Recall: Agent-Based Models

- **One or more populations composed of individual agents, each associated with**
  - **Parameters – discrete (e.g., Gender, Ethnicity) or continuous (e.g., birthweight, income)**
  - **State (continuous or discrete) e.g., age, smoking status, networks, preferences**
  - **Rules for evolving state**
  - **Means of interaction with other agents via one or more environments (e.g. spatial & topological context)**
- **Time horizon & characteristics**
- **Initial state**

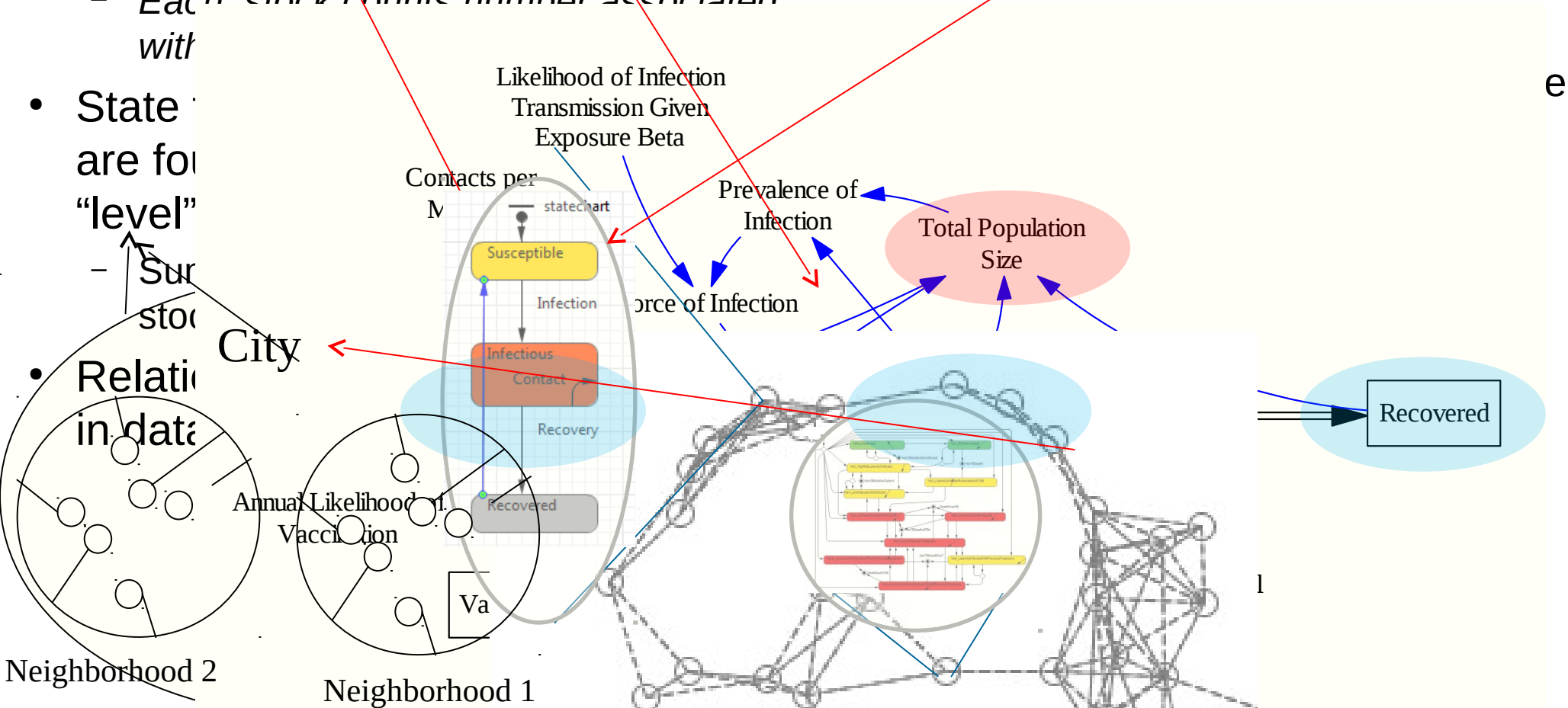
Contrasting Organization in Aggregate Stock-Flow & ABM

Aggregate Stock & flow models *Agent-based modeling*

- Within unit (e.g. city)
  - Subdivided according to *state (eg # susceptible, # infective)*
  - *Each stock counts number associated with*
- State
  are fo
  "level"
  - Su
    sto
- Relati
  in data

- Within unit (e.g. city)
  - Subdivided according to constitutive smaller units (e.g.

e



Contrasting Organization in Aggregate Stock-Flow & ABM

Aggregate Stock & flow models

- **Within unit (e.g. city)**
  - **Subdivided according to *state and characteristics (e.g. SES)***
  - ***Each stock counts # people in associated population group***
- **State for different levels and other actors are found in stocks & flows at same "level" of the model**
  - **Summaries for entire pop. & subpops are stocks in model**
- **Relationships between units implicit in data (e.g. mixing matrix)**

Agent-based modeling

- **Within unit (e.g. city)**
  - **Subdivided according to constitutive smaller actors (e.g., individual people)**
  - **Each unit maintains its own state, attributes**
- **The nested or networked relations among actors mirror that in world**
  - **If a city "contains" people, the (references to) people appear "inside" the city**

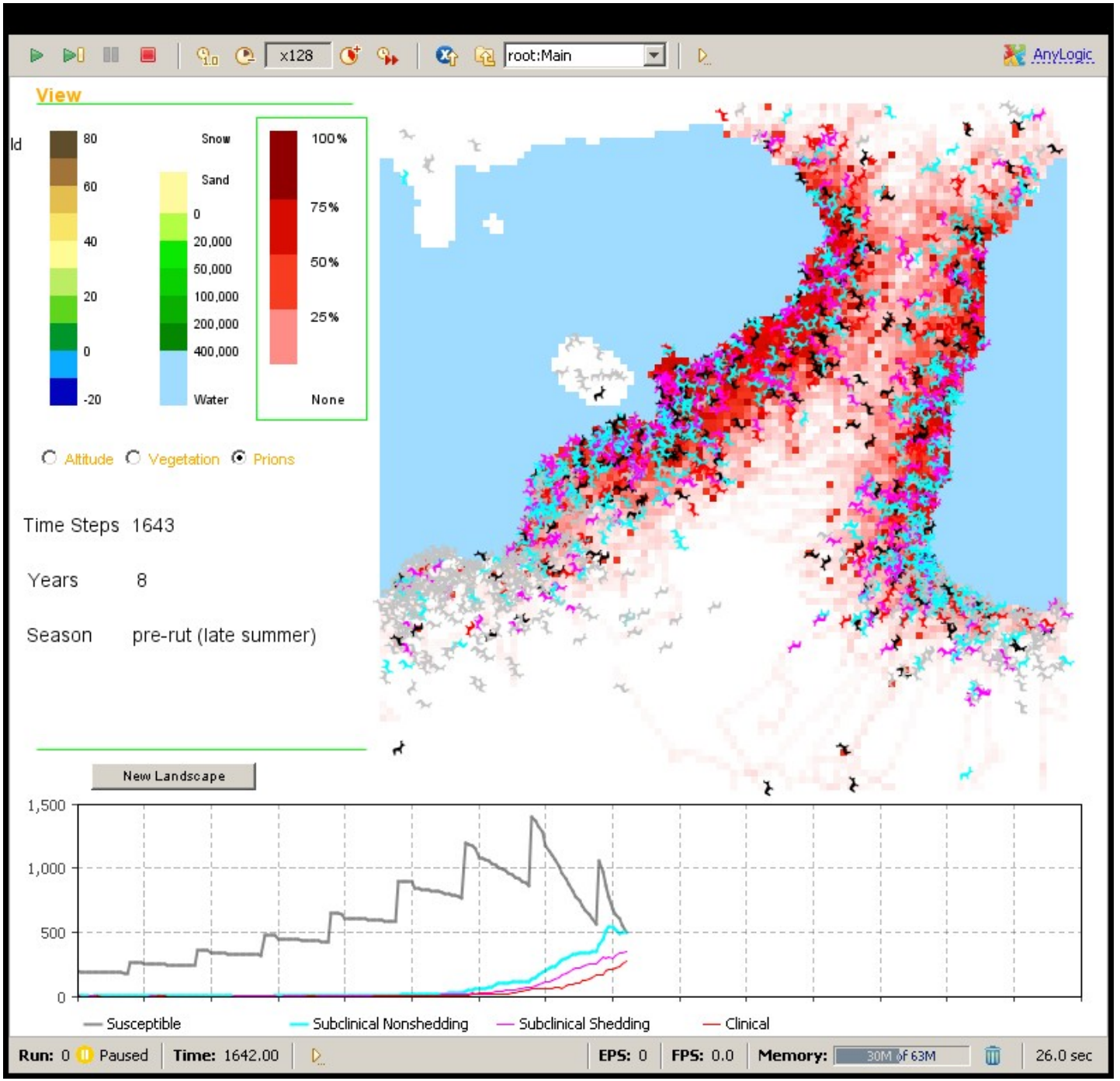
Emergent Behavior

- **“Whole is greater than the sum of the parts”, “Surprise behavior”**
- **Frequently observed in stock and flow models as interaction between stocks & flows**
- **In ABMs, we see this phenomena not only at level of aggregate stocks & flows, but – most notably – *between agents***
  - ***Patterns over time***
  - ***Patterns over space***
  - **Patterns over networks**

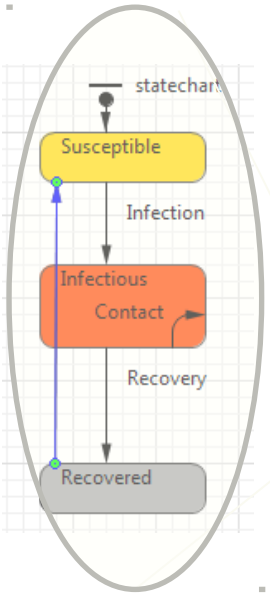
Emergent Behaviour & Modeling Types

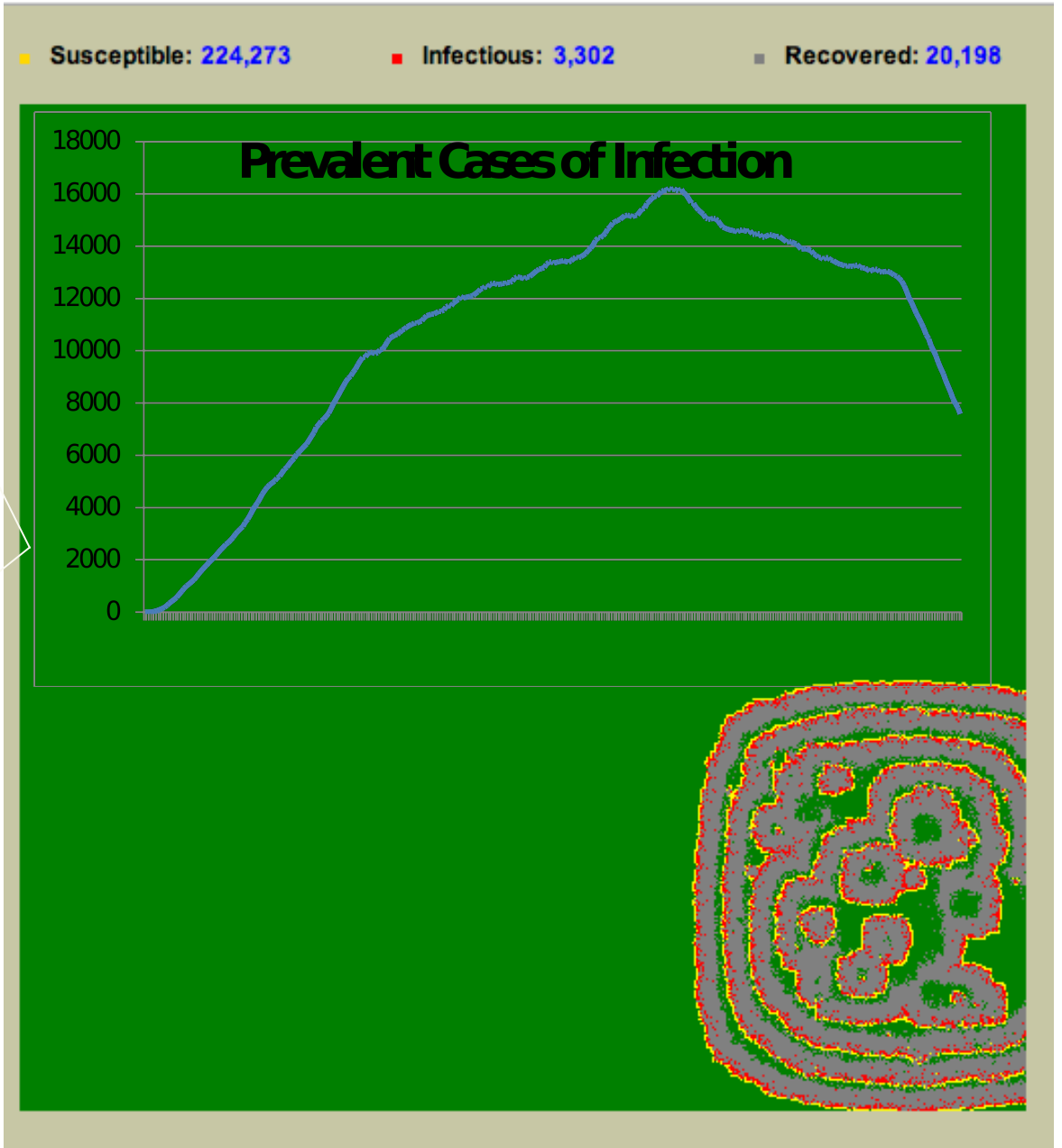
- Agent-based modeling particularly emphasizes multi-level emergence – how distinctive patterns can emerge at different levels of the system
  - Ability to look at high-level emergence reflects the presence of many individual agents within one model
- The emergent behaviours can change significantly with changes in model structure&parameter values

Aggregate & Spatial Emergence



Emergent Aggregate & Spatial Dynamics





Early Origins of Modern ABM

- Modern Agent-Based Modeling reflects two
  - Origins
  - Theoretical bases
- **Computer Science/Applied Mathematics/Physics**: Von Neumann's and Ulam's theory of automata (1940s)
  - Interacting finite state automata
  - Cellular automata
  - Reproduction
  - Theoretical&practical foundation in FSA/FSM formulaion
- **Economics**: Microsimulation
  - Statistical formulation of transitions
  - Sometimes framed as challenge to neoclassical economics and rational actor theory
  - Often less central focus on direct agent interactions
- These contributions are each associated with distinct underlying theories, traditions

Additional Comments on ABM History

- Other notable influences:
  - Axelrod's work on interacting prison's dilemma
  - Los Alamos (CS/Physics)
  - Schelling segregation model (1971)
  - Conway's game of Life
  - Computational Mathematical & Organization Theory (OR/MS, SS, Mathematics)
  - Lucas' Microfoundations
  - Object-oriented programming

Agent-Based Models & Theory

- While ABM has roots in domains with deep theory (e.g., finite state automata, object orientation), it has hewed pluralism and openness to evolving and diverse formalisms
- Reflecting ABM's emphasis on decentralization&evolution, it embraces community contributions of software & theory
  - Open-source software has been fundamental to the evolution of the field
- Particular ABMs may be associated with well-defined theoretical underpinnings

Comments on Building Up an Agent-Based Model

Model Specification

Stock & Flow Models

- Small modeling vocabulary
- Power lies in combination of a few elements (stocks & flows)
- Analysis conducted predominantly in terms of elements of model vocabulary (values of stocks & flows)
- Directly maps onto crisp mathematical description (Ordinary Differential Equations)

Agent-Based Modeling

- Large modeling vocabulary
- Different subsets of vocabulary used for different models
- Power in flexibility & combination of elements & algorithmic specification
- Variety in analysis focus
- Mathematical underpinnings differ
- In most cases, lacks transparent mapping to mathematical formulation

ABMs: Larger Model Vocabulary & Needs

- Events
- Multiple mechanisms for describing dynamics
  - State transition diagrams
    - Multiple types of transitions
  - Stock and flow
  - Custom update code
- Inter-Agent communication (sending & receiving)
- Diverse types of agents
- Data output mechanisms
- Statistics
- Subtyping
- Mobility & movement
- Graphical interfaces
- Stochastics complicated
  - Scenario result interpretation
  - Calibration
  - Sensitivity analysis
- Synchronous & asynchronous distinction, concurrency
- Spatial & topological connectivity & patterning

Agent-Based Models: Skill Sets

- **Construction of ABMs have traditionally required significant software engineering**
- **In recent years, ABM platforms have included increasing support for declarative specification**
  - **Such features greatly lower the programming requirements**
  - **Maintaining on-call computational consults remains important**

