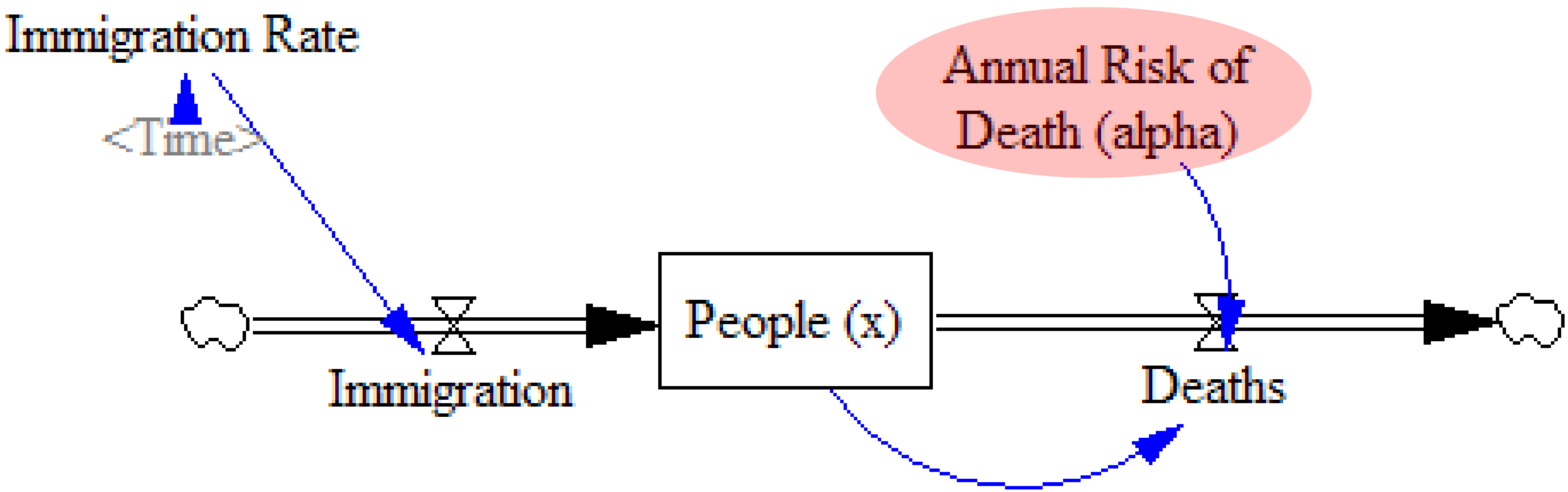
Transition Type: Fixed Rate



Rates & Flows

- We’ve seen fixed rates before – in the form of “transition rates” in System Dynamics models
- Within the System Dynamics model, a flow out of a stock was commonly set by the multiplication of the
  - Stock
  - Some rate of transition
- We use different names for these rates
  - “Transition rates”
  - “Likelihood of transition per *Unit Time*”
  - Transition (e.g. “infection”, “mortality”) “hazard”

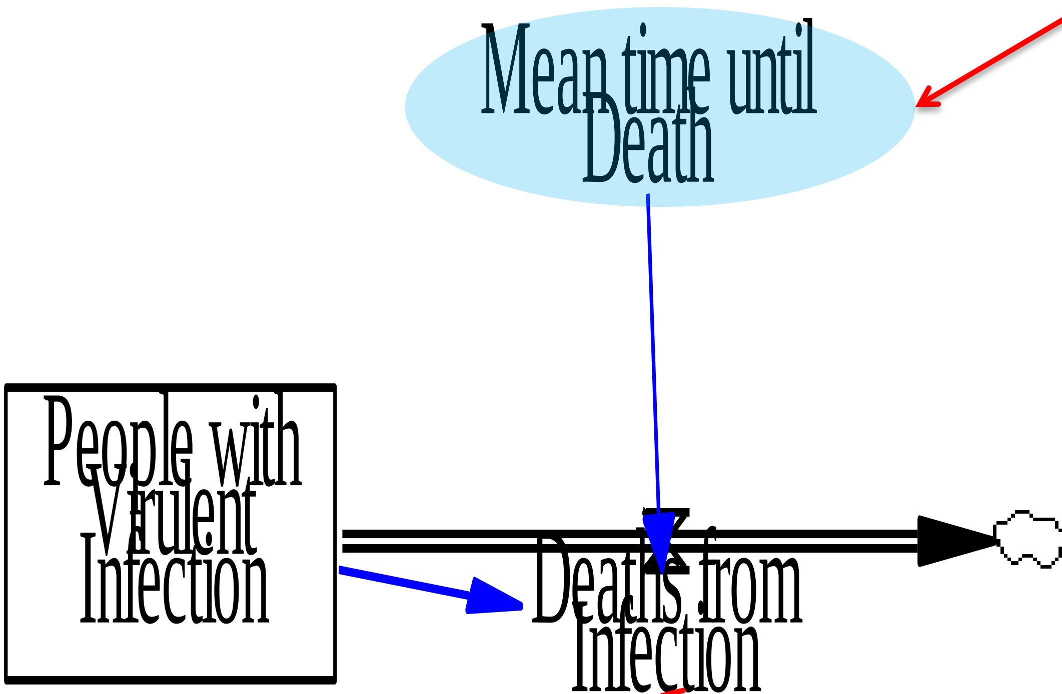
Example Fixed Transition Rate/Hazard



Example Fixed Transition Rate/Hazard

The transition rate is the *reciprocal* of this number i.e.

$$\frac{1}{Mean\ time\ until\ Death}$$



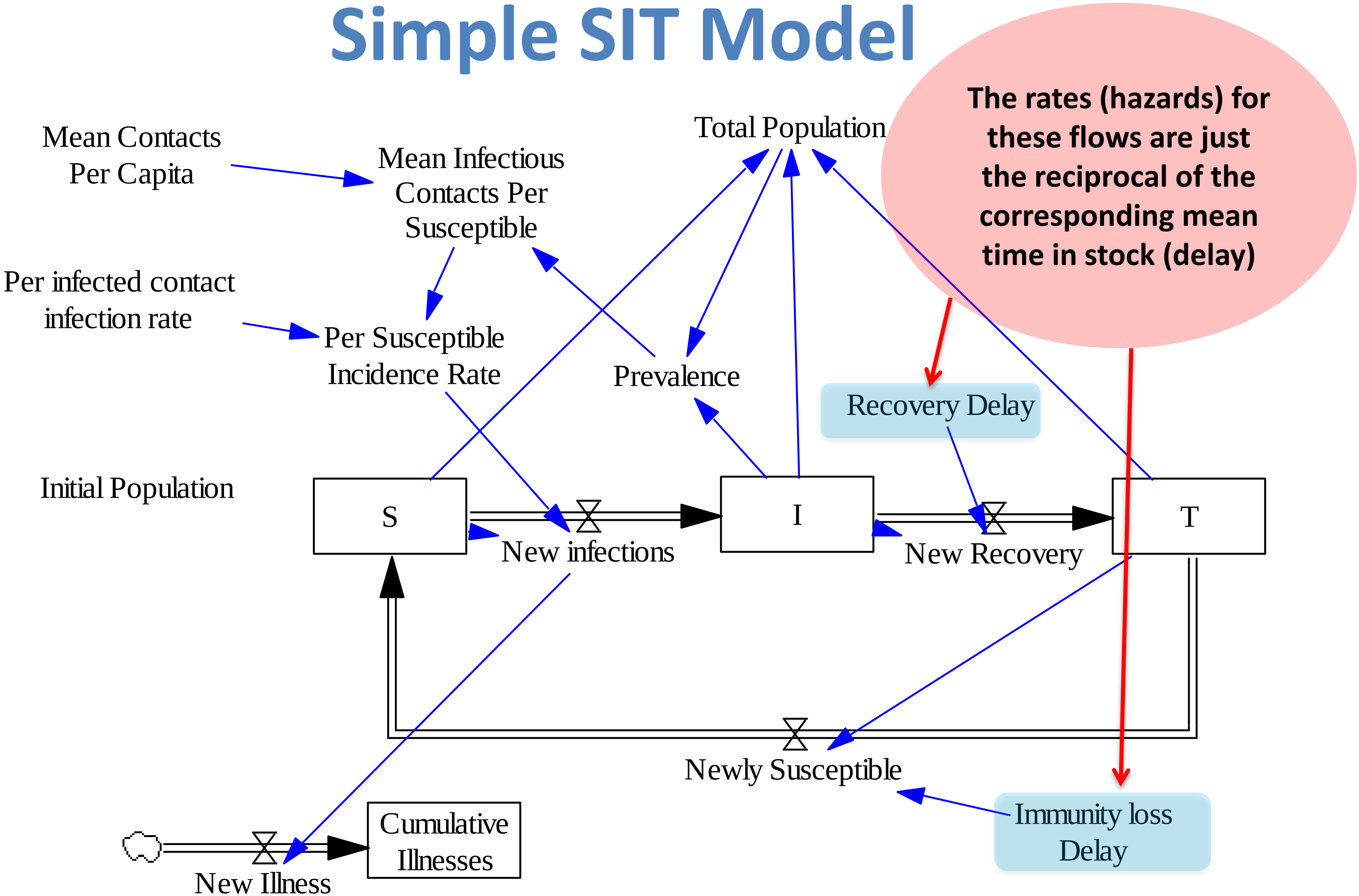
People with Virulent Infection/Mean time until Death