Agent-Based Systems

- Agent-based model characteristics
  - One or more populations composed of individual agents
    - Each agent is associated with some of the following
      - State (continuous or discrete e.g. age, health, smoking status, networks, beliefs)
      - Parameters (e.g., Gender, genetic composition, preference fn.)
      - Rules for interaction (traditionally specified in general purpose programming language)
    - Embedded in an environment (typically with localized perception)
    - Communicate via messaging and/or flows
  - Environment
- Emergent aggregate behavior

Agent-Based Models

- **One or more populations composed of individual agents, each associated with**
  - **Parameters – discrete (e.g., Gender, Ethnicity) or continuous (e.g., birthweight, income)**
  - **State (continuous or discrete) e.g., age, smoking status, networks, preferences**
  - **Rules for evolving state**
  - **Means of interaction with other agents via one or more environments (e.g. spatial & topological context)**
- **Time horizon & characteristics**
- **Initial state**

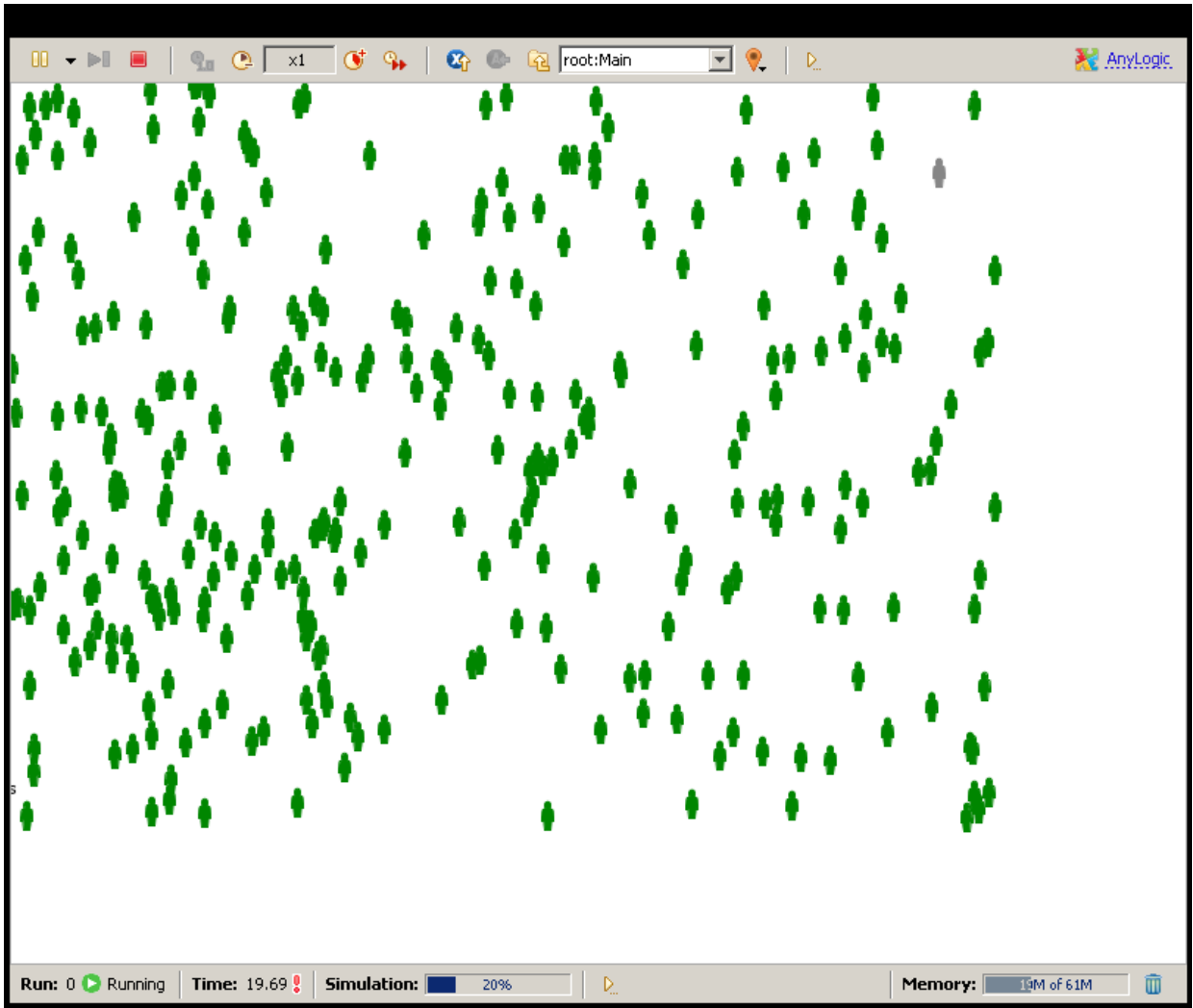
Agent-Based Models: Skill Sets

- **Construction of ABMs have traditionally required significant software engineering**
- **In recent years, ABM platforms have included increasing support for declarative specification**
  - **Such features greatly lower the programming requirements**
  - **Maintaining on-call computational consults remains important**

Model Population

In Model….

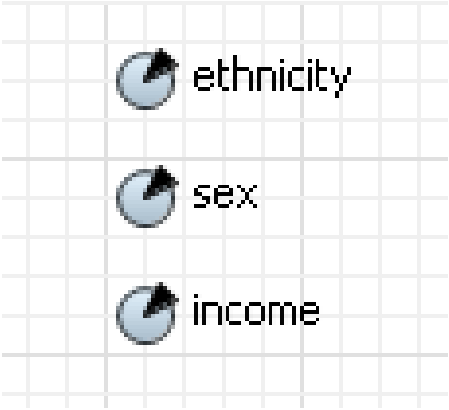
In Simulation….



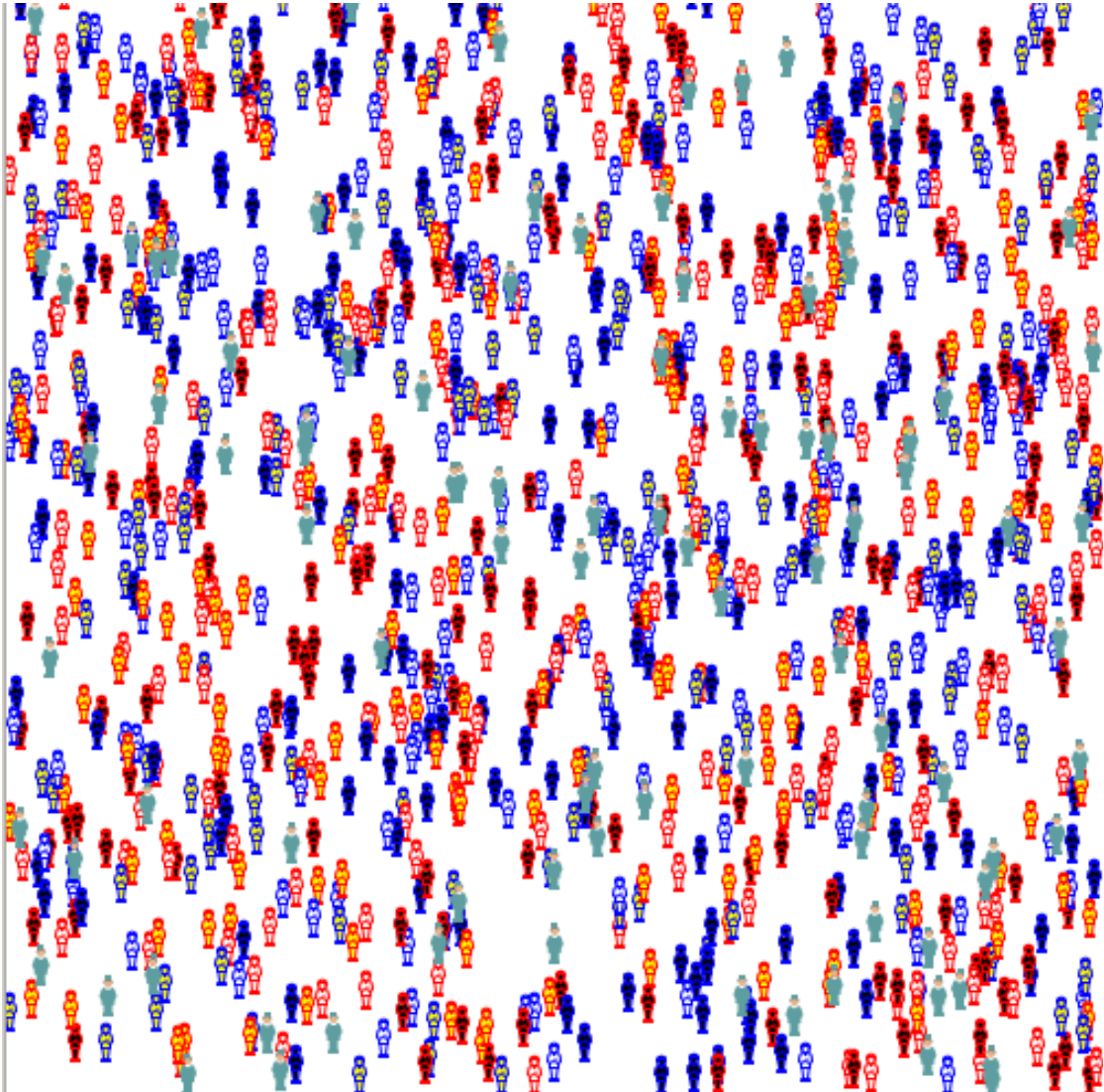
Agent-Based Models

- **One or more populations composed of individual agents, each associated with**
  - **Parameters – discrete (e.g., Gender, Ethnicity) or continuous (e.g., birthweight, income)**
  - **State (continuous or discrete) e.g., age, smoking status, networks, preferences**
  - **Rules for evolving state**
  - **Means of interaction with other agents via one or more environments (e.g. spatial & topological context)**
- **Time horizon & characteristics**
- **Initial state**

Model



In Simulation



Properties of Individual Agents

ethnicity
Caucasian

sex
Male

income
19,389.152

ethnicity
African_American

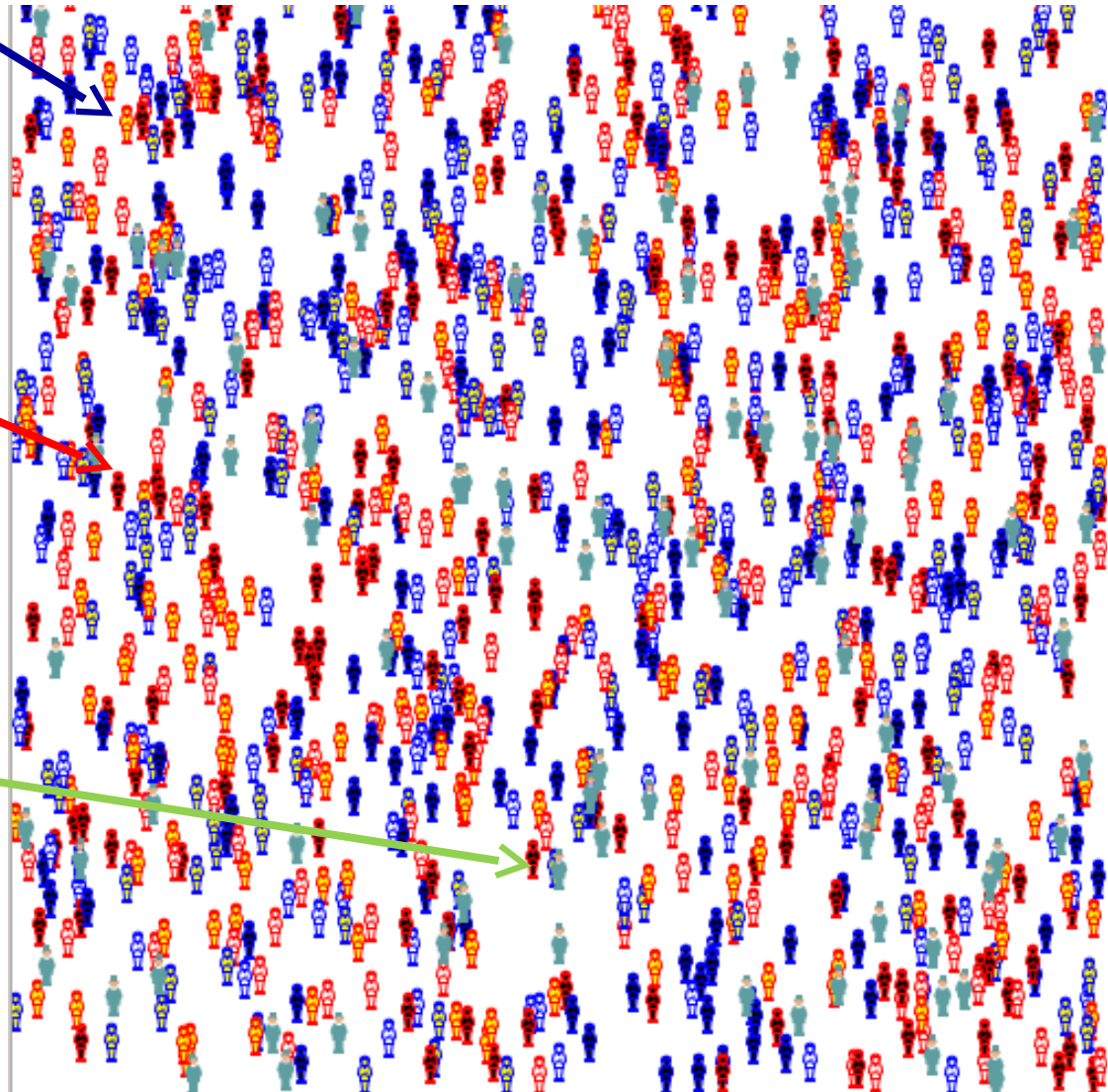
sex
Female

income
18,439.904

ethnicity
African_American

sex
Male

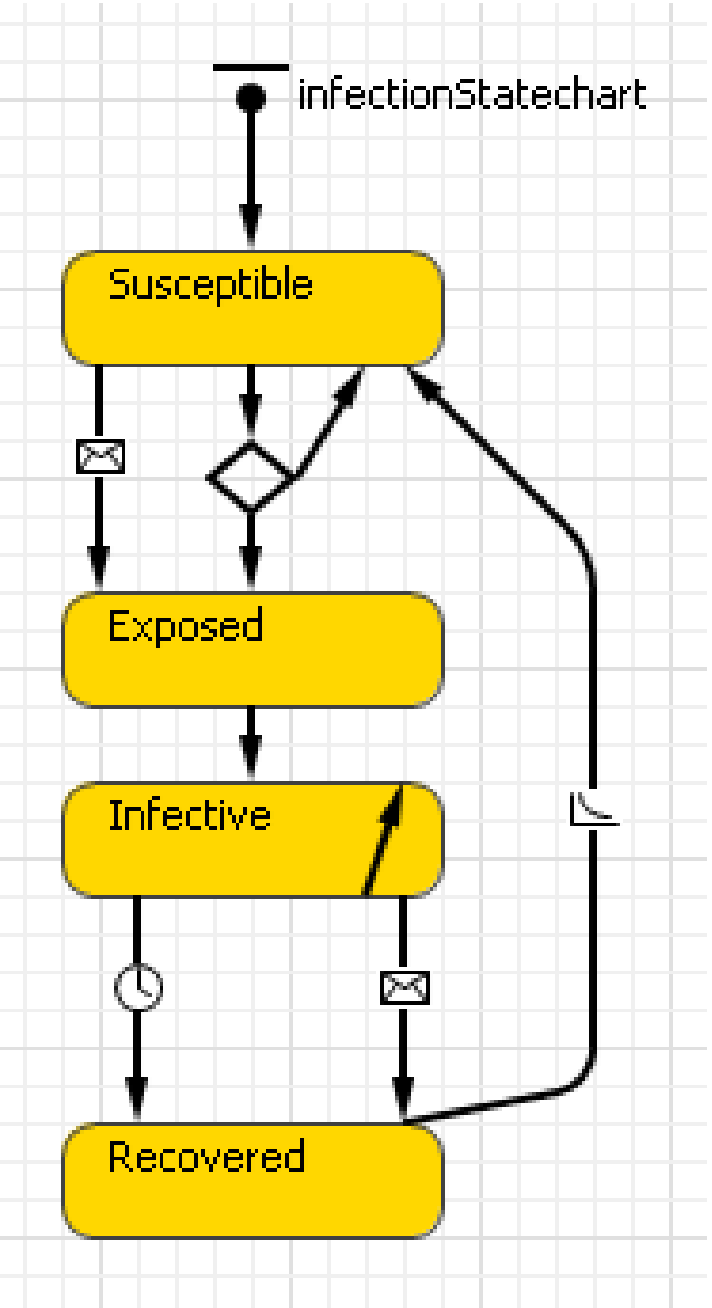
income
63,959.083



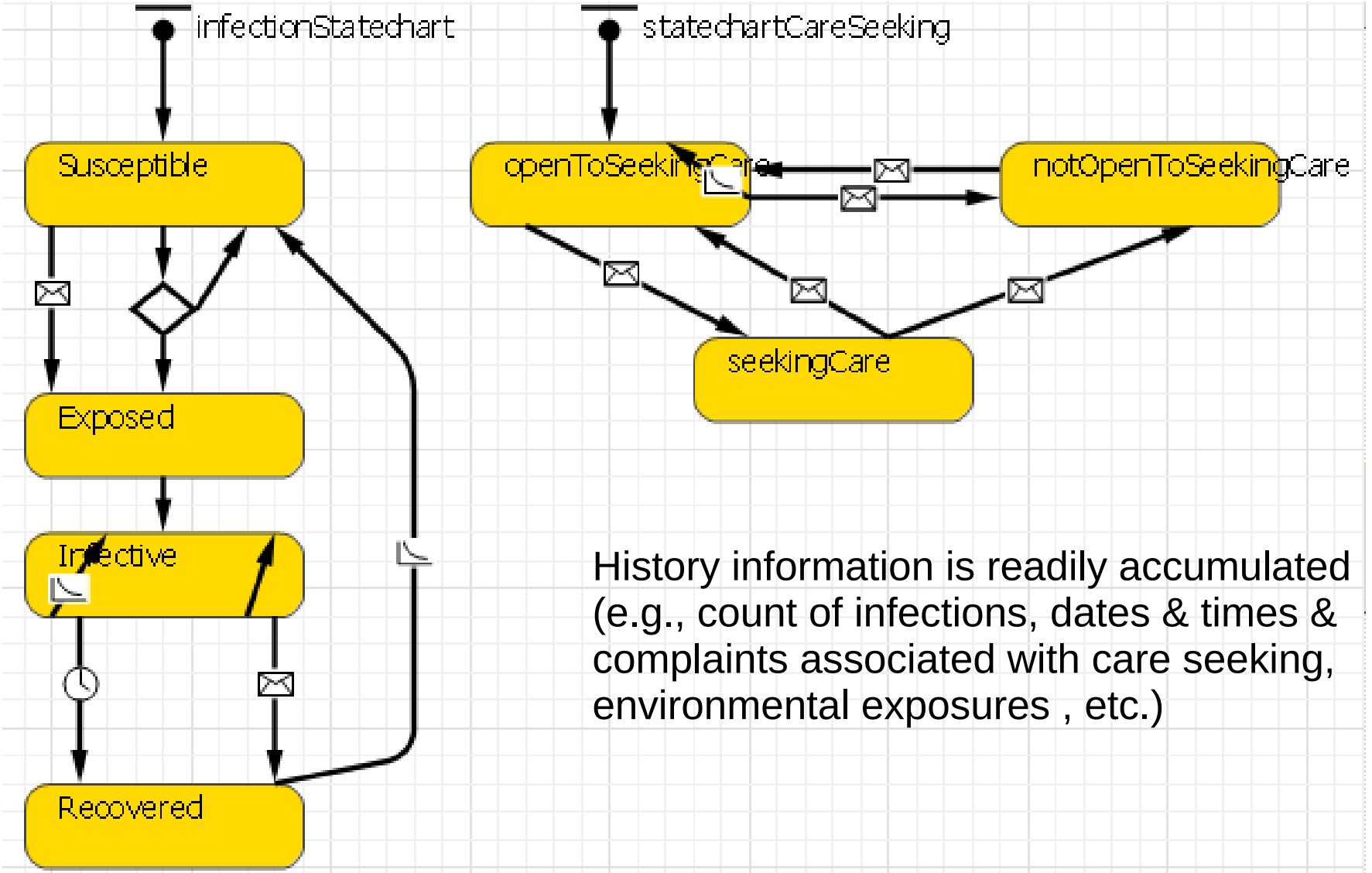
Agent-Based Models

- **One or more populations composed of individual agents, each associated with**
  - **Parameters – discrete (e.g., Gender, Ethnicity) or continuous (e.g., birthweight, income)**
  - **State (continuous or discrete) e.g., age, smoking status, networks, preferences**
  - **Rules for evolving state**
  - **Means of interaction with other agents via one or more environments (e.g. spatial & topological context)**
- **Time horizon & characteristics**
- **Initial state**

Example of Discrete States & Associated Transitions



Contrast to Agg. Stock & Flow Models: Adding Heterogeneity Yields No Combinatorial Explosion in Structure

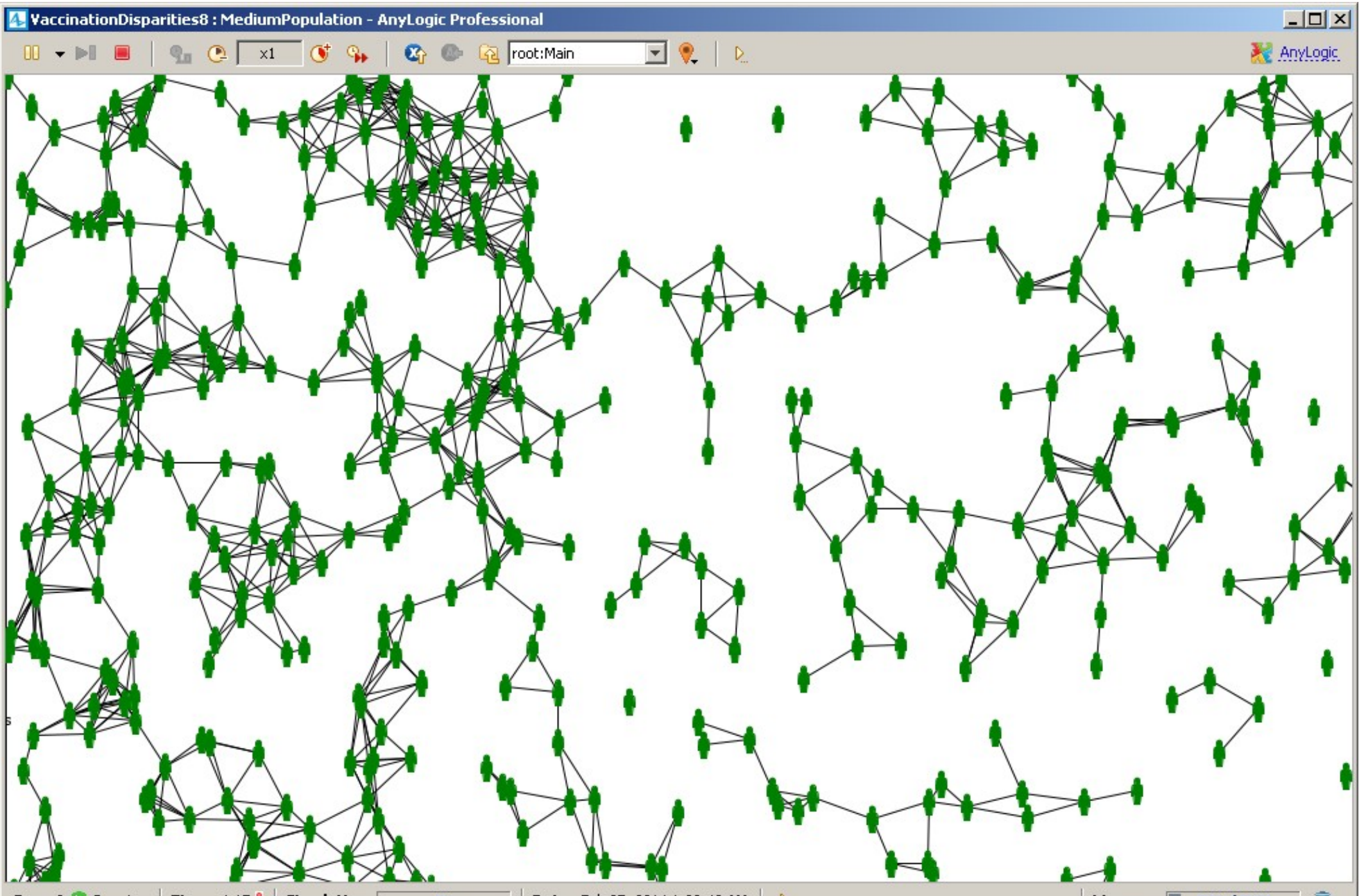


History information is readily accumulated (e.g., count of infections, dates & times & complaints associated with care seeking, environmental exposures , etc.)

Agent-Based Models

- **One or more populations composed of individual agents, each associated with**
  - **Parameters – discrete (e.g., Gender, Ethnicity) or continuous (e.g., birthweight, income)**
  - **State (continuous or discrete) e.g., age, smoking status, networks, preferences**
  - **Rules for evolving state**
  - **Means of interaction with other agents via one or more environments (e.g. spatial & topological context)**
- **Time horizon & characteristics**
- **Initial state**

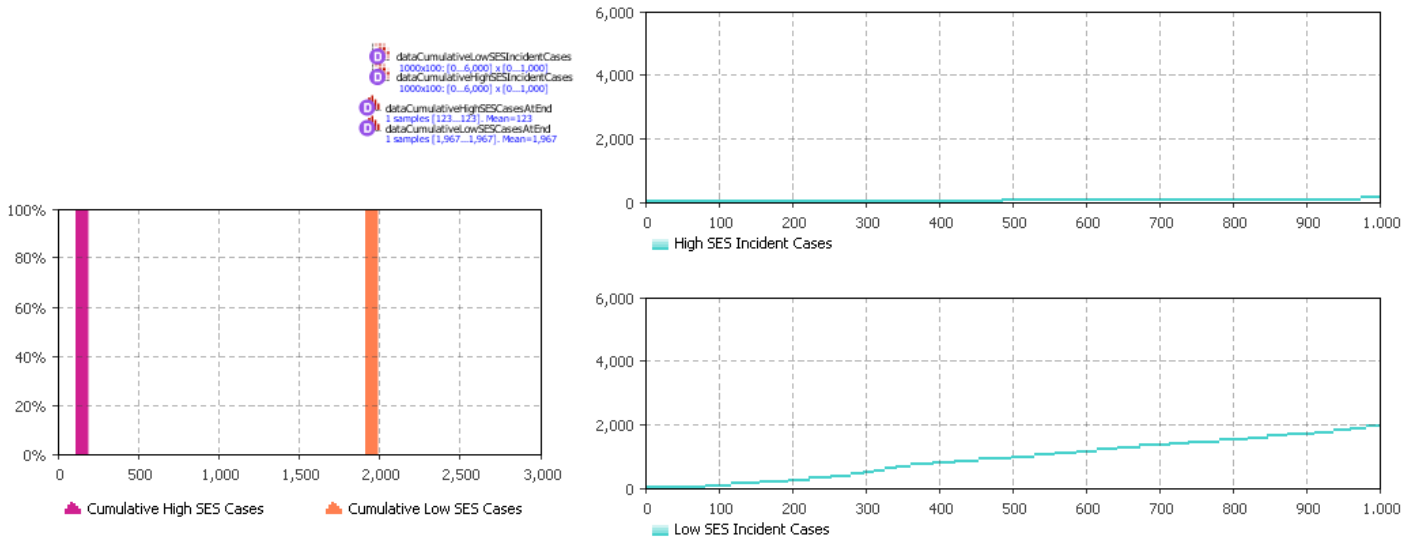
Adding Contact Network



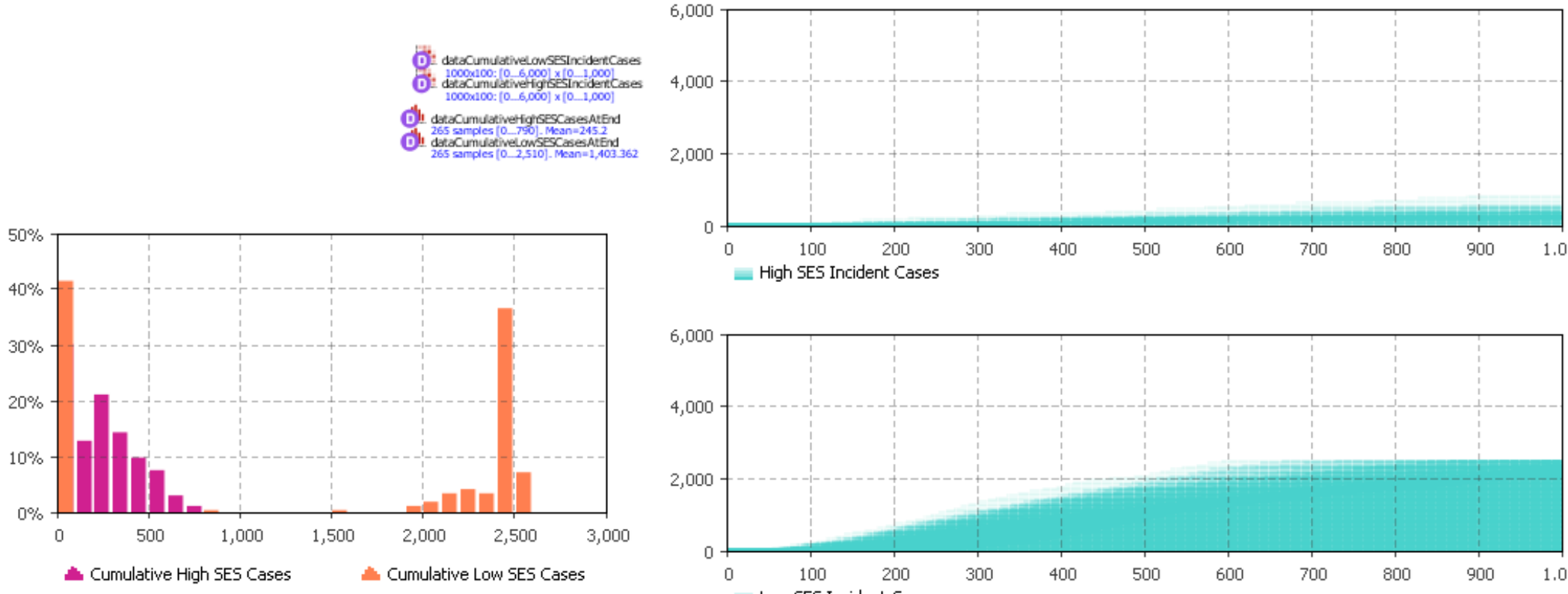
Stochastics

- **In contrast to most system dynamics models, ABMs are typically stochastic**
- **To ensure model results are not merely flukes, a model must be run many times**
  - **This adds substantially to the cost associated with such models**
  - **This is easily parallelizable**
- **Stochastics as assets: Observing variability can lend insights into the variability seen in real-world data**

Single Run



Monte Carlo Ensemble Output



Incremental Model Development

- Great advantages are conferred by building a simulation model in a step-by-step fashion
- With each iteration, the model is modified in some small fashion
- A new version of the model is “docked” against older versions of the model
  - Confirming identical behavior when the changes are disabled
  - Understanding behavior with the new feature enabled
- Frequently these incremental versions
  - Can be demonstrated to system stakeholders
  - Produce insight that inform the next step undertaken

Benefits of Incremental Development

- Greater understanding of where model patterns emerge & interactions
- Much faster defect identification & diagnosis
- Flexibility to change direction based on learning
- Capacity to secure feedback from stakeholders (e.g. observations of unexpected emergent model patterns, prioritization of issues)
- Greater clarity in prioritization
- More effective time-boxing
- Enhanced stakeholder confidence
- Improved morale