=

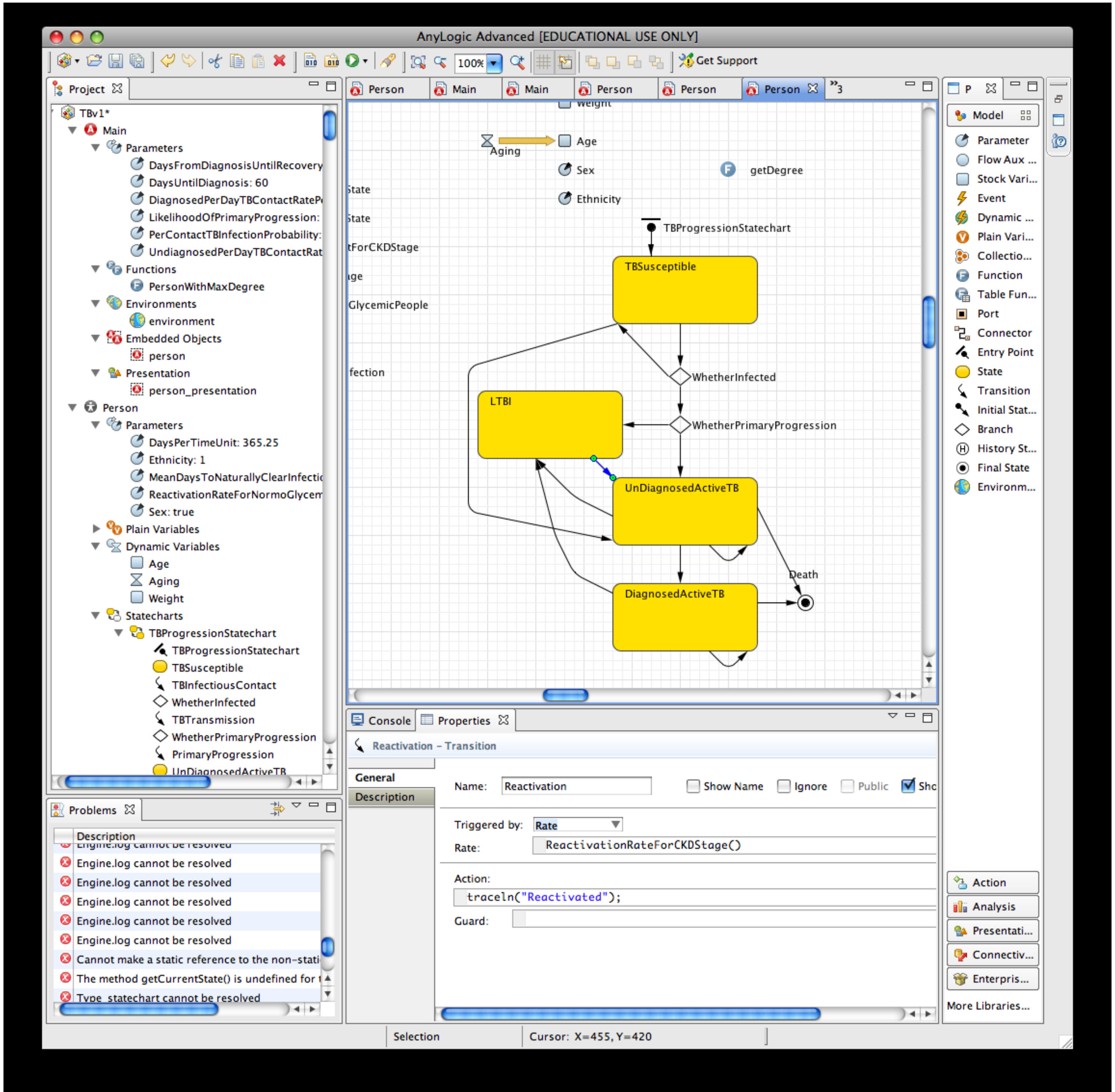
People with Virulent Infection*(1/Mean time until Death)

i.e. People with Virulent Infection*Rate

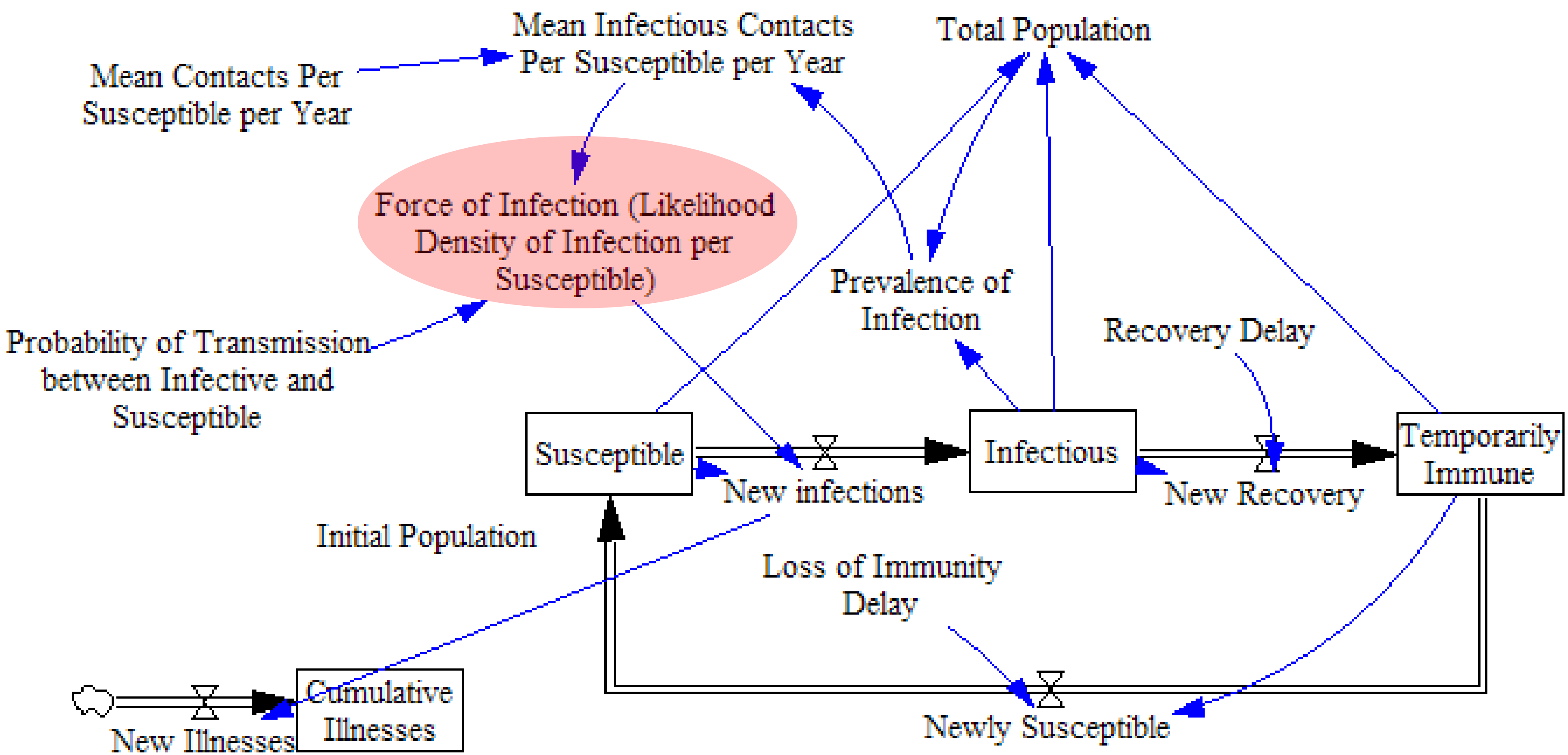
First Order Delays in Action: Simple SIT Model



Transition Type: Variable Rate



Example Transition Rate/Hazard

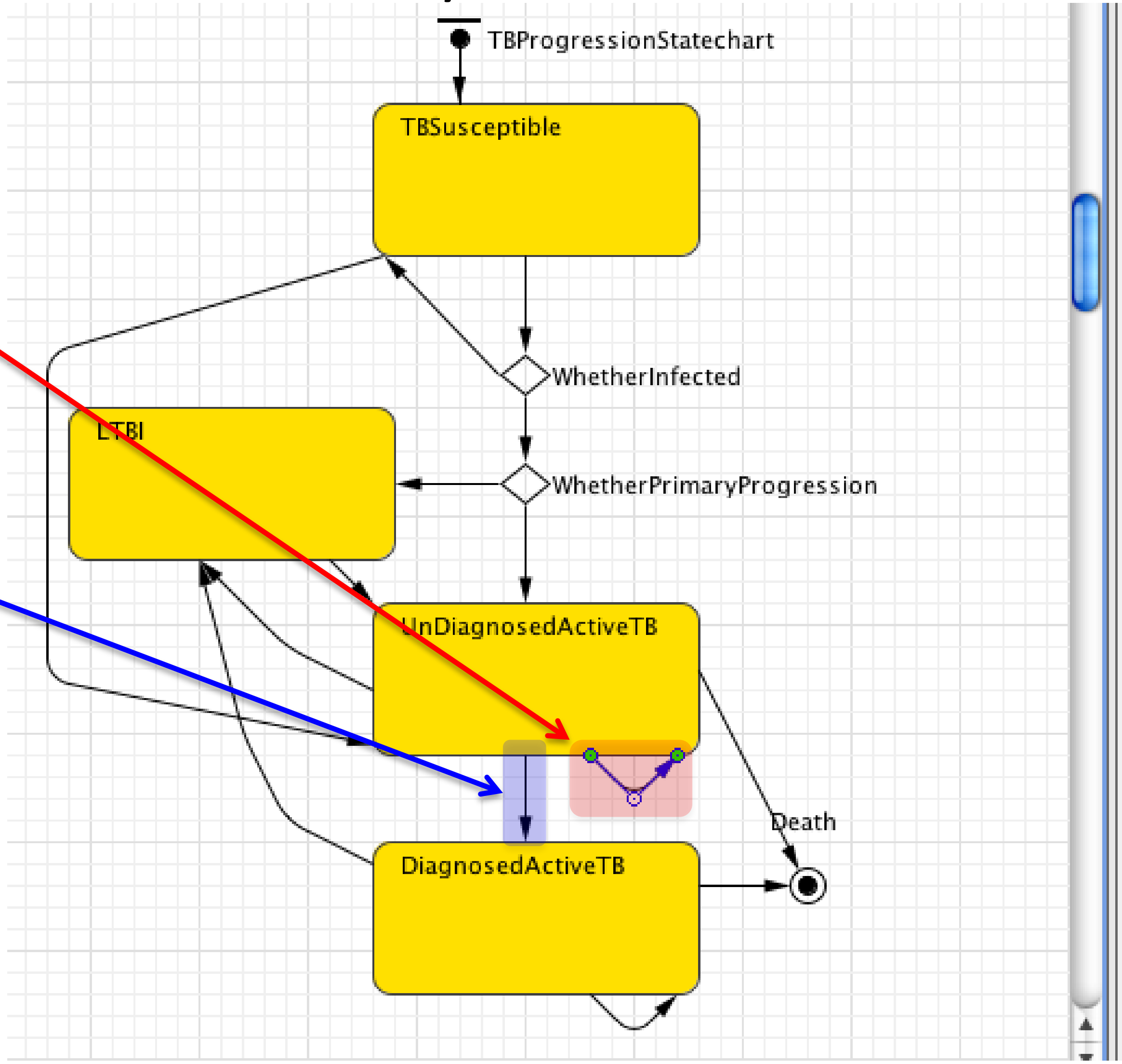


Fixed Rates: Transition "Hazards"

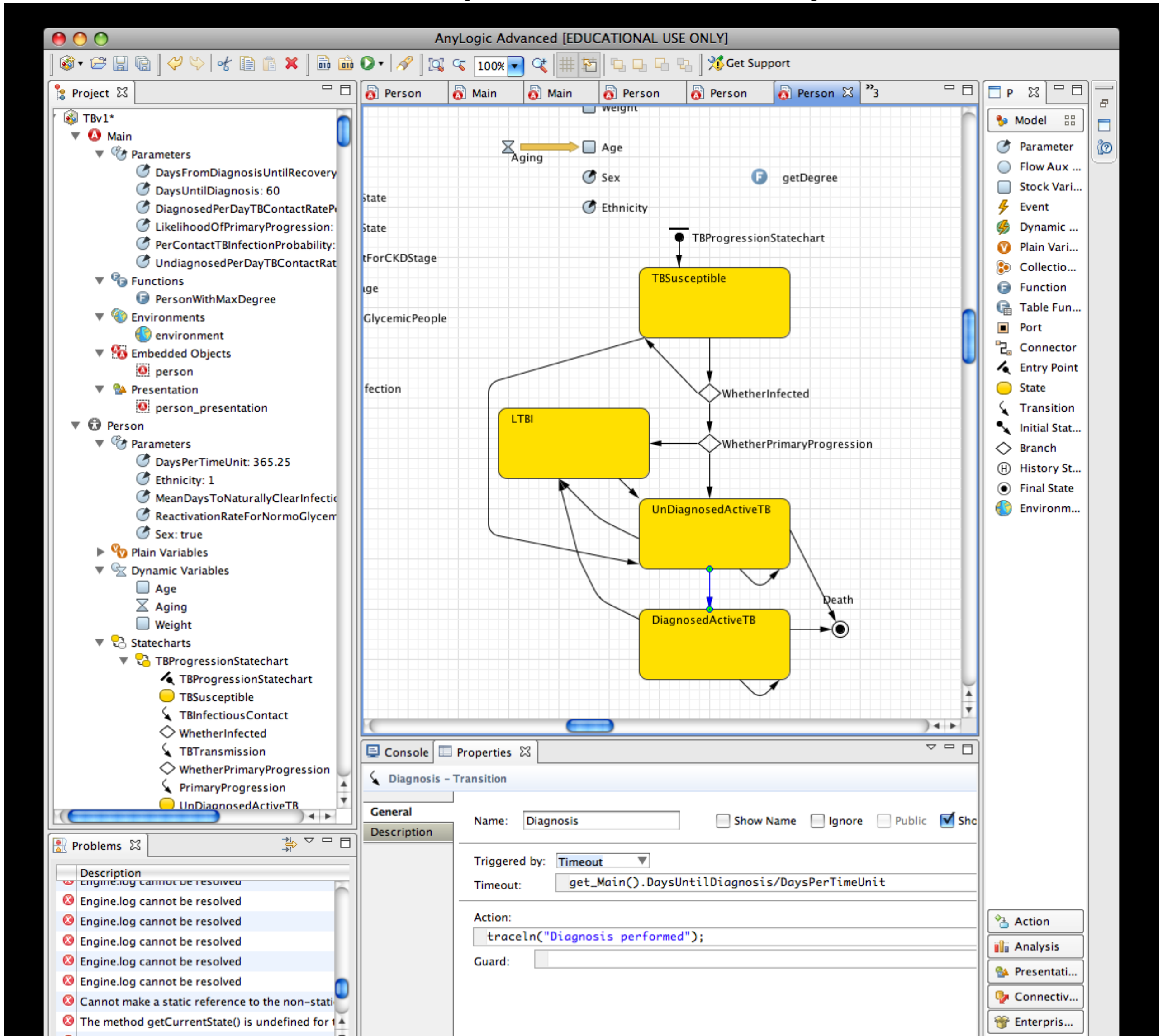
- With "fixed rates", we are specifying rates of transitions
- Because we are dealing with the chance that each individual transitions, we don't need to multiply by the number of people at risk
  - Here, there is just 1 person at risk!
- As in SD models, these rates can change over time, but the statechart needs to be "made aware" of these changes
  - Leave & go back into current state (circular transition)
  - Likely: trigger "change" event in Agent (see manual)

Special Elements: Self-Transition
(Use if Wish To Have State Register Changing Out-transition rates)

The self-transition will "make the state realize" that the rate associated with any out transition (e.g. this one) has changed

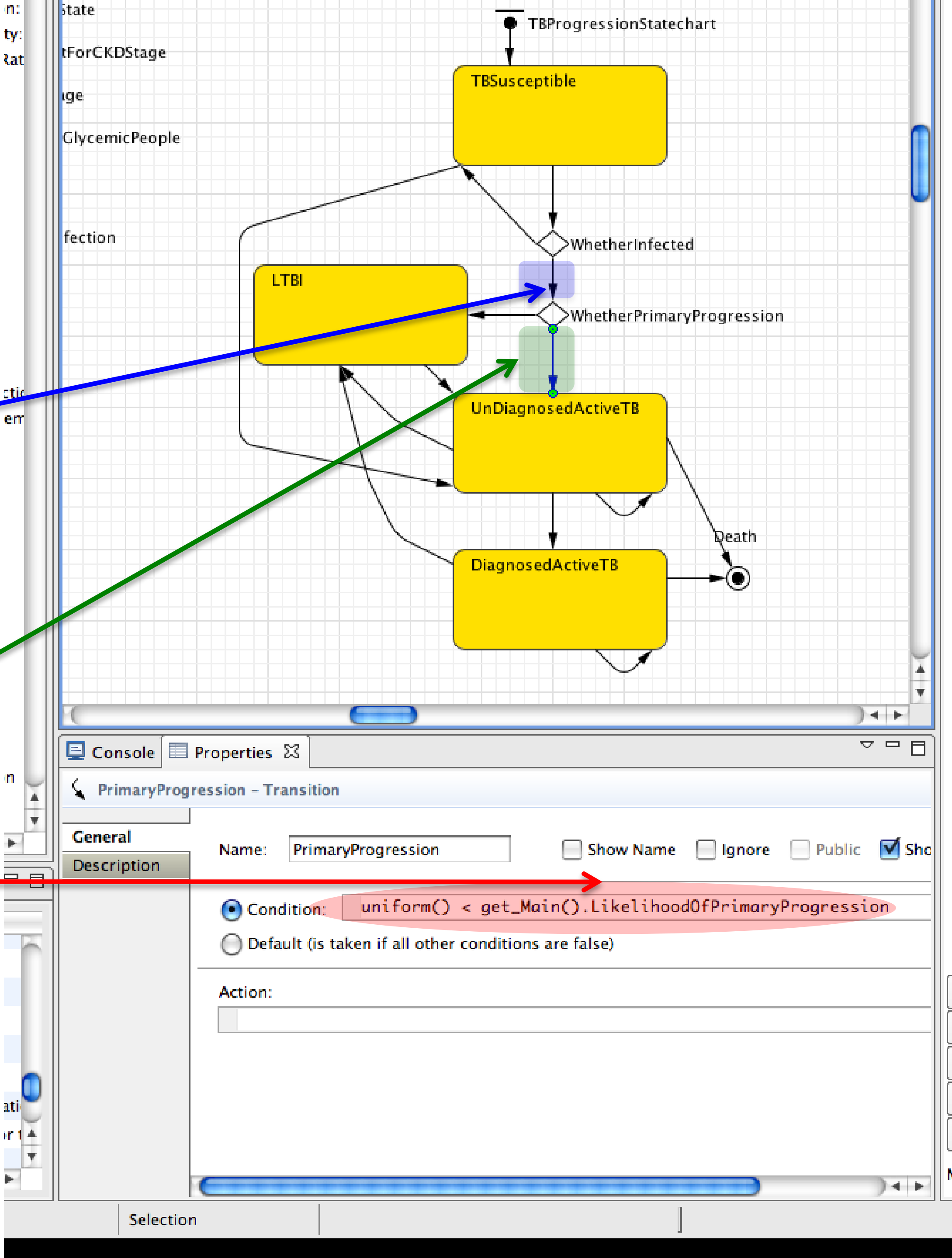


Transition Type: Fixed Residence Time (Timeout)