Hands-On Components

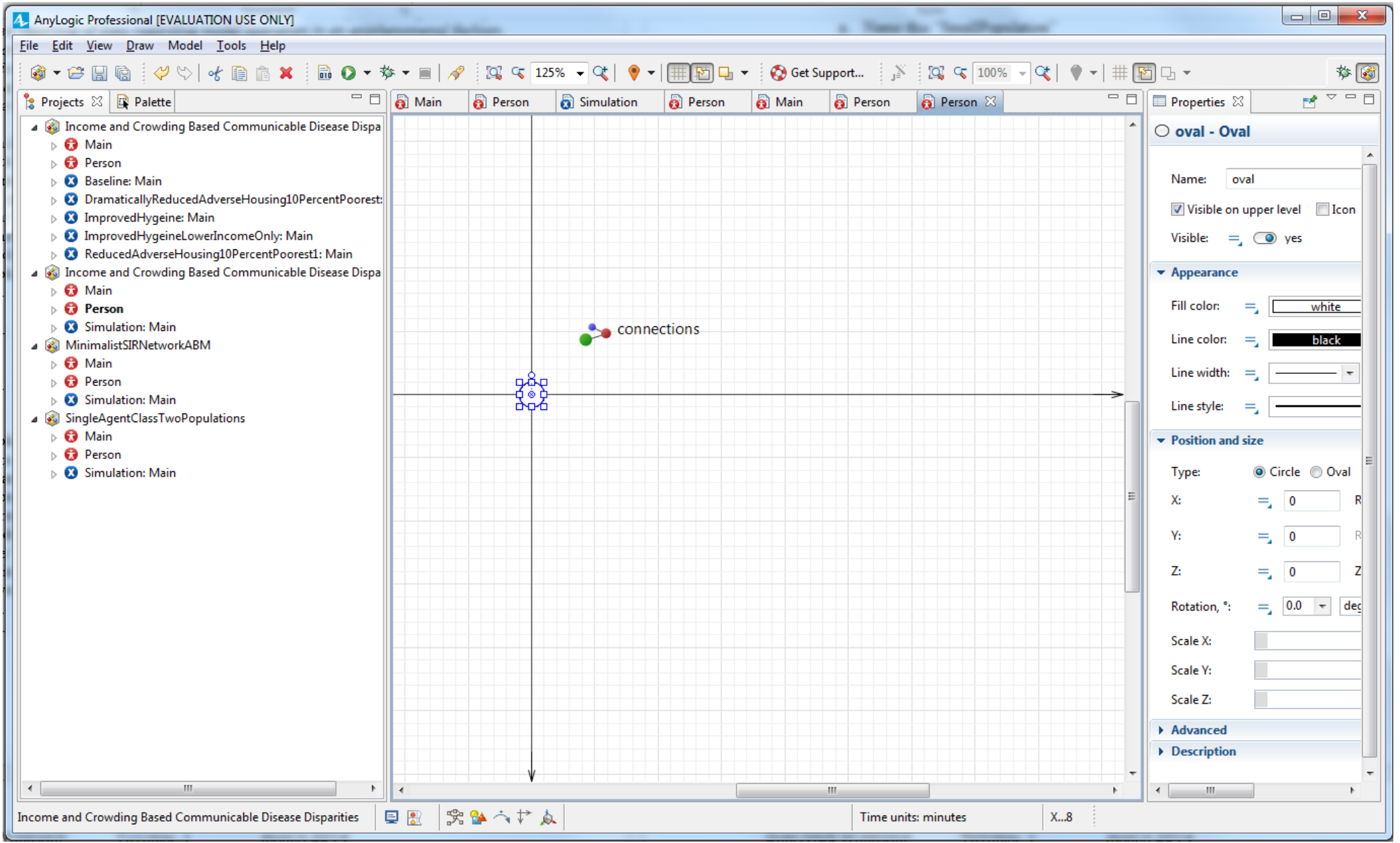
Notable Themes

- Heterogeneity – continuous & discrete
- Events
- Non-linearity
- ABM Structure
- Network & spatial context
- Agent parameters and state
- Specification of agent evolution
- Agent interactions
- Maintaining longitudinal (“biographical”) information
- Scenarios for intervention
- Reporting processes & granularity
- Emerging declarative ABM specification mechanisms
- Stochastics

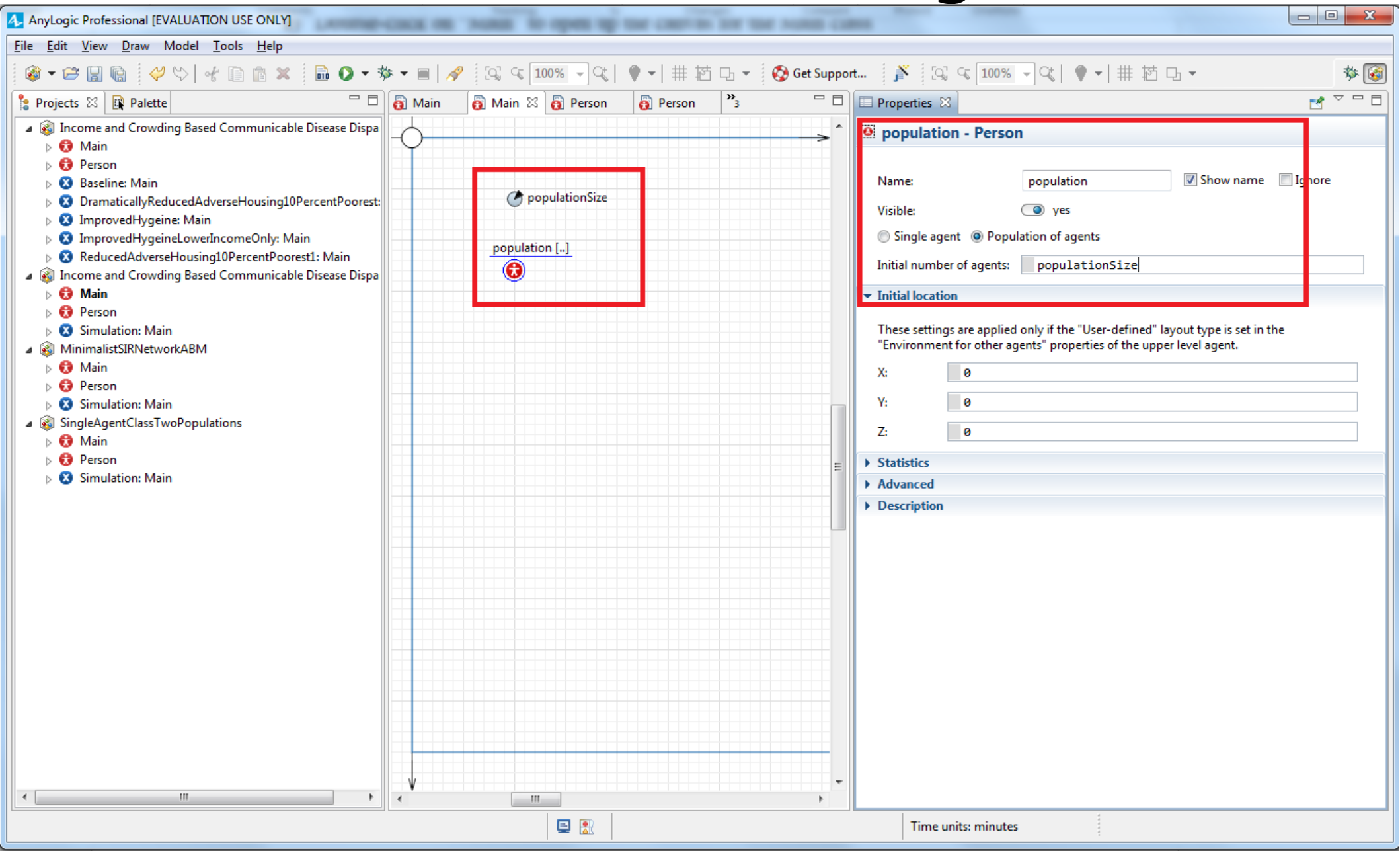
Avoiding a Common Mistake

- AnyLogic projects typically contain a variety of “classes”
- The AnyLogic interface for accessing these classes is deceptively similar
- The semantics of the model will typically be very different depending on whether you add a component to one class or another
- Reflect on be very clear as to which class you wish to add an element

Creating an Agent



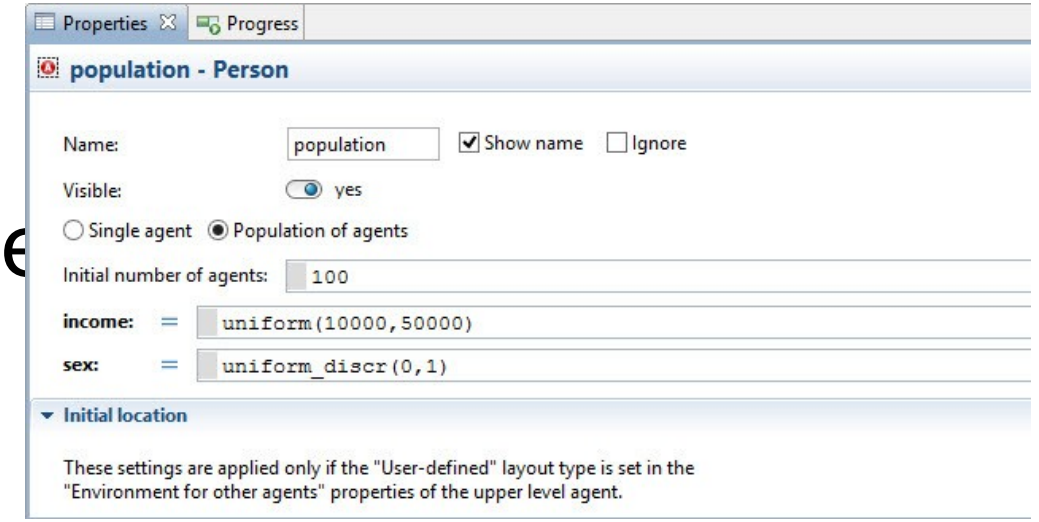
Adding a Parameter for pop Size & Enforcing it



Parameters: Static Quantities

- Parameters normally
  - Define constants that represent assumptions
  - Serve as mechanism to *communicate* such assumptions
- In Java, such parameters can have many types
  - Integer, Double precision value, boolean, etc.
- For parameters in the *Main* class, we can override the value of the parameters in an experiment
- Presentation elements associated with an Agent have special “Presentation” tab for their parameters

Model-Wide Parameters

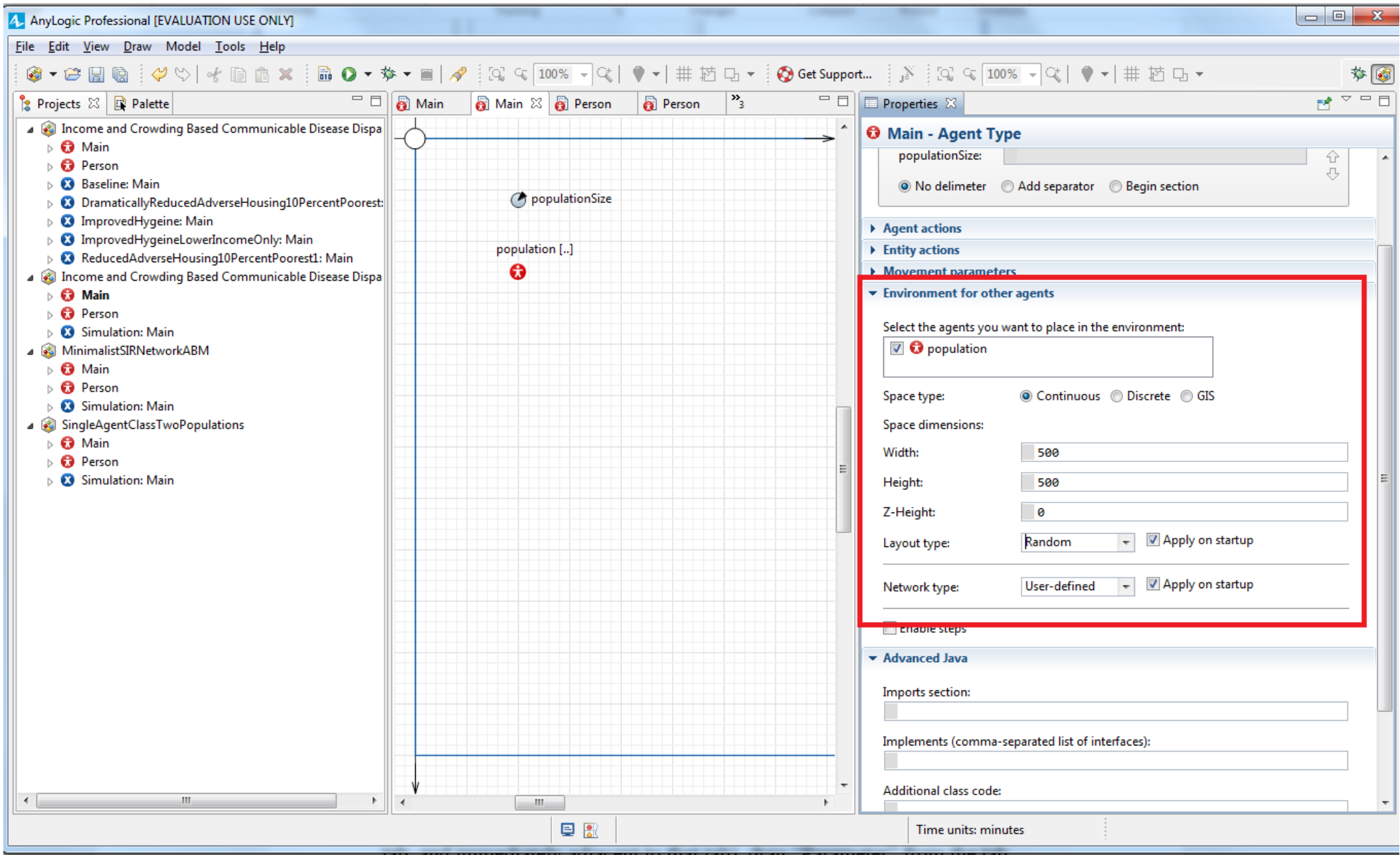


- Values for agent parameters are specifie by the associated Population
- We can also associate parameters with the “Main” class
  - These parameters can be model-wide quantities (e.g. the size of the population, or the duration of infectiousness to assume for all agents)
  - Values for these parameters are specified by *Experiments*

Parameters and Communication

- Beyond defining assumptions, parameters in AnyLogic serve as mechanism to *communicate* such assumptions
- This communication takes place from an enclosing object at the point of creation of an enclosed object
  - From an Experiment (scenario) to the single instance of the Main class (as it is being created)
  - From the single instance of the Main class to a particular agent (as it is being created)
  - From a collective agent (e.g. City, Farm) to a particular enclosed agent (Person, Horse) as that enclosed agent is being created

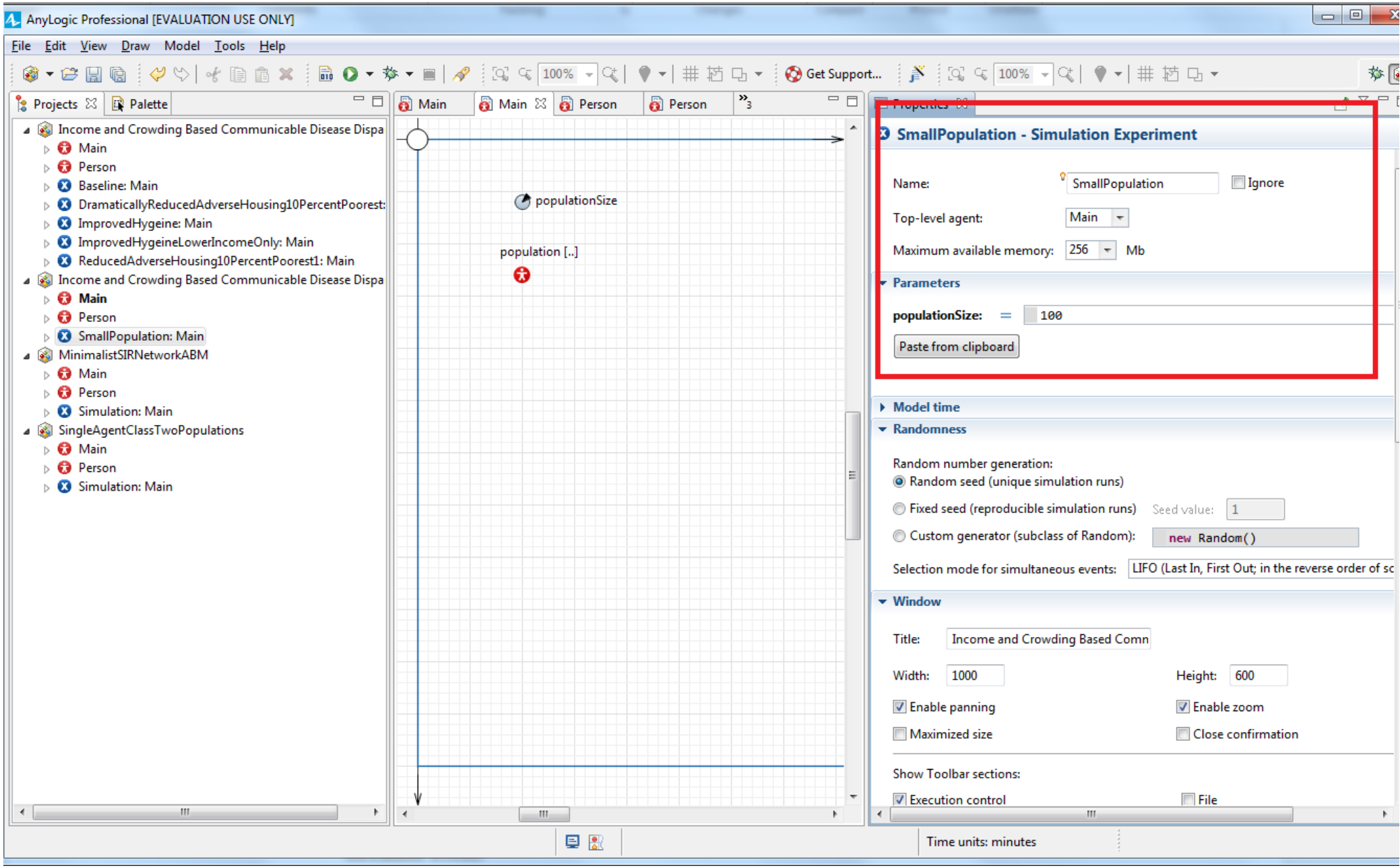
Specifying a (Temporary) Layout for Agents