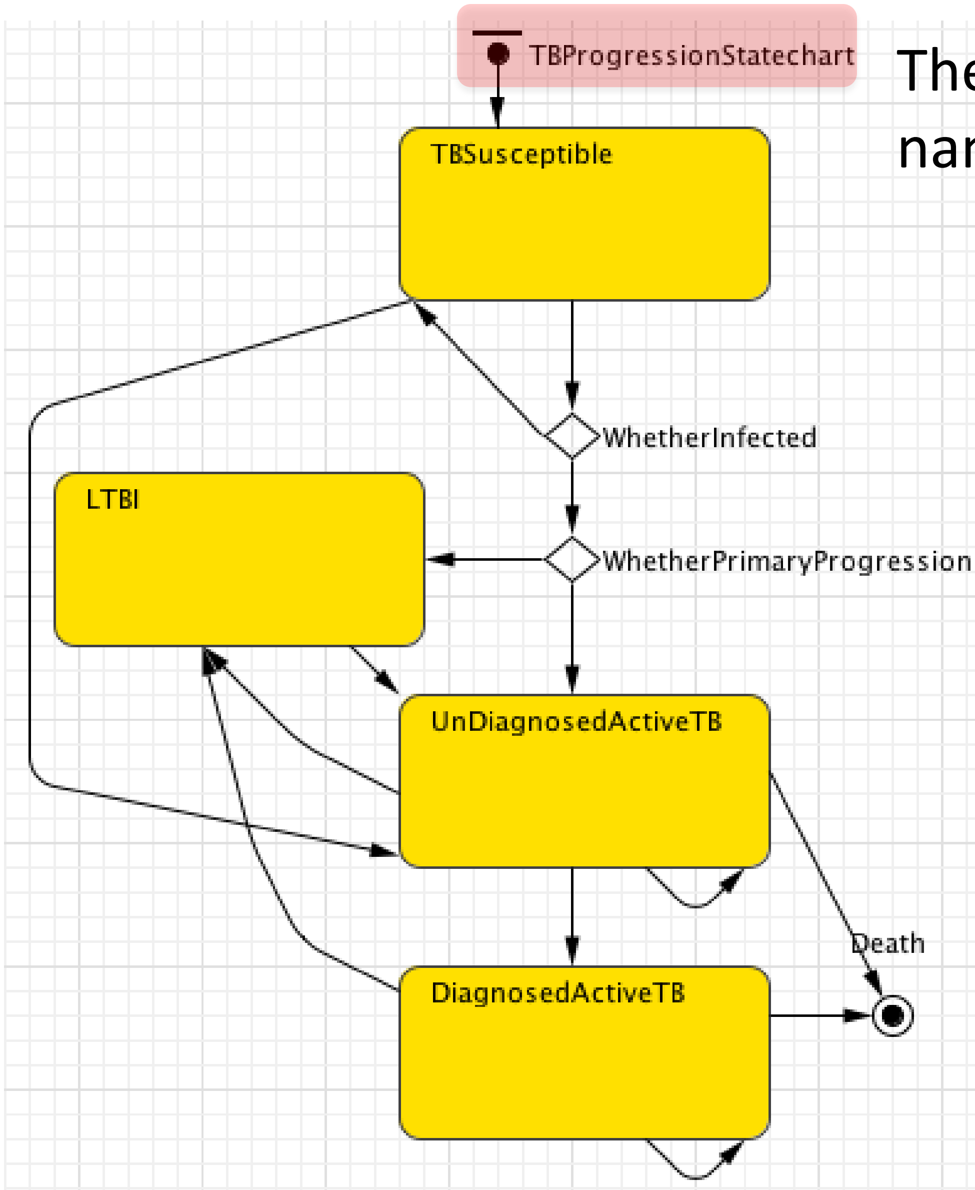


Example Conditional Transition

The incoming transition into "WhetherPrimaryProgression" will be routed to this outgoing transition if this condition is true

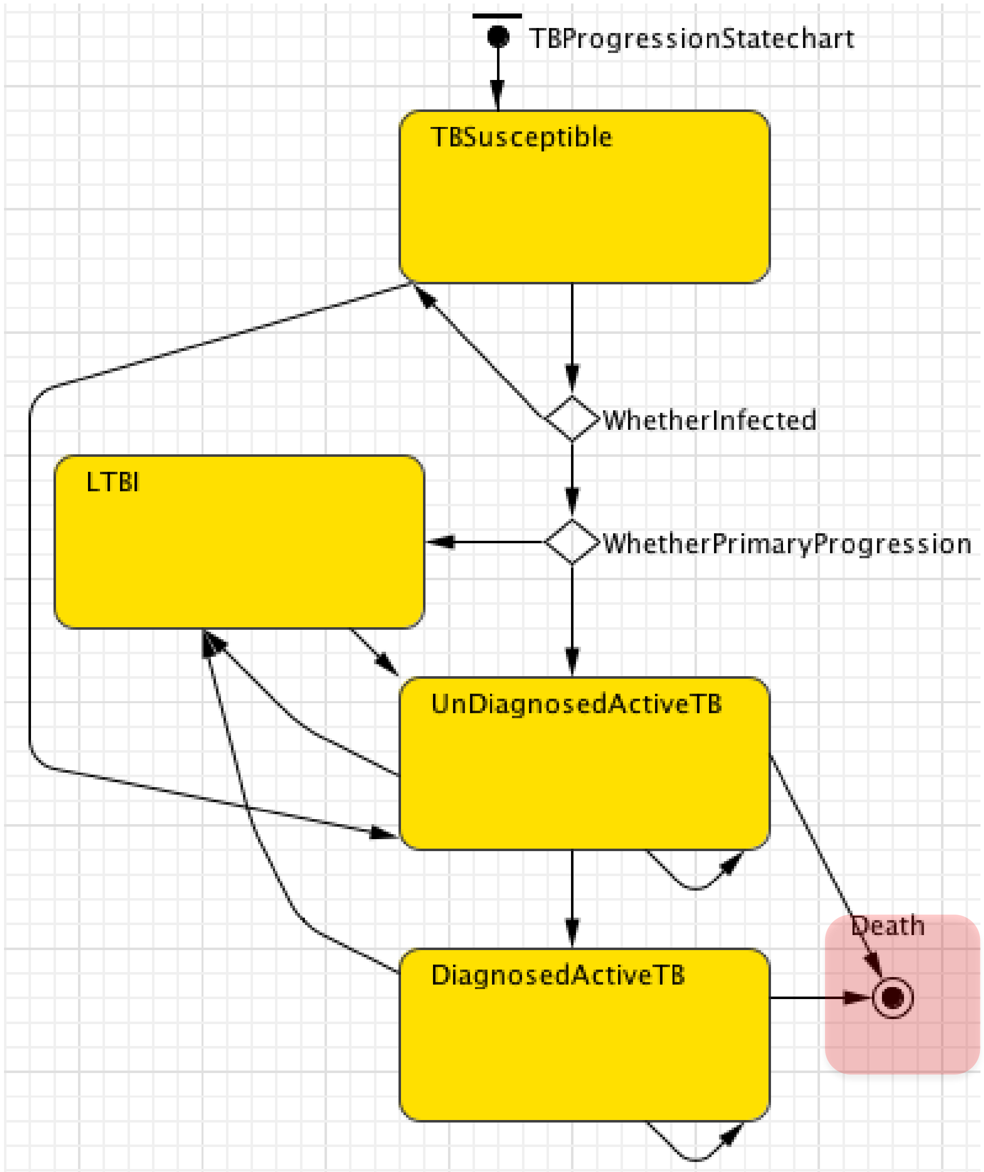


Special Elements: Entry Point

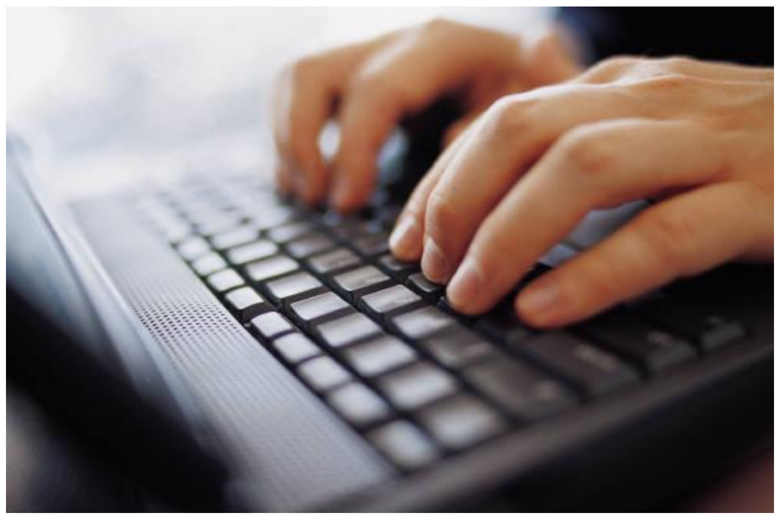


The associated text is the name of the statechart!

Special Elements: Exit Point



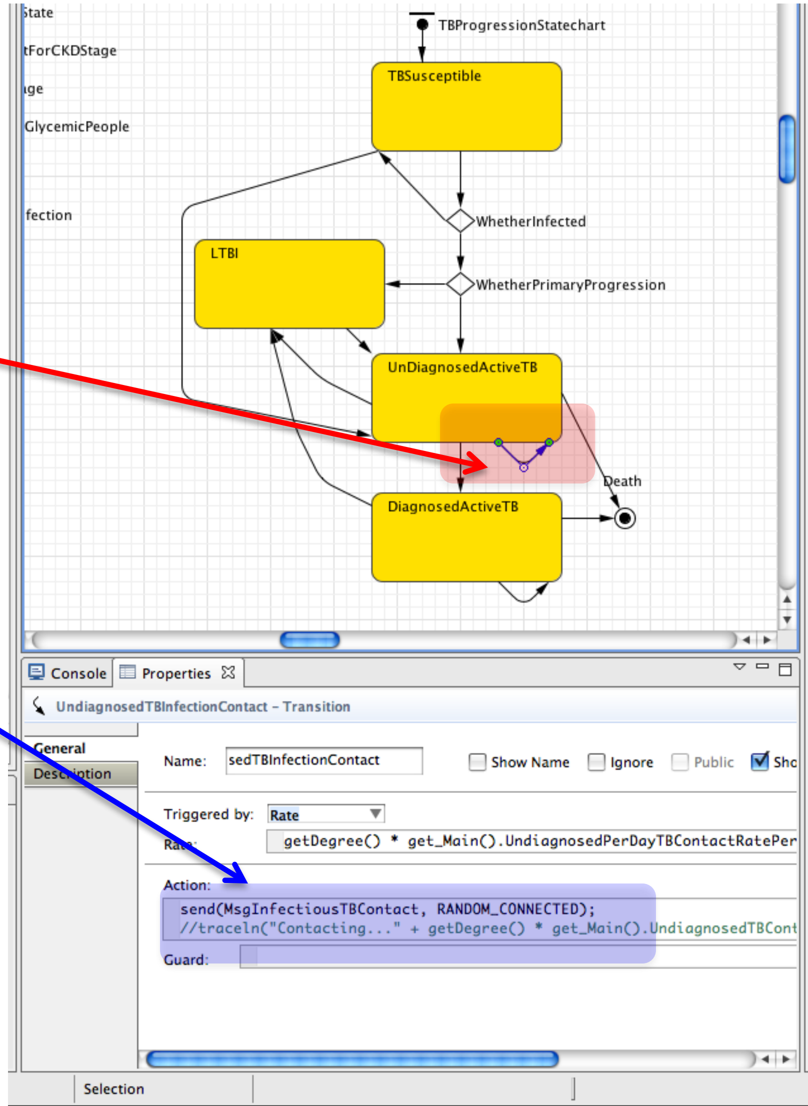
Hands on Model Use Ahead



Load model: TBv1.alp

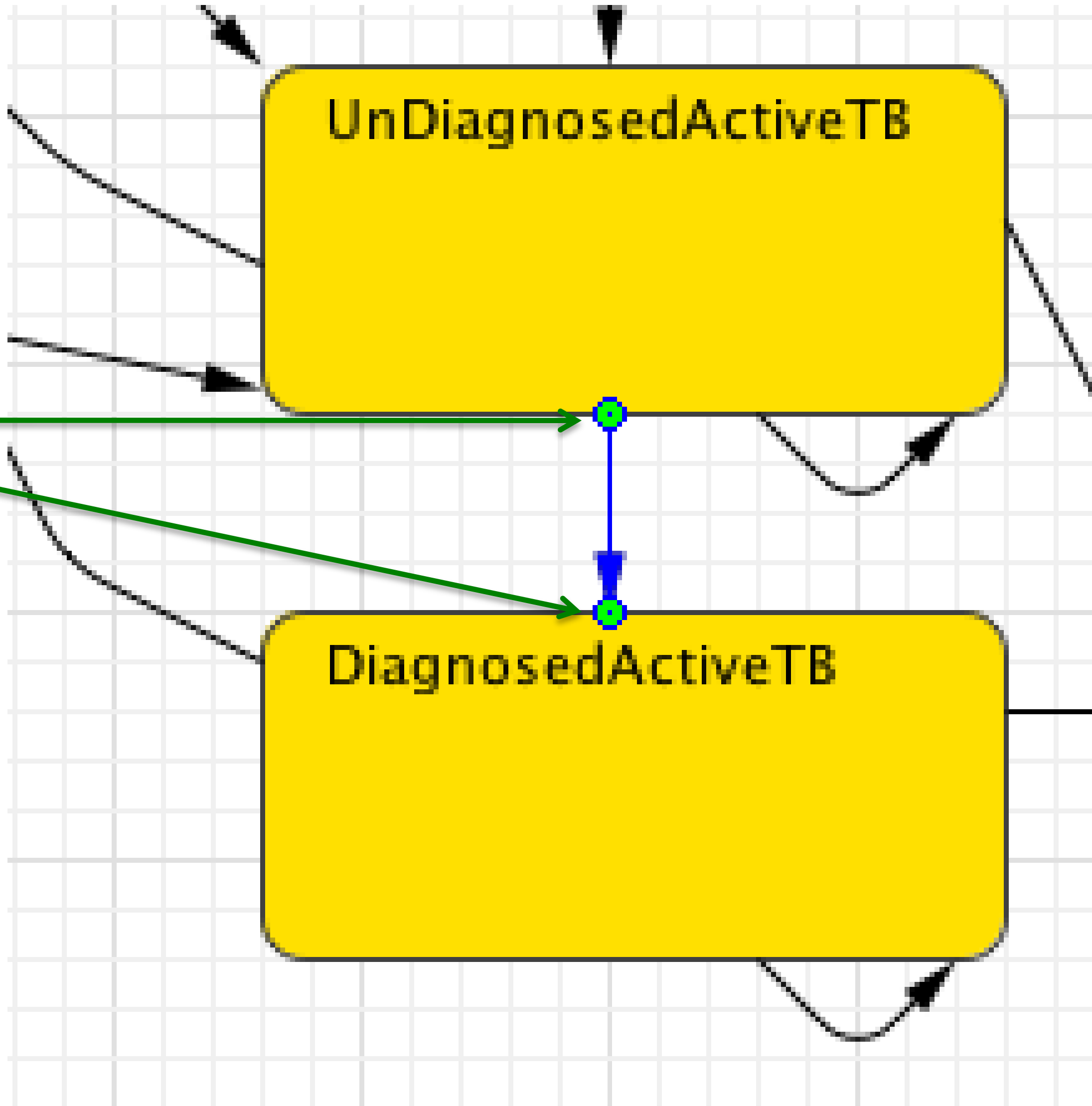
Special Elements: Self-Transition
(Use if Wish To Trigger an Action w/o Leaving State)