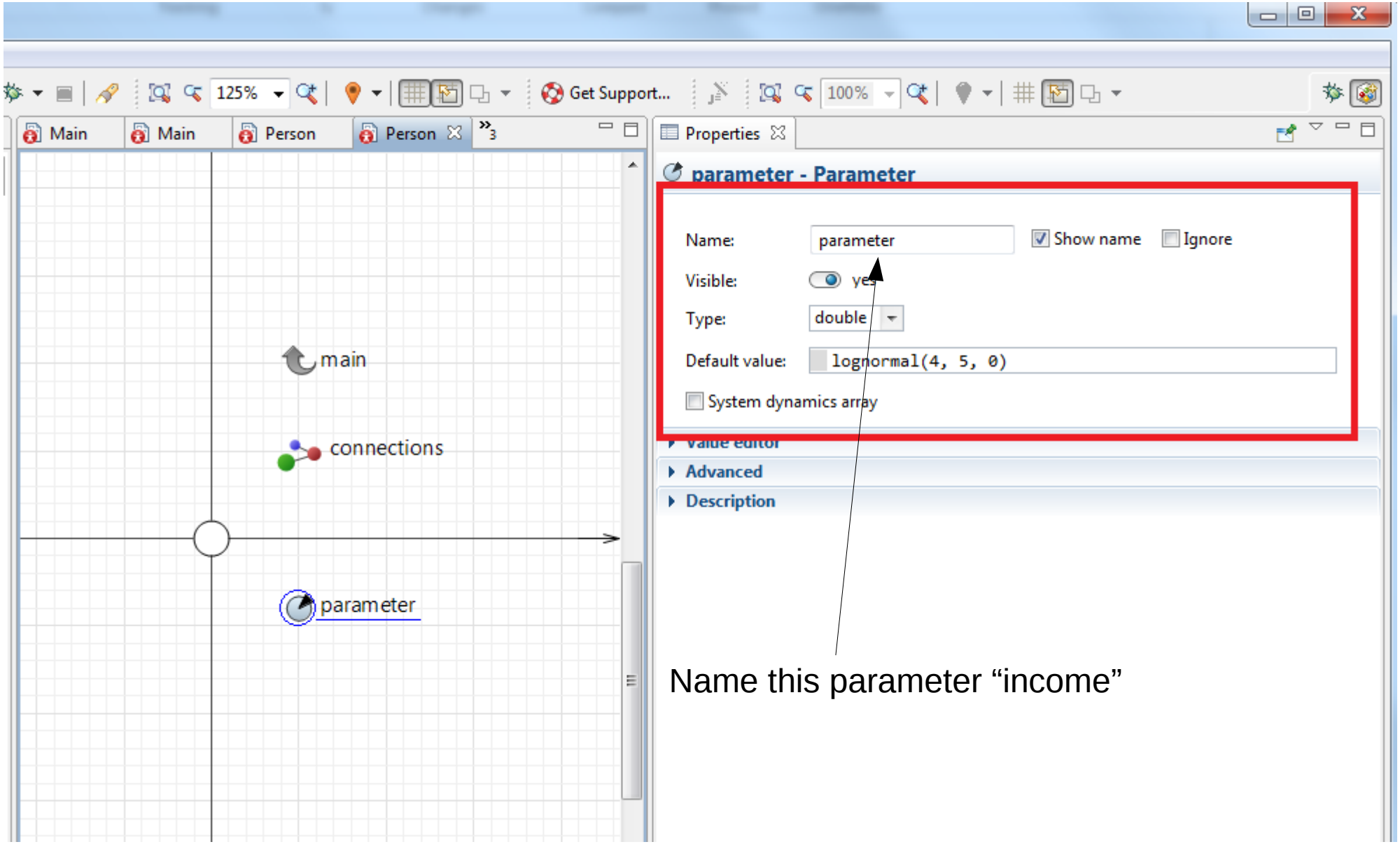
Layout Type

- **Random**: Uniformly distribute X and Y position of nodes
- **Arranged**: Set node locations in a regular fashion (normally in a 2D grid)
- **Ring**: Set node locations in periodically spaced intervals around a ring shape
- **Spring Mass**: Adjust node locations such that node locations that are most tightly connected tend to be closer together
  - (Sets location based on network!)
- **User-Defined** User can set location (e.g. in initialization code)

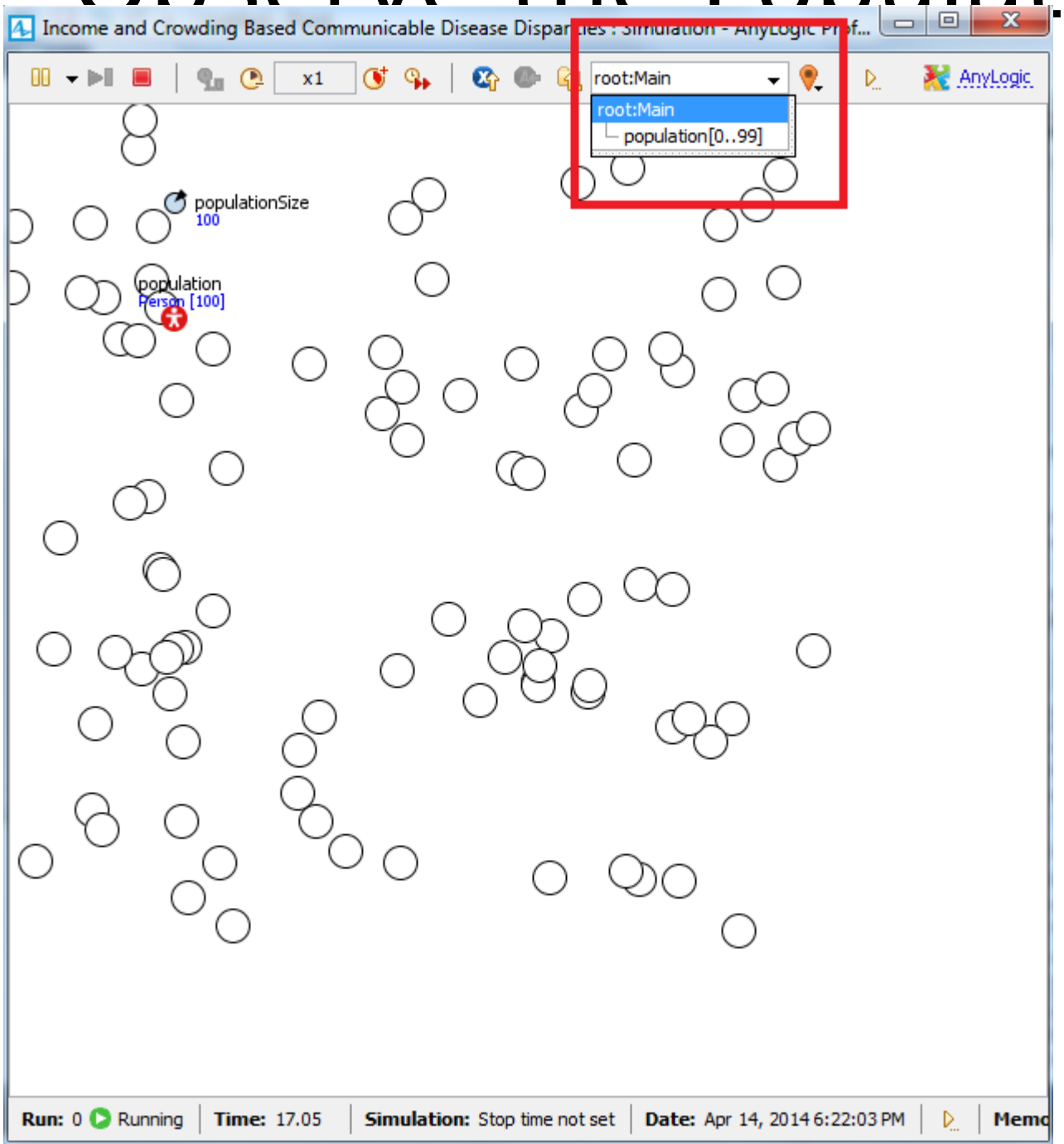
Adding an Experiment



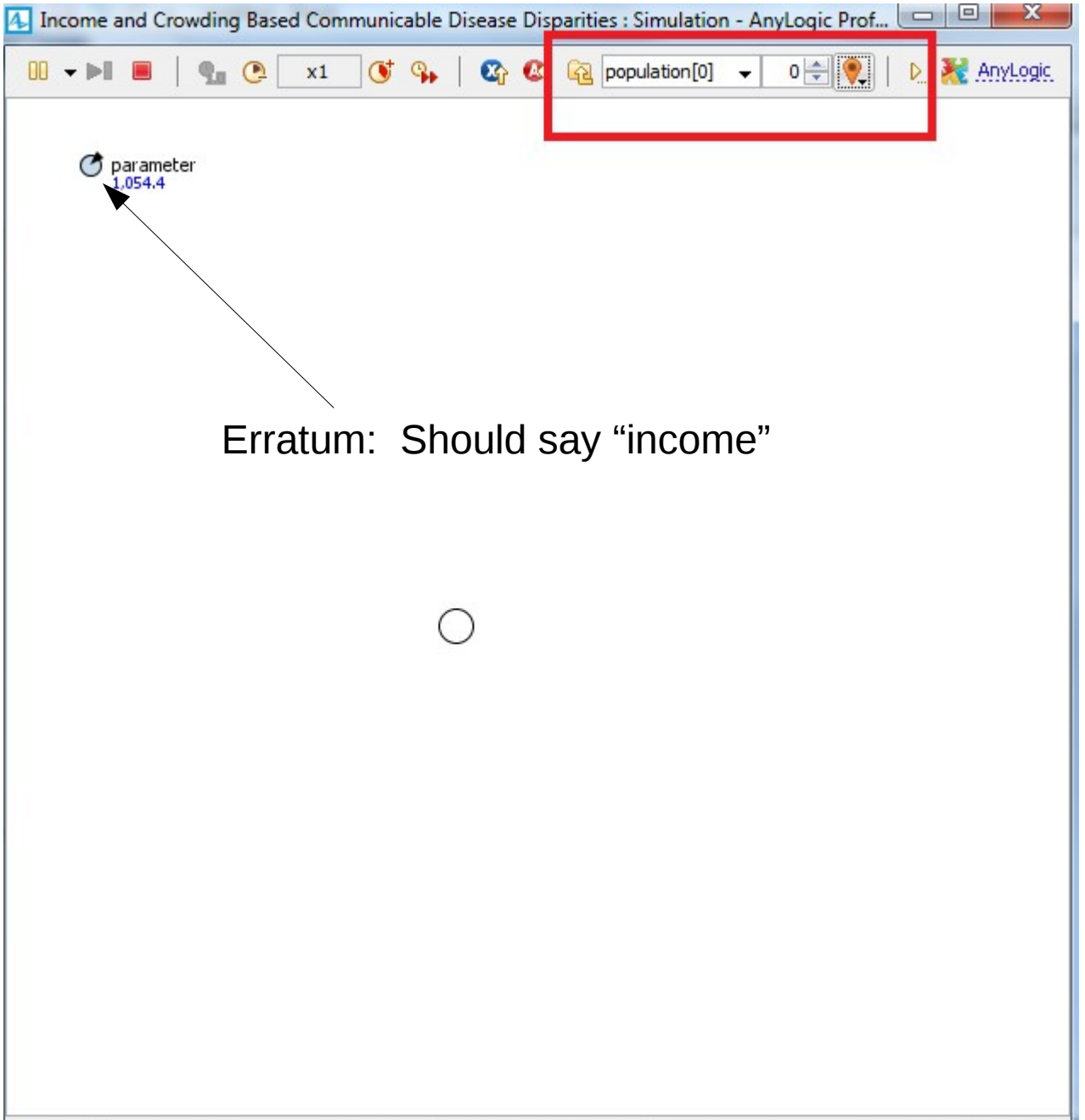
Adding a Parameter



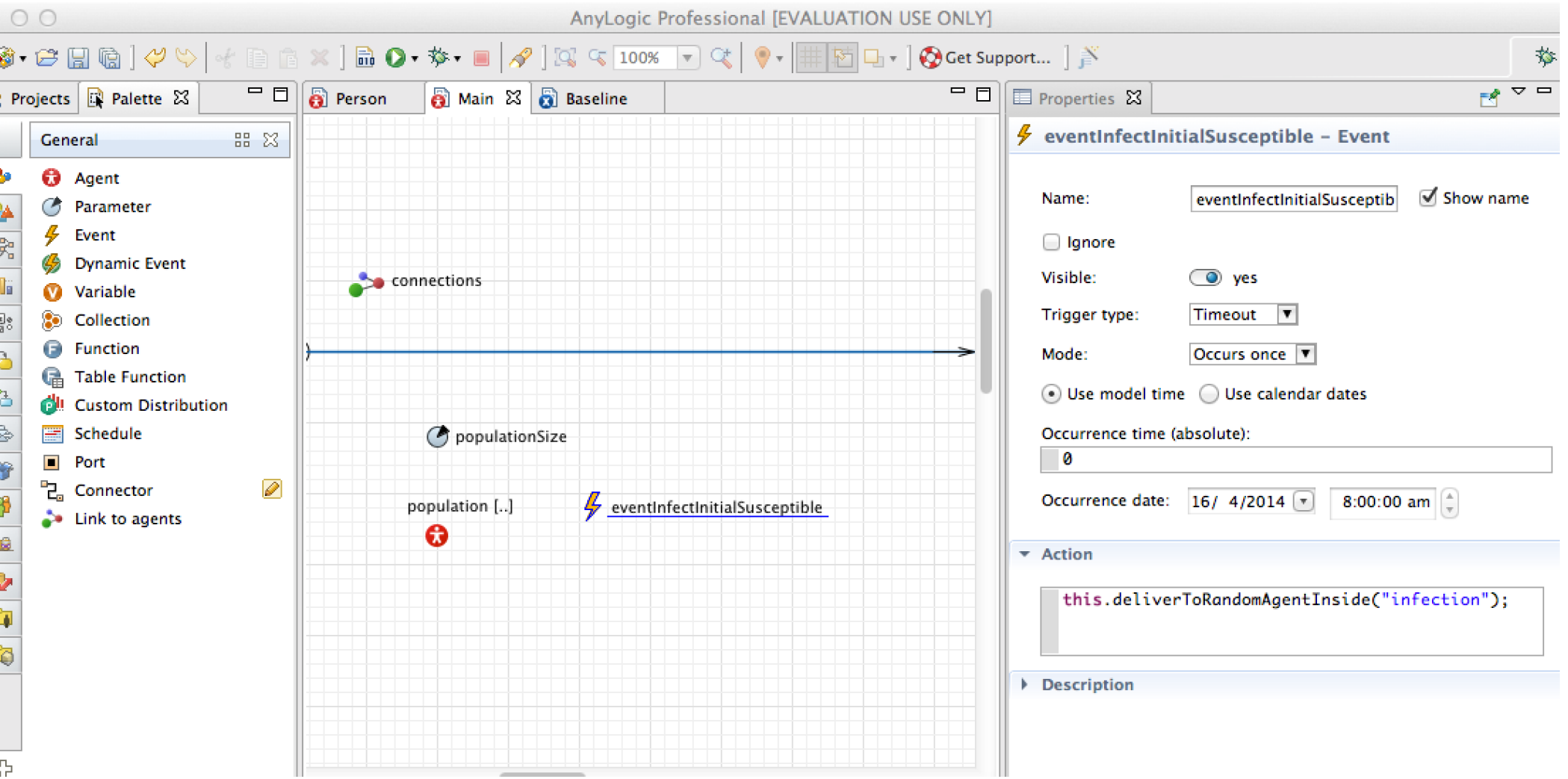
Run the Model & Observe the Population



Drill Down to Population Members



The Associated Code



Hands-On Components: Heterogeneity

Elements of Individual Characteristics

- Example Discrete
  - Ethnicity
  - Gender
  - Categorical infection status
- Continuous
  - Age
  - Elements of body composition
  - Metabolic rate
  - Past exposure to environmental factors
  - Glycemic Level