The self-transition will invoke this action when it occurs



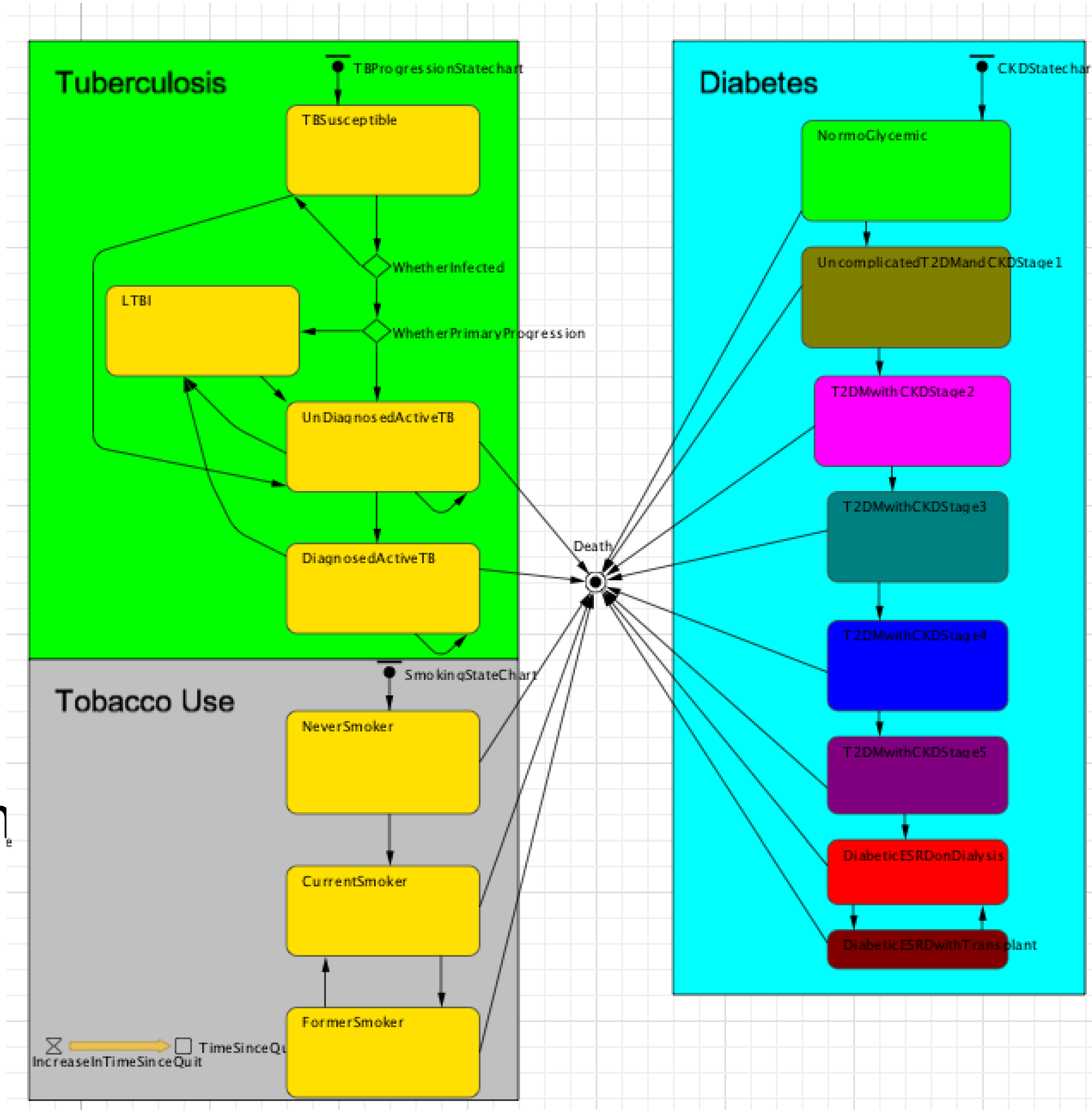
Tip: Confirming Transition Connectivity

- Ensure that both sides of the transition show green circles when connected
  - Otherwise, may appear connected but will actually be disconnected!



Parallel Statecharts

- By default, each statechart evolves independently.
- If coupling is desired, can make transitions/action s dependent on state of other statecharts

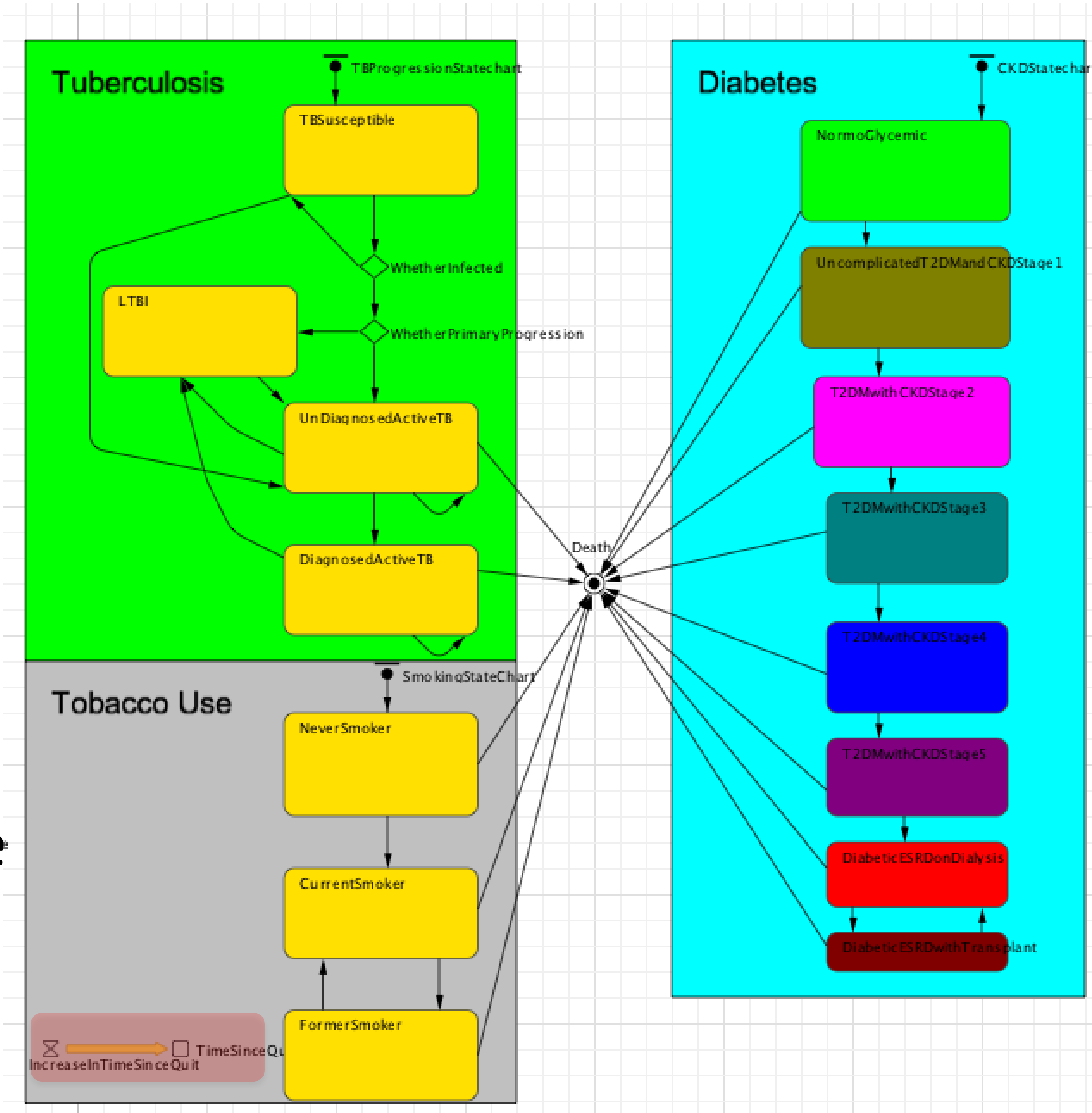


Comparison with Aggregate Stock & Flows

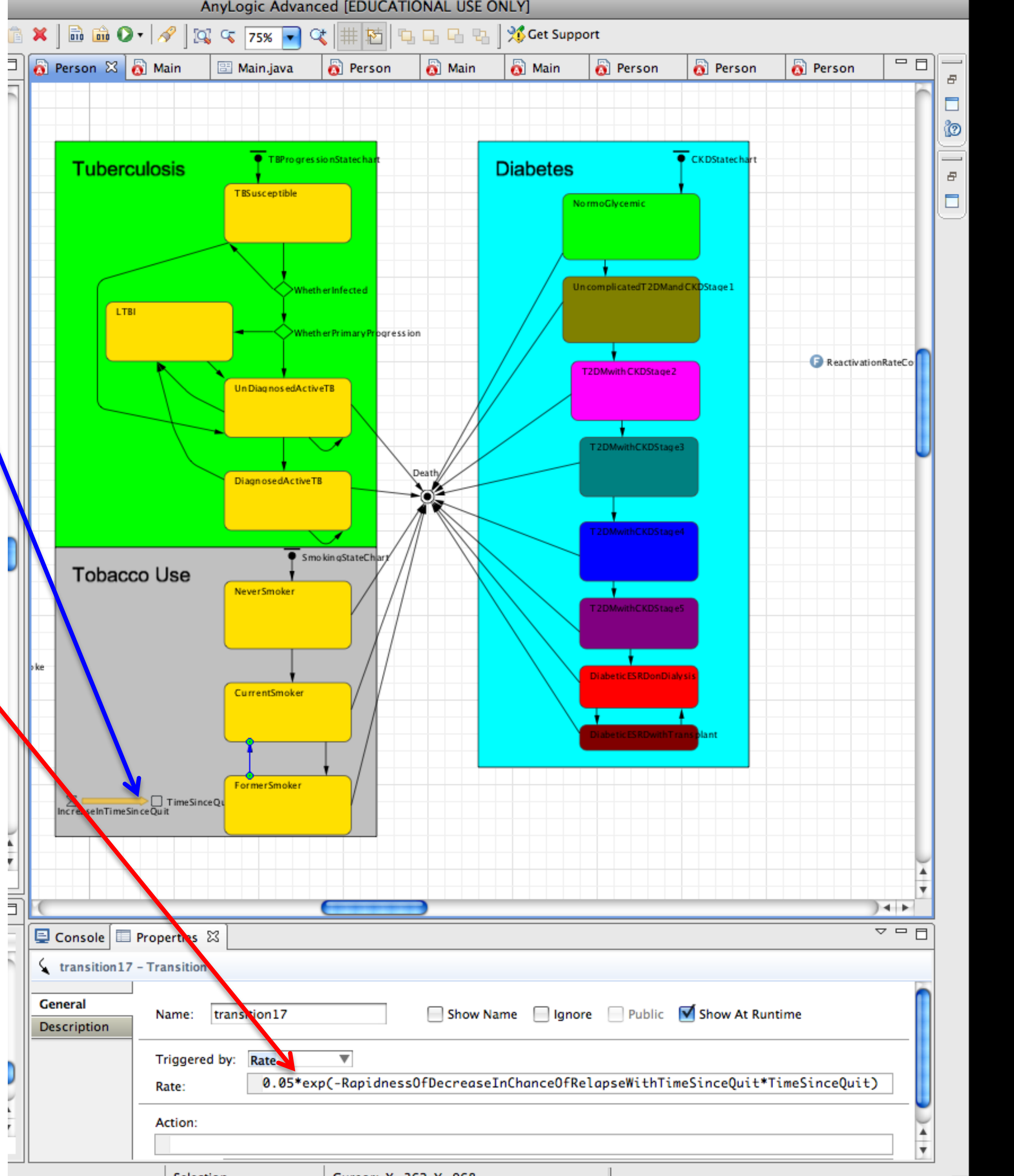
- As for aggregate stocks & flow, individuals' states are discrete
- Unlike aggregate stocks & flows
  - One state within a given statechart is active at a time
  - For parallel flows (e.g. comorbidities), there is no need for considering all combinations of the possible states
  - We can keep track of how long an individual is in a given state & adjust the transition rate accordingly

Parallel Transitions

- Example recording the residence time in a state (via a stock with unit inflow -- i.e. just accumulates the time present in that state)

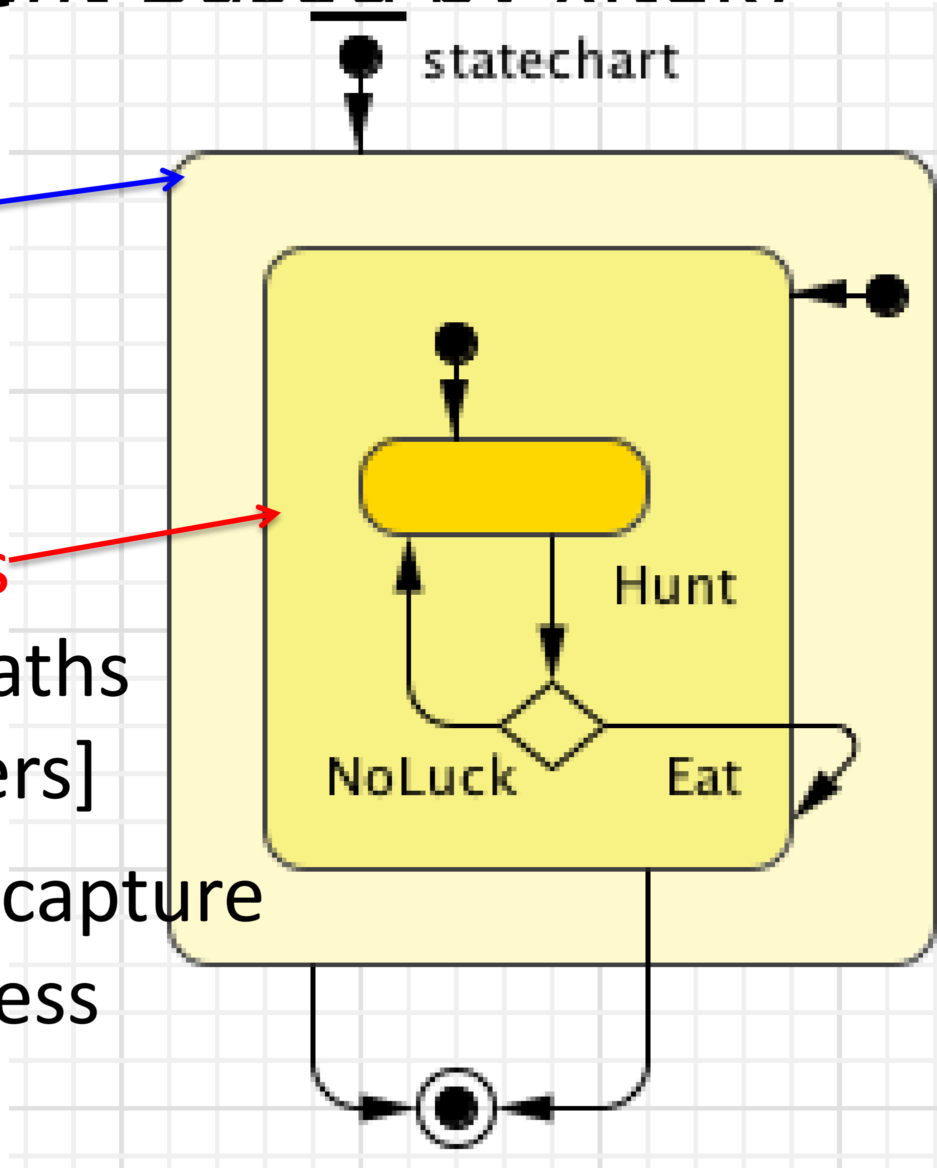


- The residence time in the state determines the transition rate out of that state.
- Transition rates depending on residence time are generally not possible with aggregate models

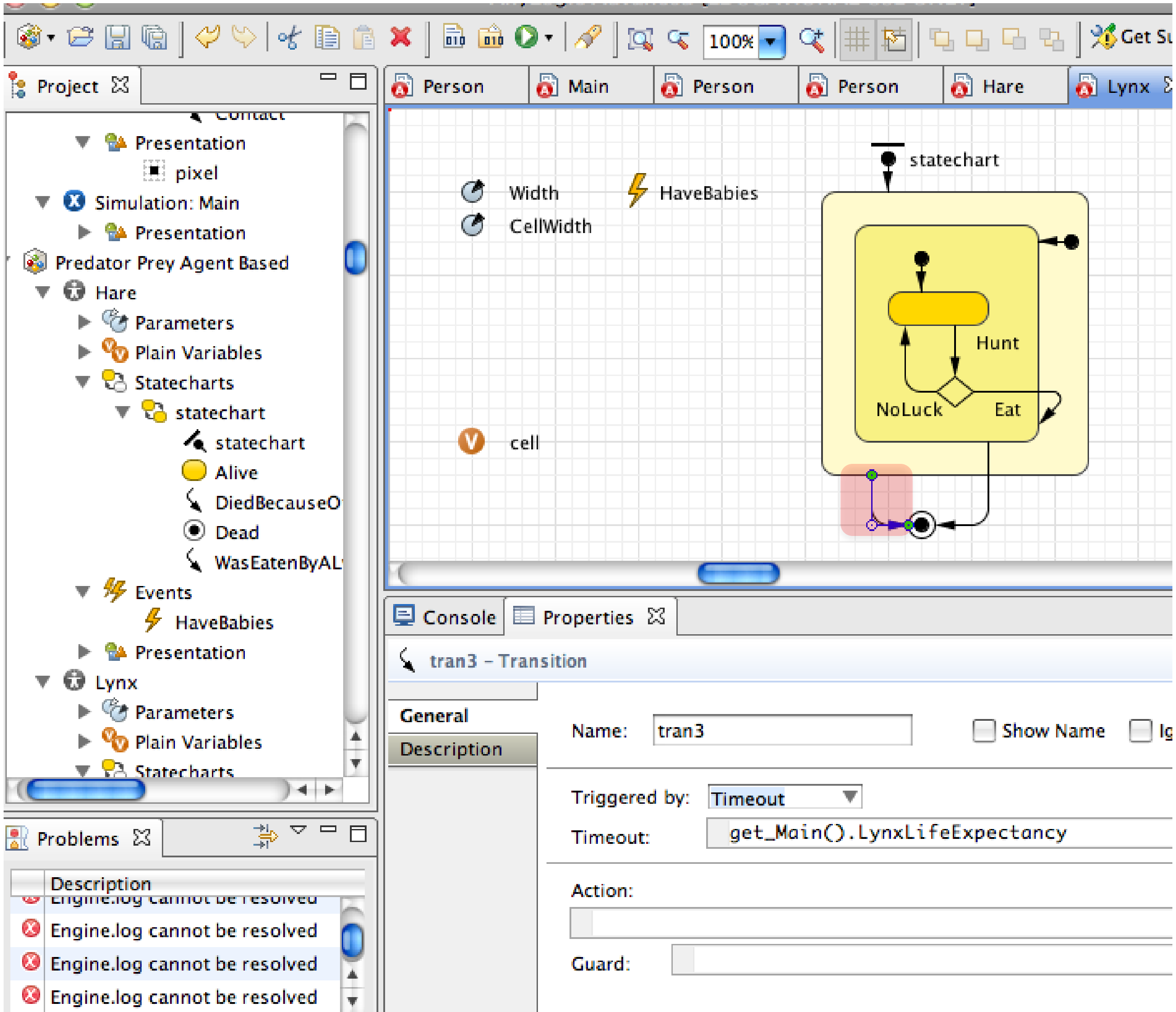


Advanced Element: Hierarchical States (*Predatory Prey Agent Based* bv xitek)

- The outermost state captures time since born (for natural deaths)
- The middle-state captures time since last ate (for deaths by hunger). [Eating reenters]
- The inner state transition capture hunting frequency & success