Importance of Heterogeneity

- Heterogeneity often significantly impacts policy effectiveness
  - Policies preferentially affect certain subgroups
    - Infection may be maintained within certain subgroups even though would tend to go extinct with random mixing in the entire population
  - Policies alter balance of heterogeneity in population
    - Shifts in the underlying heterogeneity can change aggregate population statistics
  - Given a non-linear relationship, inaccurate to use the mean as a proxy for whole distribution
- **Assessing policy effectiveness often requires representing heterogeneity**
- ***Flexibility* in representing heterogeneity is hard to achieve in aggregate (coarse-grained) models**

Impacts of Heterogeneity on Policy Effectiveness

- Value of breast cancer detection (Park & Lees)
- Impact of airbags on deaths (Shepherd&Zeckhauser)
- Value of hernia operations (Neuhauser)
- Impact of cardiovascular disease interventions (Chiang)
- Controlling blood pressure (Shepherd&Zeckhauser)
- Effectiveness of mobile cardiac care unit (Shepherd&Zeckhauser)
- Value of breast cancer treatment (Fox)
- Taeuber paradox (Keyfitz)

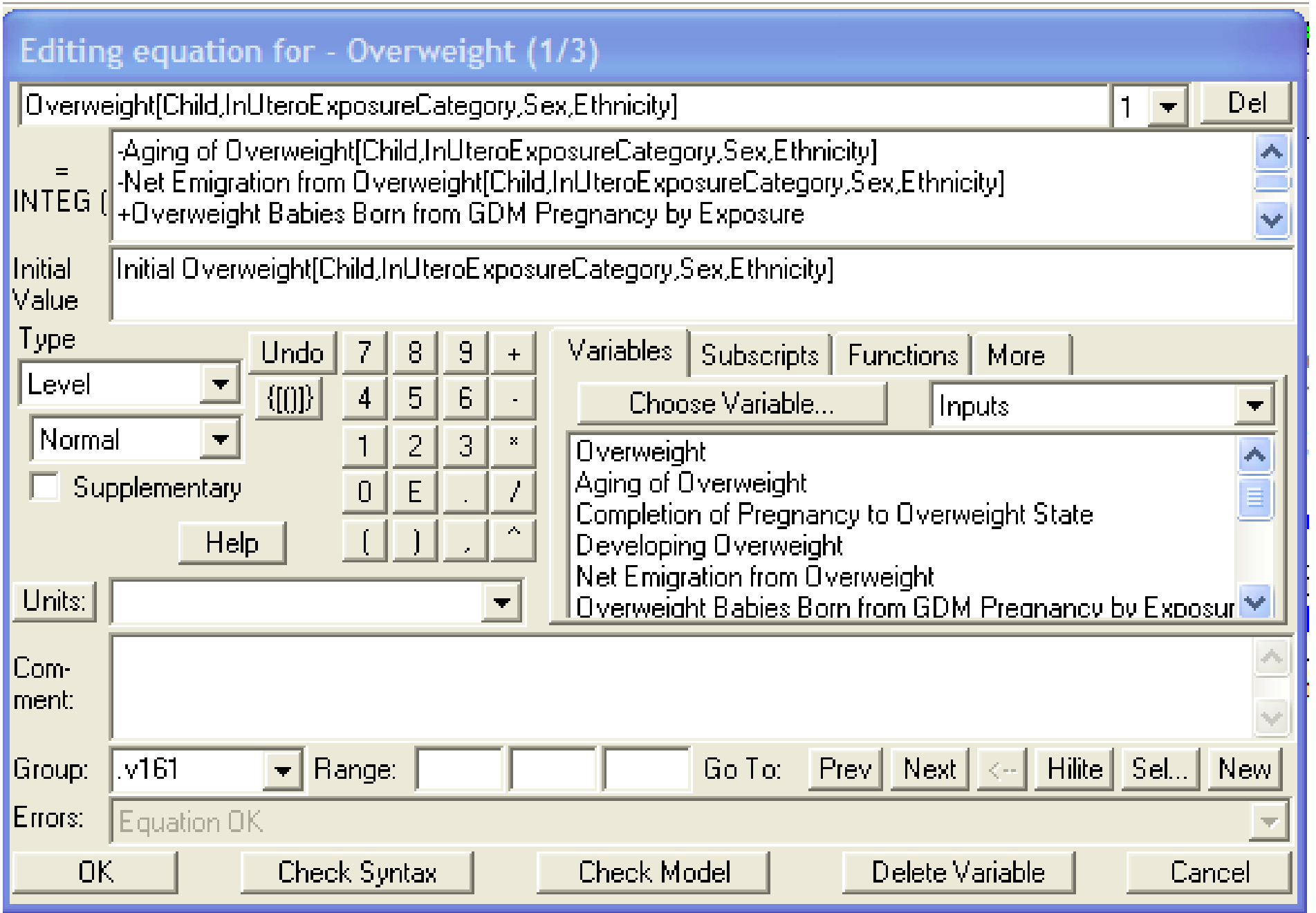
Heterogeneity & Equity Considerations

- Failure to disaggregate (to represent heterogeneity) can impose implicit value judgements! e.g.
  - Treating situation as net zero cost if favouring group A while disadvantaging group B

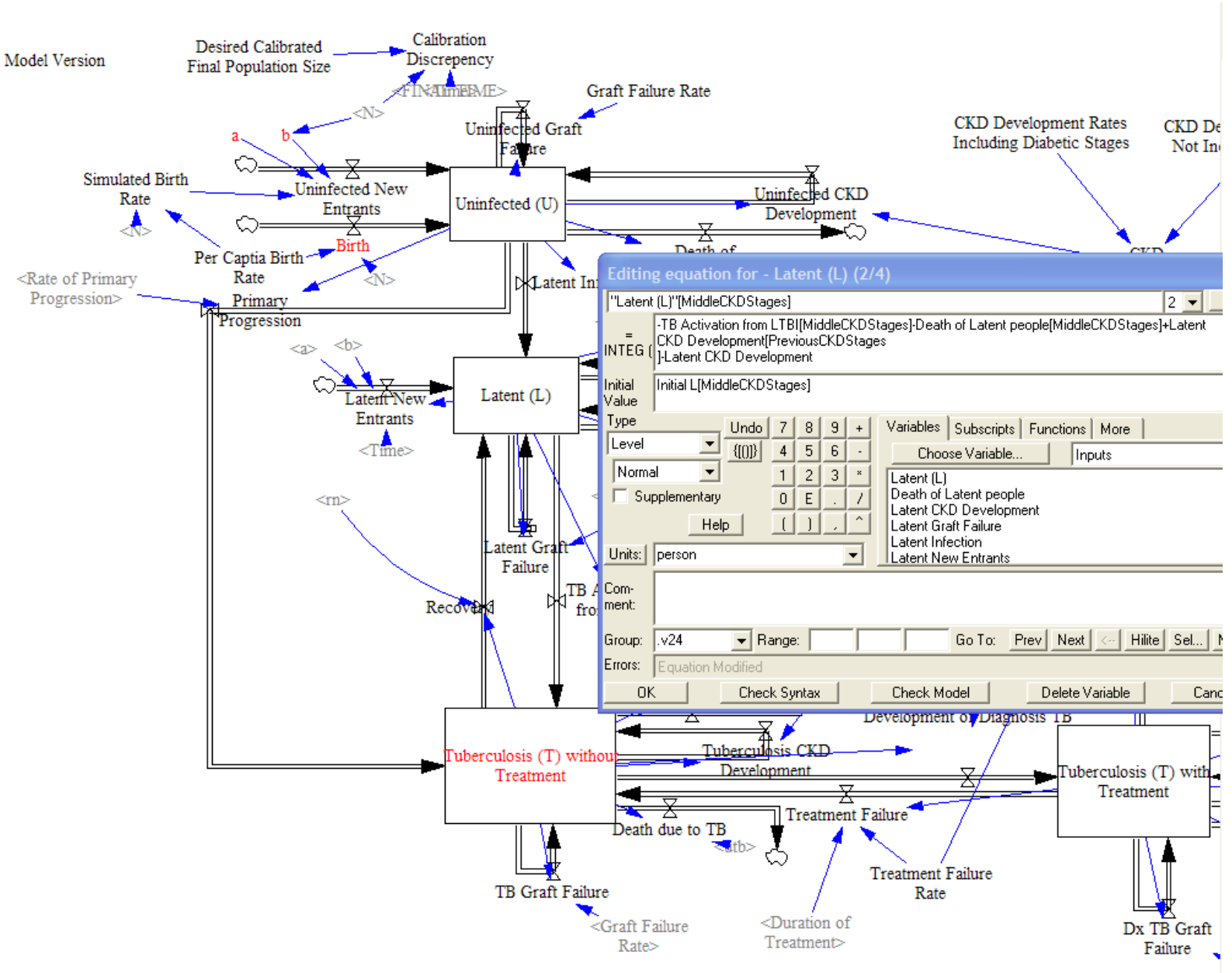
Challenges for Aggregate Model Formulation: Heterogeneity

- Two aggregate means for representing heterogeneity are limited:
  - Attribute-based disaggregation (e.g. via subscripts)
    - Need *n* dimensions to capture individual state with respect to *n* factors of heterogeneity
    - **Poor (geometric) scaling to large # dimensions**
    - Global structural, equation changes required to incorporate new heterogeneity dimensions
    - Awkwardness in stratifying
  - Co-flows
    - Efficient and precise but highly specialized

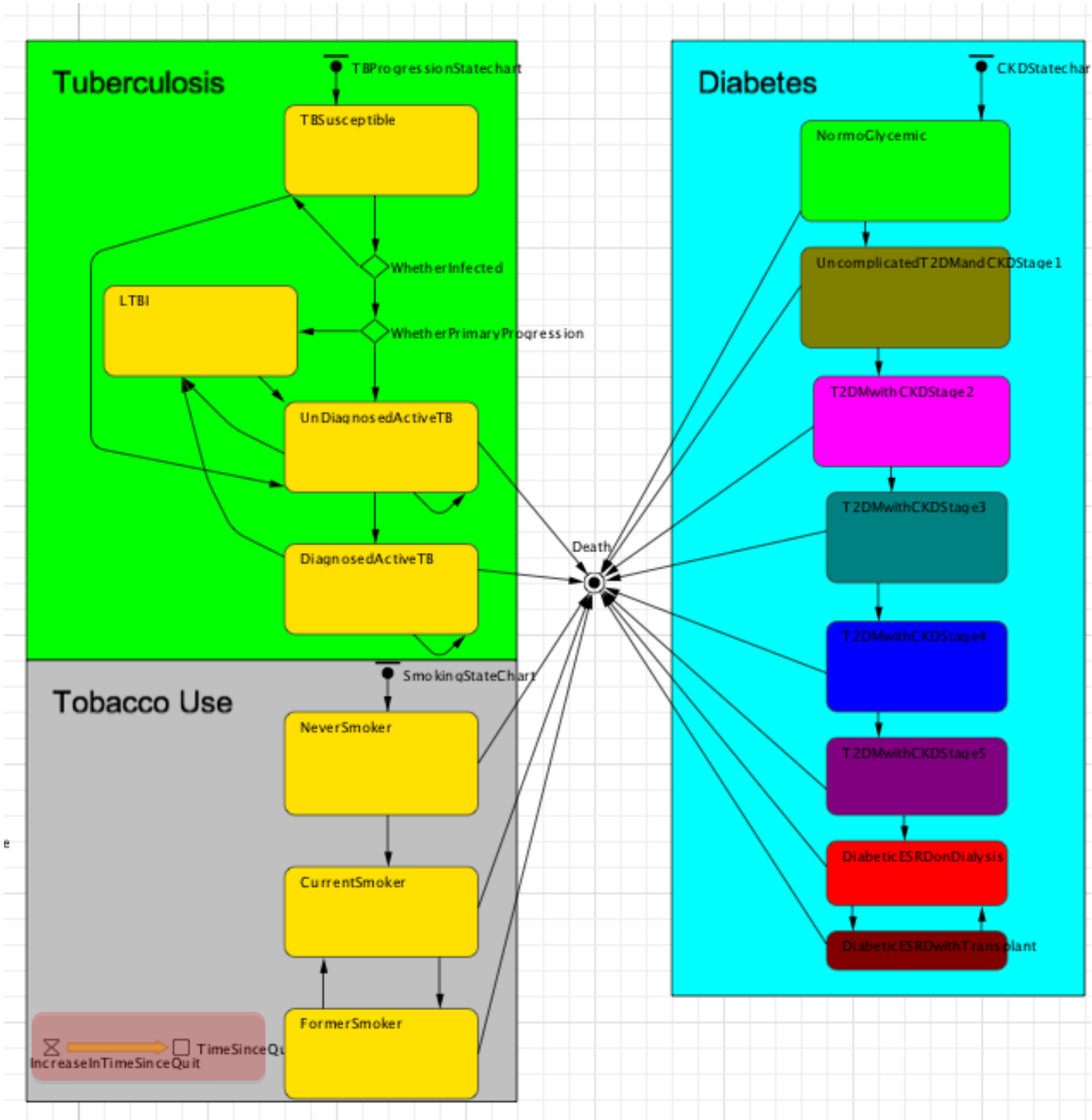
Fragility of Multi-Dimensional Subscripting



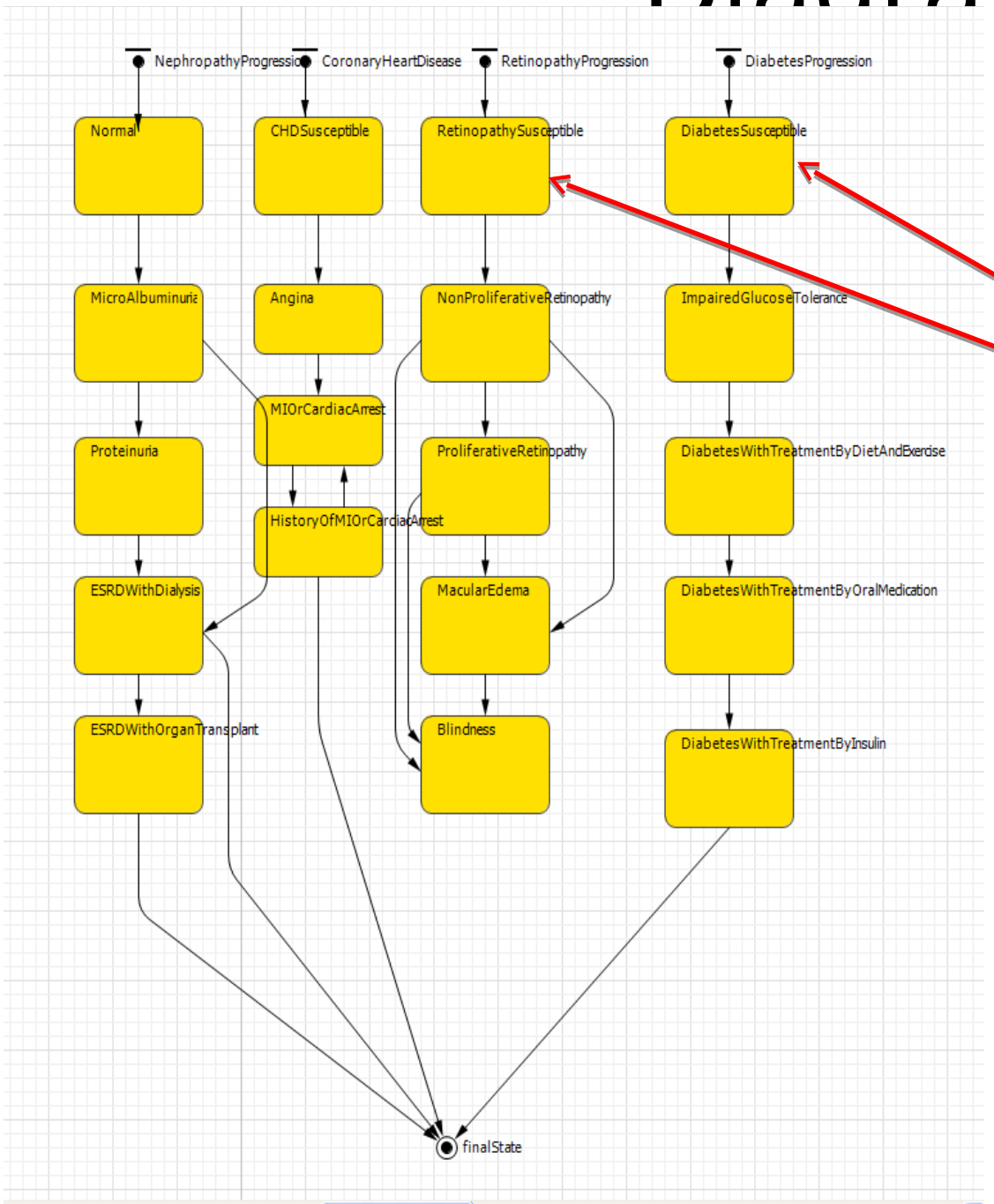
Combinatorial Subscripting: Multi-Dimensional Progression



Parallel Transitions



Parallel State Transition Diagrams



A person is in some particular state with respect to each of these (condition specific) state transition diagrams

This requires representing combinations of possibilities in an aggregate model

Capturing Heterogeneity in Individual-Based vs. Aggregate Models

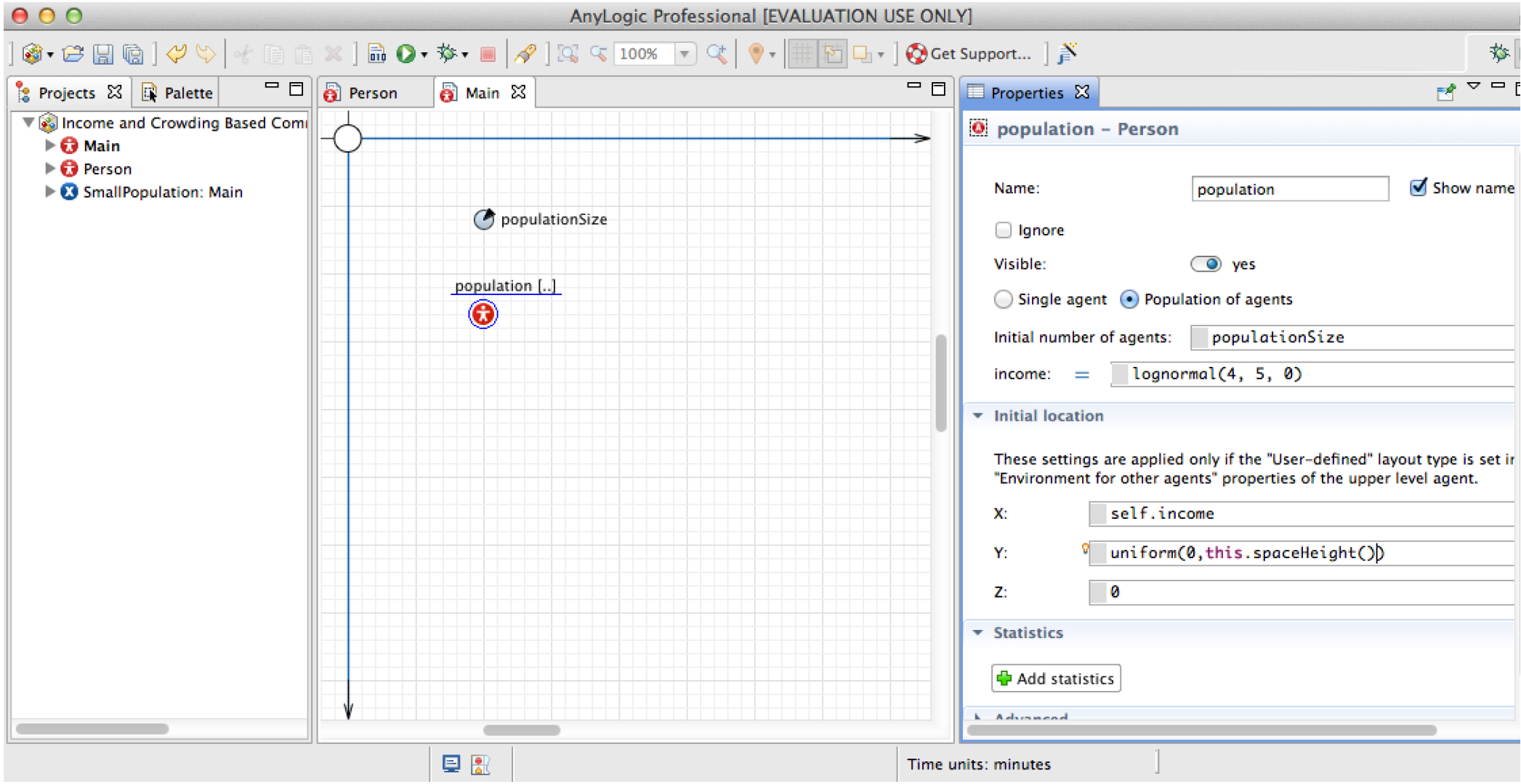
- Consider the need to keeping track a new piece of information for each person (with d possible values)
  - E.g. age, sex, ethnicity, education level, strain type, city of residence, etc.
- Aggregate Model: Add a subscript
  - This multiplies the model size (number of state variables into which we divide individuals) by *d*!
- Individual based model: Add field (variable/param)
  - If model already has c fields, this will increase model size by a fraction 1/c.

Desired: Flexibility in Representing Heterogeneity

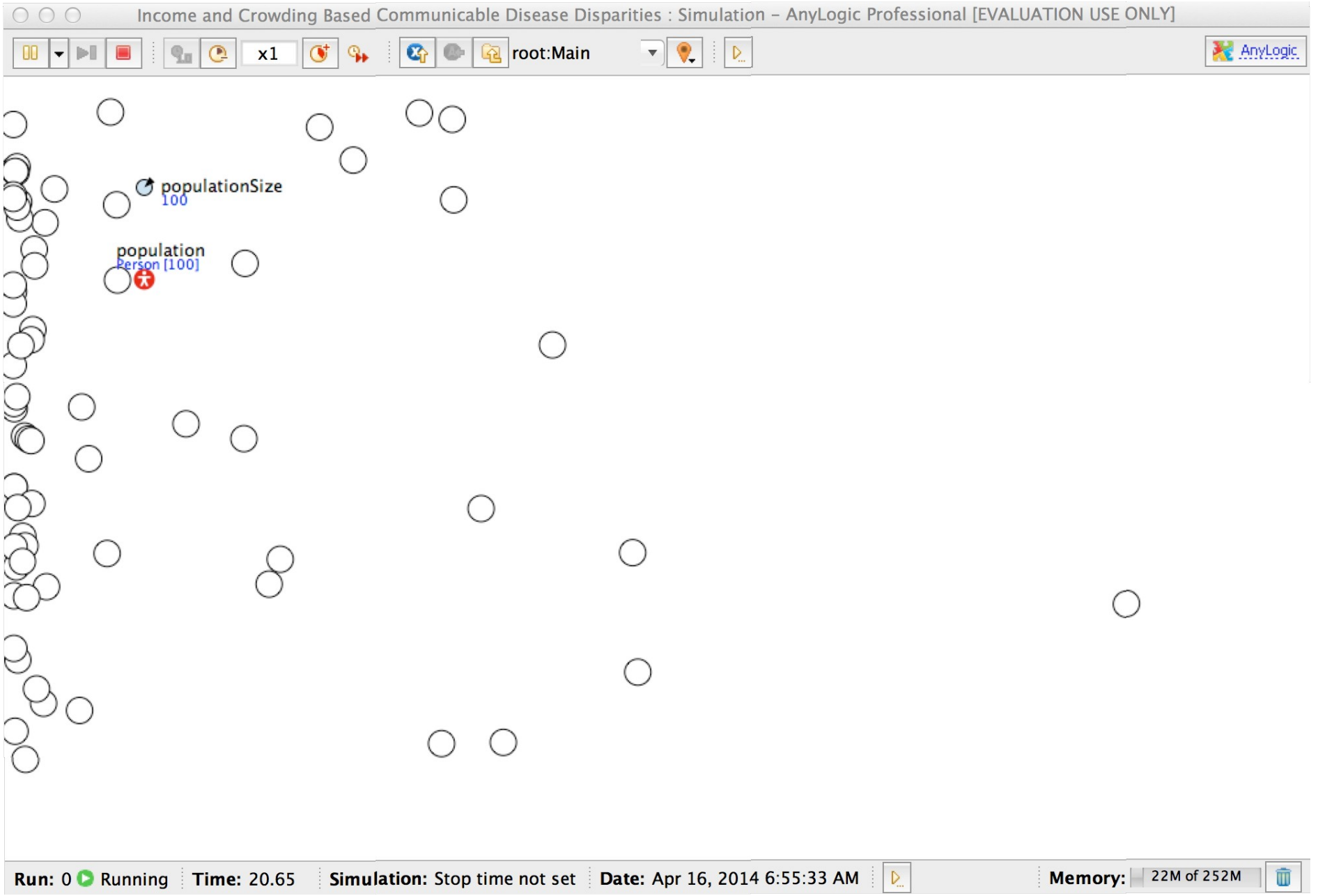
- It is desirable to capture heterogeneity in a flexible fashion.
- More judicious exploration of whether to represent heterogeneity
  - Examine whether some observed covariation might simply be due to colinearity
    - Represent added heterogenity dimensions with no causal interaction, see if model covariations matches what is seen in external world
      - e.g. represent age in a TB model, see if rates of LTBI by age in the model match age-specific infection rate observations
  - Try adding in new dimension of heterogeneity & effects, and see if has impact that is both substantive & plausible

Hands-On Components: Spatial Embedding

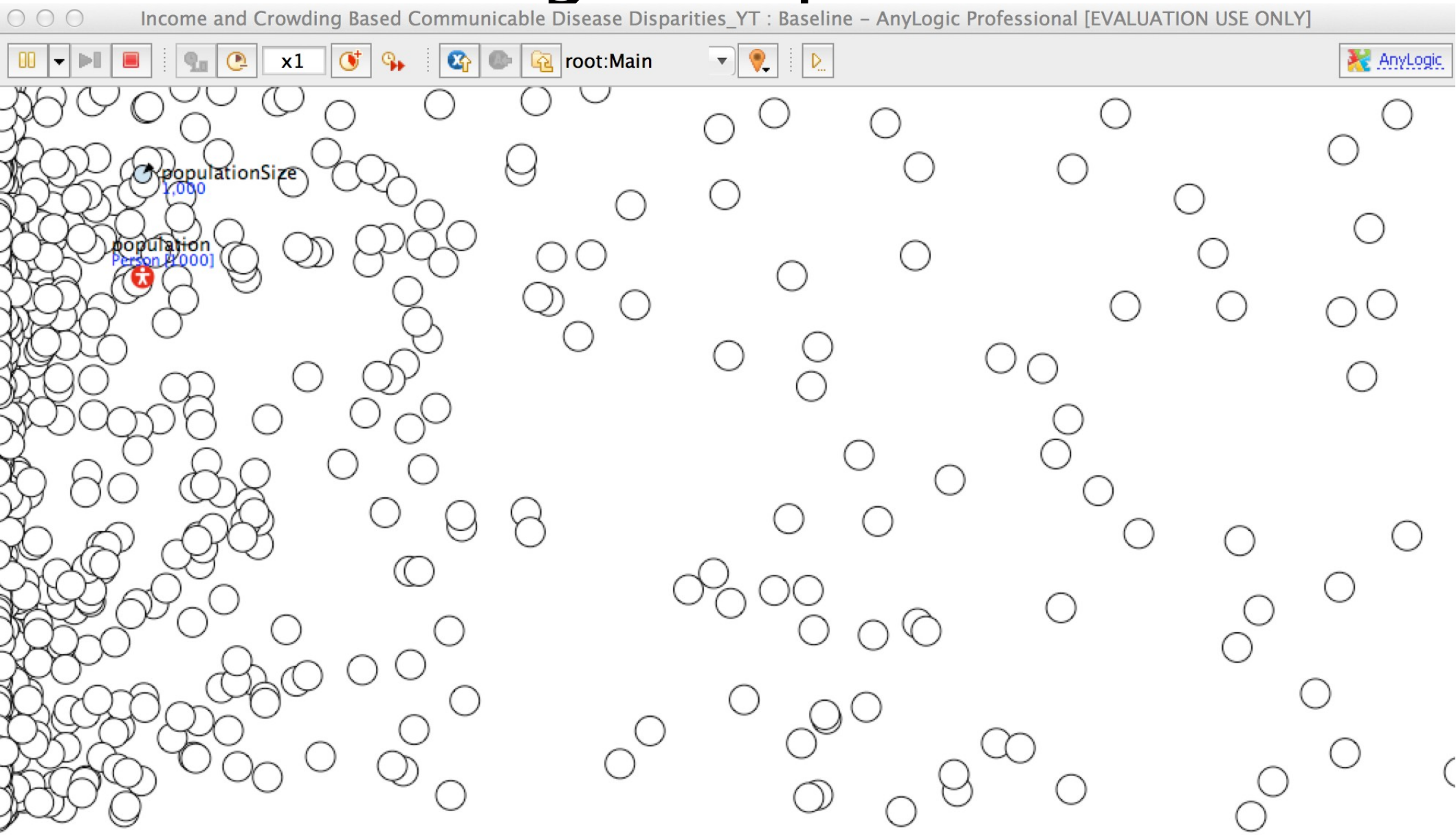
Linking Location for each Agent to their Income



Income Segregation



A Larger Population



2D Spatial Embedding: Two Options

- Continuous embedding (e.g. Wandering elephants, our built-up model)
  - No physical exclusion: Agents are assumed to be small compared to landscape scale, and exhibit arbitrary spatial density without interfering
  - We have seen this much with distributing agents initially around the space, adding agents in
- Discrete cells (e.g. The Game of Life, Agent-based predator prey, Schelling Segregation)
  - Divided into “Columns” and “Rows”
  - Physical exclusion: Only one agent in a cell at a time

Hands-On Components: Network Context

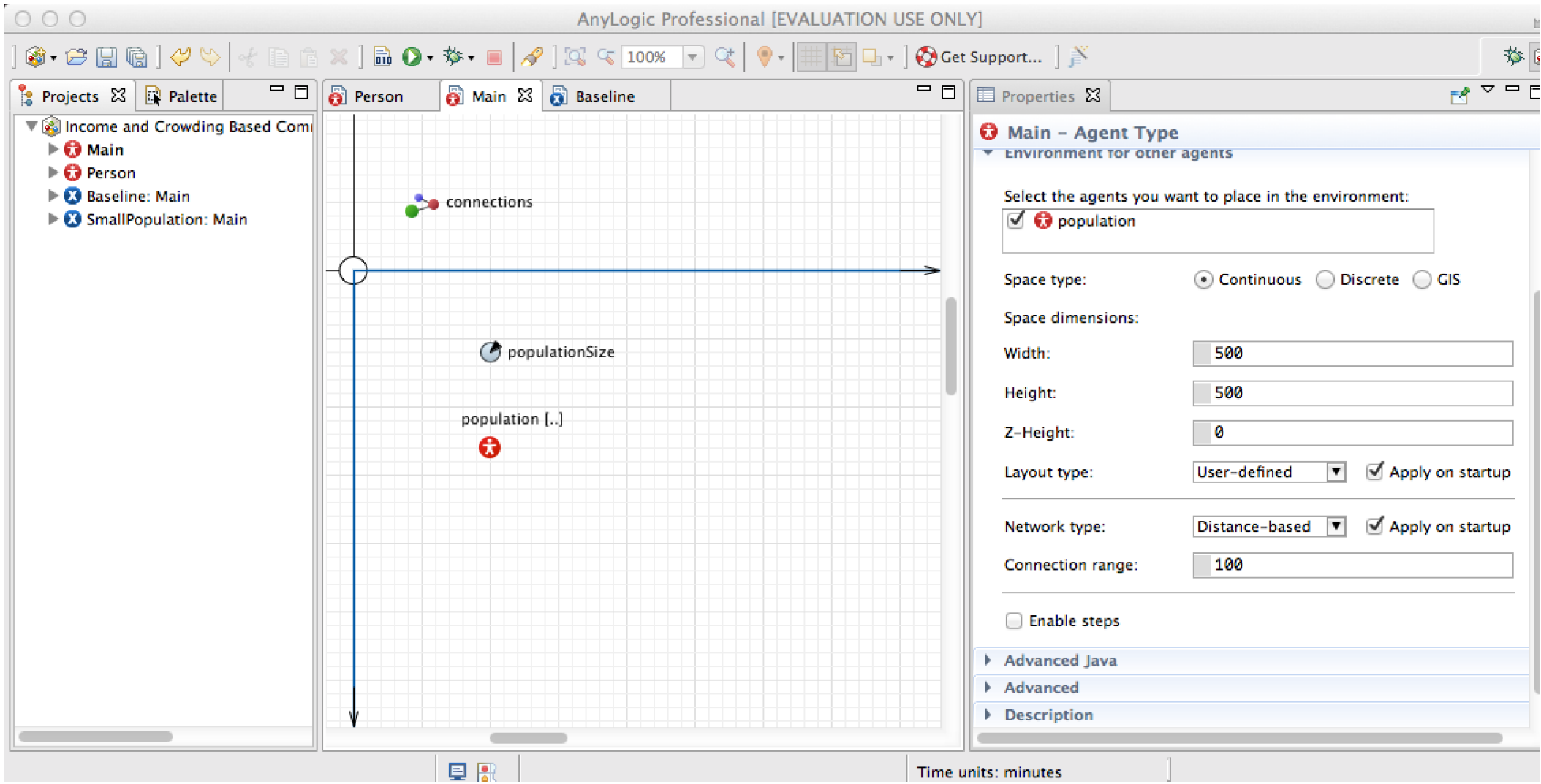
Networks & Spatial Layouts

- Distinct node attributes: Location & connections
  - Spatial layouts determine where nodes appear in space (and on the screen!)
  - Network type determines who is connected to whom
  - For the most part, these characteristics are determined independently
- Network topologies (connectedness) can be defined either *alternative to* or *in addition to spatial layouts*
  - Agents will have spatial locations in either case