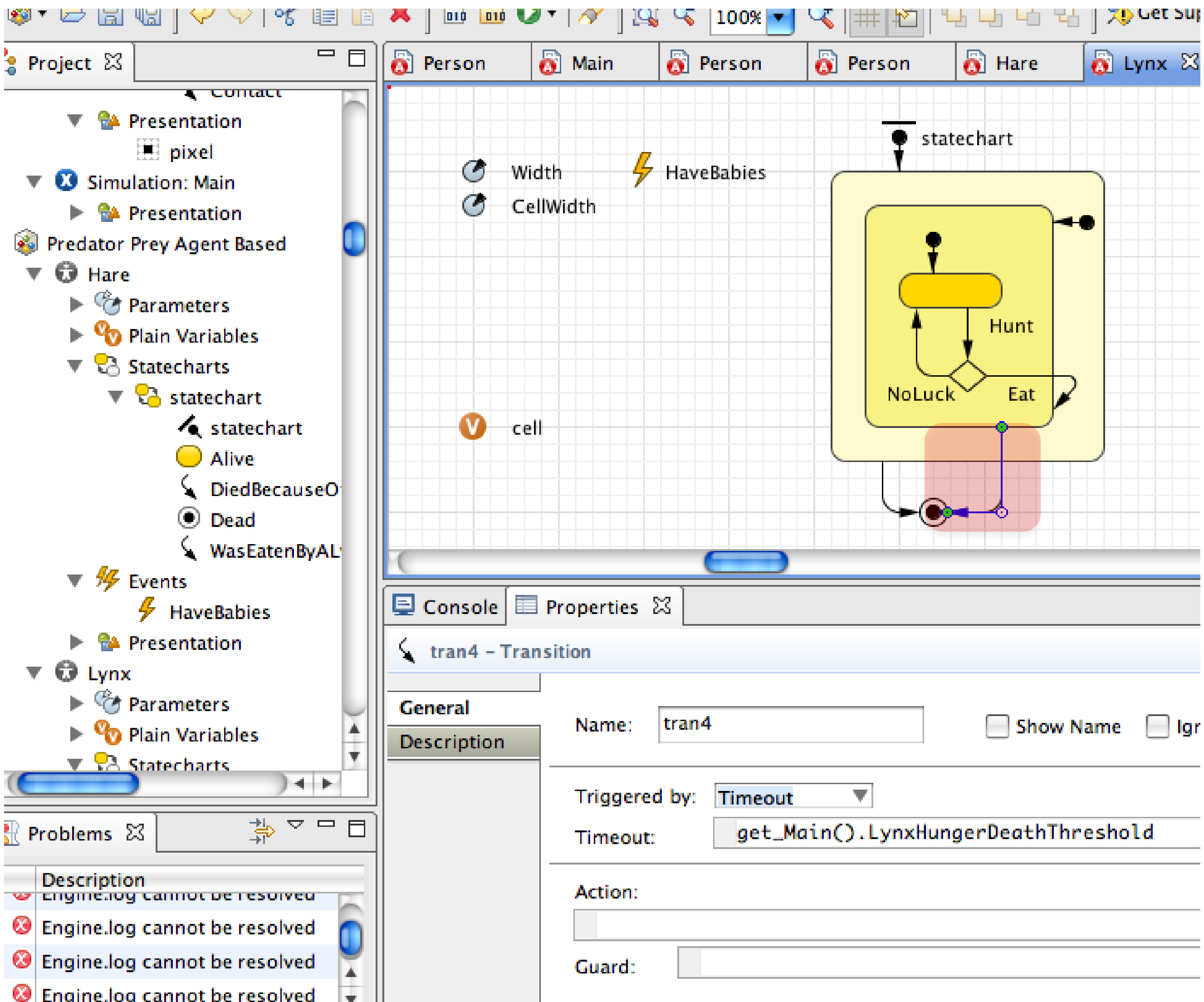


Natural Death Transition

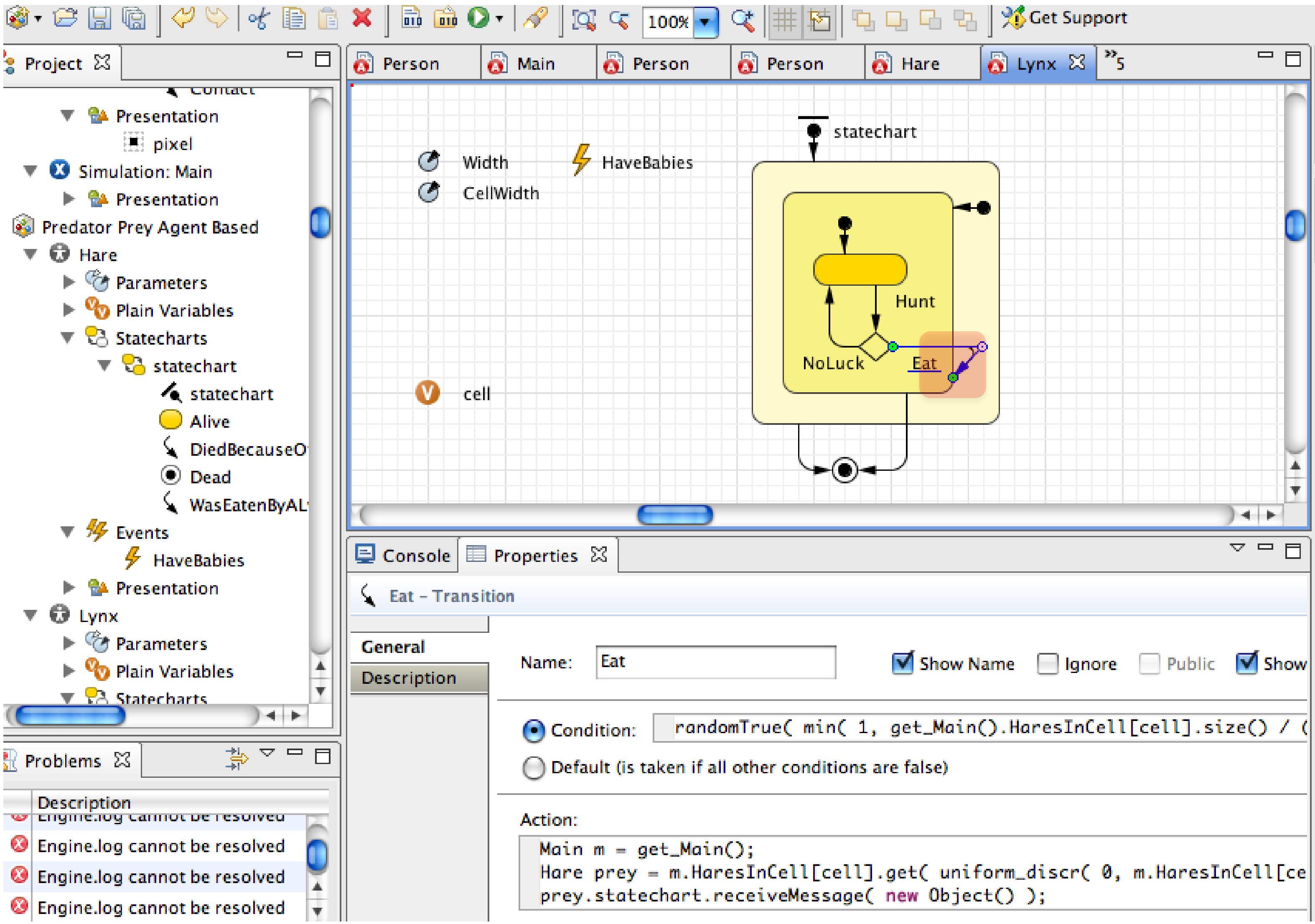


Death By Hunger

(Note that Depends on Time in State – i.e. time Since last ate)



Eating Transition Leaves & Reenters Middle State



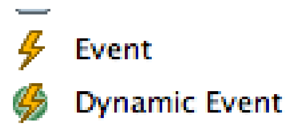
Tips on Statechart Code

- Each State & Transition has an integer index
  - This by accessed via a (static) constant holding the name of state within the statechart class (*statechart*.*StateName*)
- To determine length of time spent in state
  - *Statename*.getLocalTime(*StateIndex*)
- To determine current state
  - *statechart*.getActiveSimpleState()
- To find out if a state (either simple or composite) is currently active
  - *statechart*.isStateActive(*StateIndex*)

EVENTS IN ANYLOGIC

Rates & Events

- *Rates* and *Timeouts* are associated with types of events in AnyLogic
- Events can also be declared explicitly from the pallette



  - Dynamic events can have multiple instances
    - Each instance can be scheduled at the same time
    - The instances disappear after event firing
  - Regular (static) events can be rescheduled, enabled/disabled, but can only have one scheduled firing at a time
- There are some subtleties with events

Event Times: Options for Event Scheduling

- Manually (via restart() – see following slides)
- When boolean condition changes (depends on *onChange* being called)
- One-time
  - Can go off at a particular time (specified as a calendar time or as a double-precision value)
- At some initial time and then cyclically beyond with set "timeout" period
  - The timeout period is set according to the time unit
  - This goes off after *exactly* the timeout time
- At a specified Poisson Rate
  - Interarrival time is exponentially distributed!
  - Mean time between events is reciprocal of rate (i.e. 1/rate)

Event Subtleties

- Be very careful of what you count on for recomputation of rate – may think was recomputed, but hasn’t been
- Event rates (and likely event timeout times) are only computed occassionally, not continuously
  - These are computed when
    - Explicitly call event methods
      - start()
      - restart()
      - onChange()
    - When event fires and requires restarting
    - (For outgoing transitions) when enter a state in a statechart

- Calling “reset” will disable a rate until re-enable (e.g. with call to *restart()*)