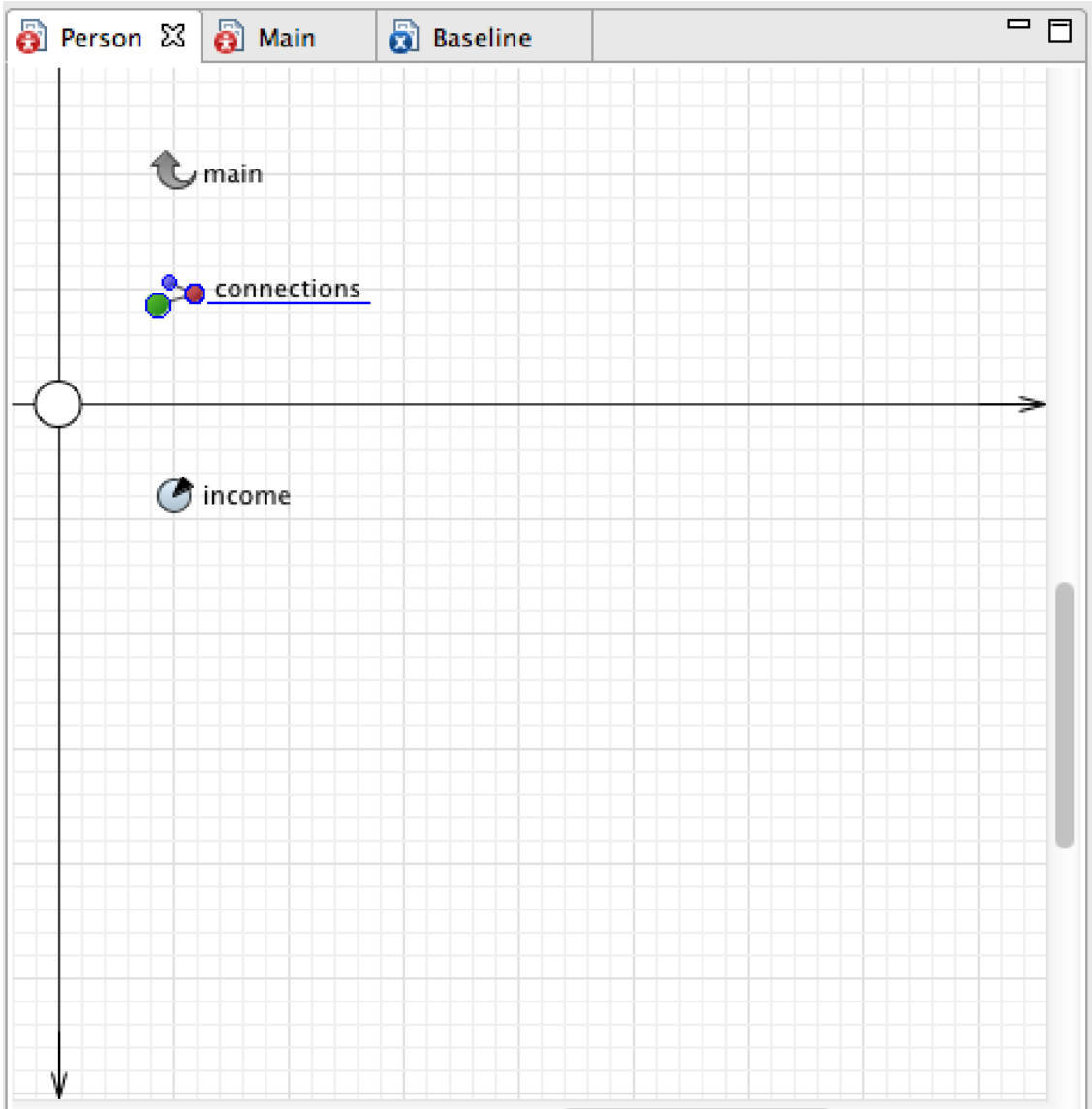
Common Supported Networks

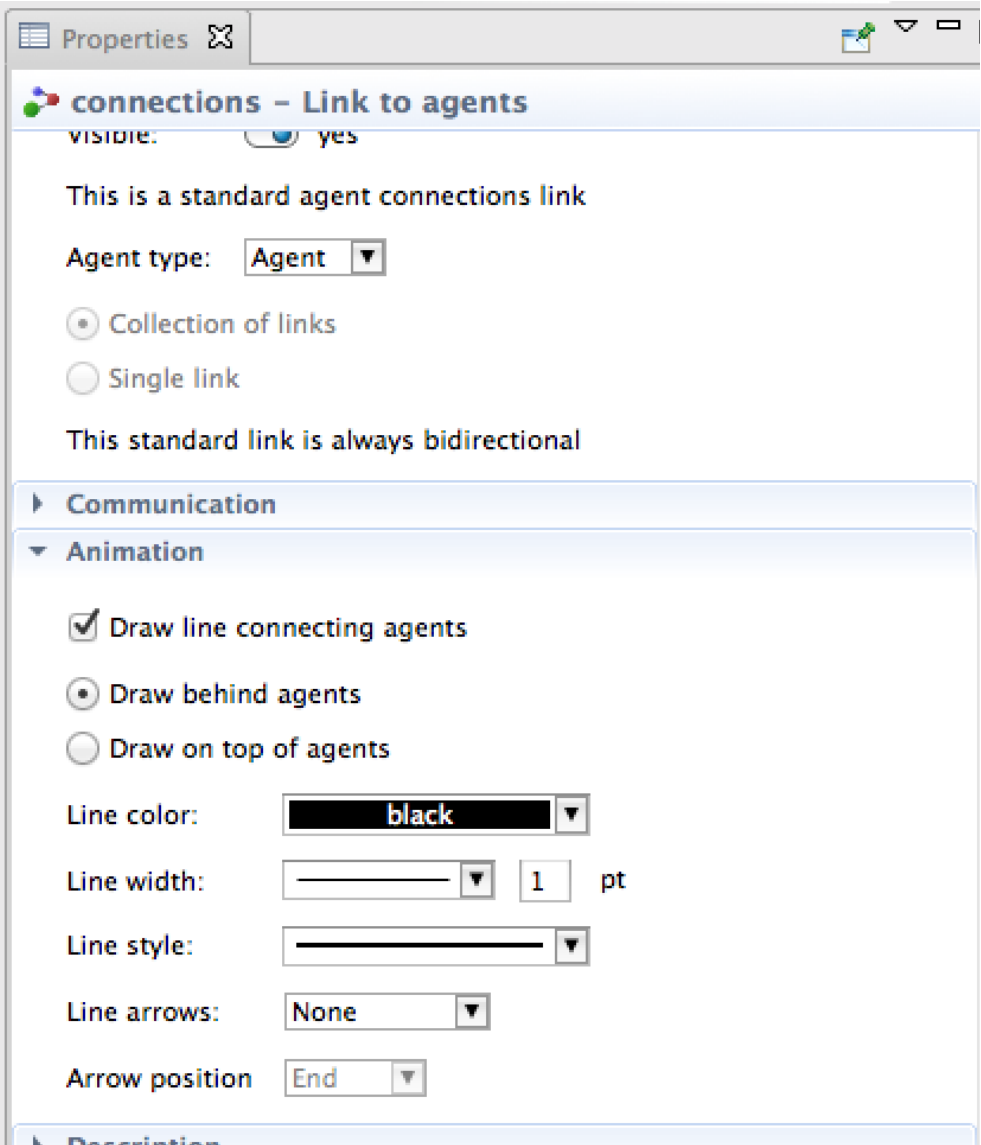
- Highly localized
  - Distance based (spatial locality)
  - Ring lattice (network locality)
- Poisson Random
  - Disordered
  - Global connections – no sense of locality
- Small world : Mix of global (poisson random) and ring lattice
- Scale free: Power-law distribution for # of connections (“long tail” to the right)

Defining a Network

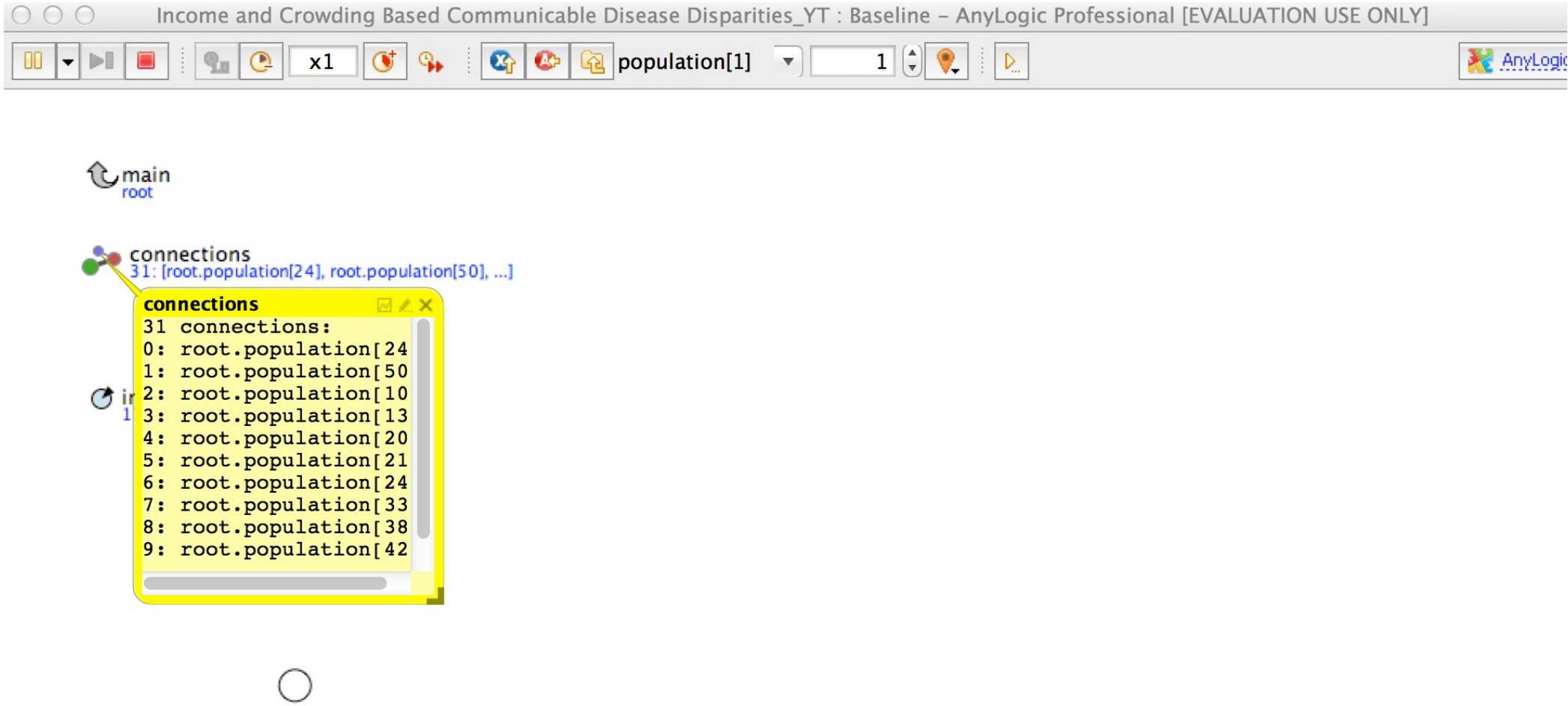


Setting each Person to be Visually Connected to Network Neighbors

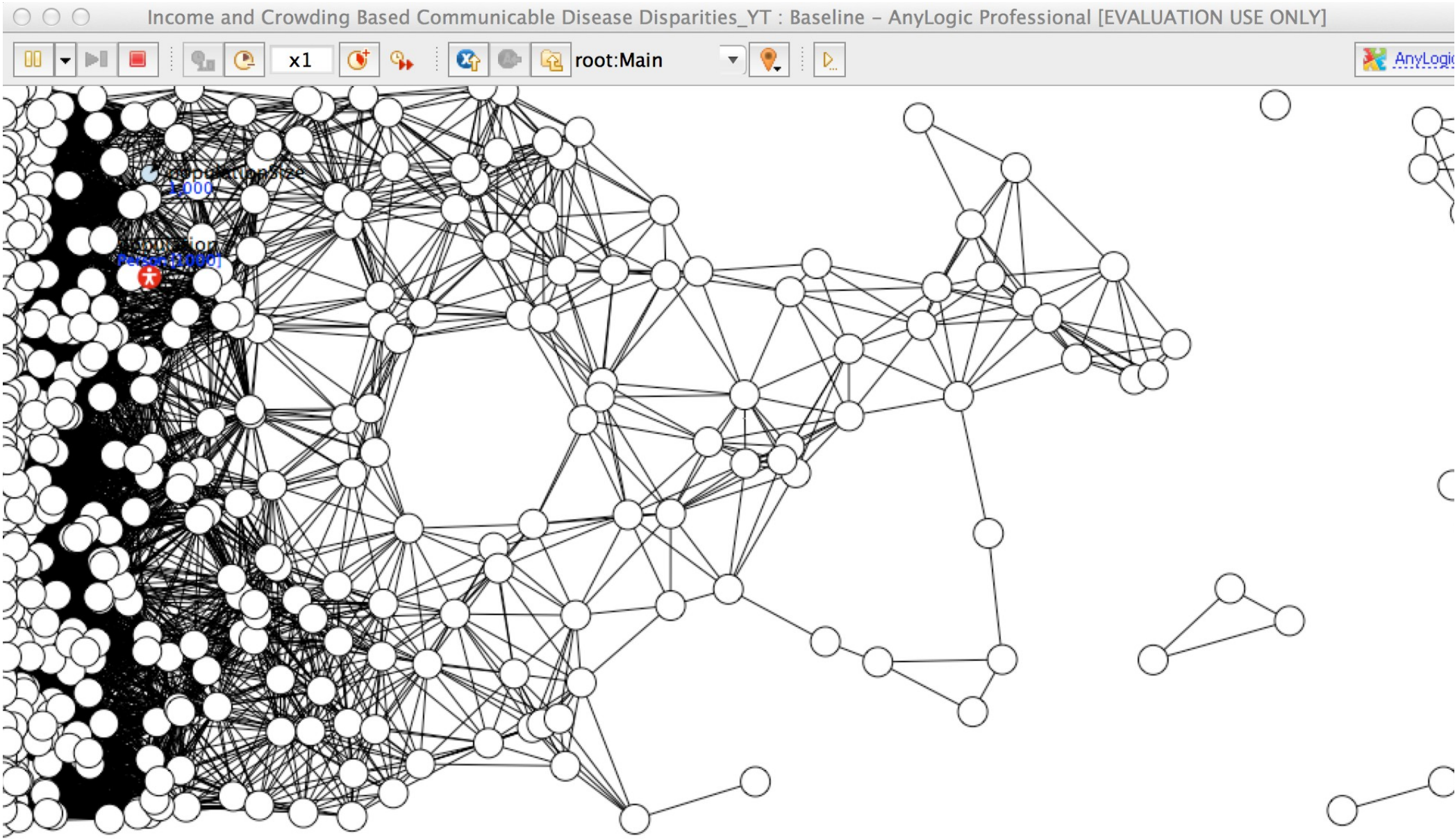




Examining the Neighbour of a Particular Person



Displaying the Network Structure



Slide Template

The Locus of Control: Environment

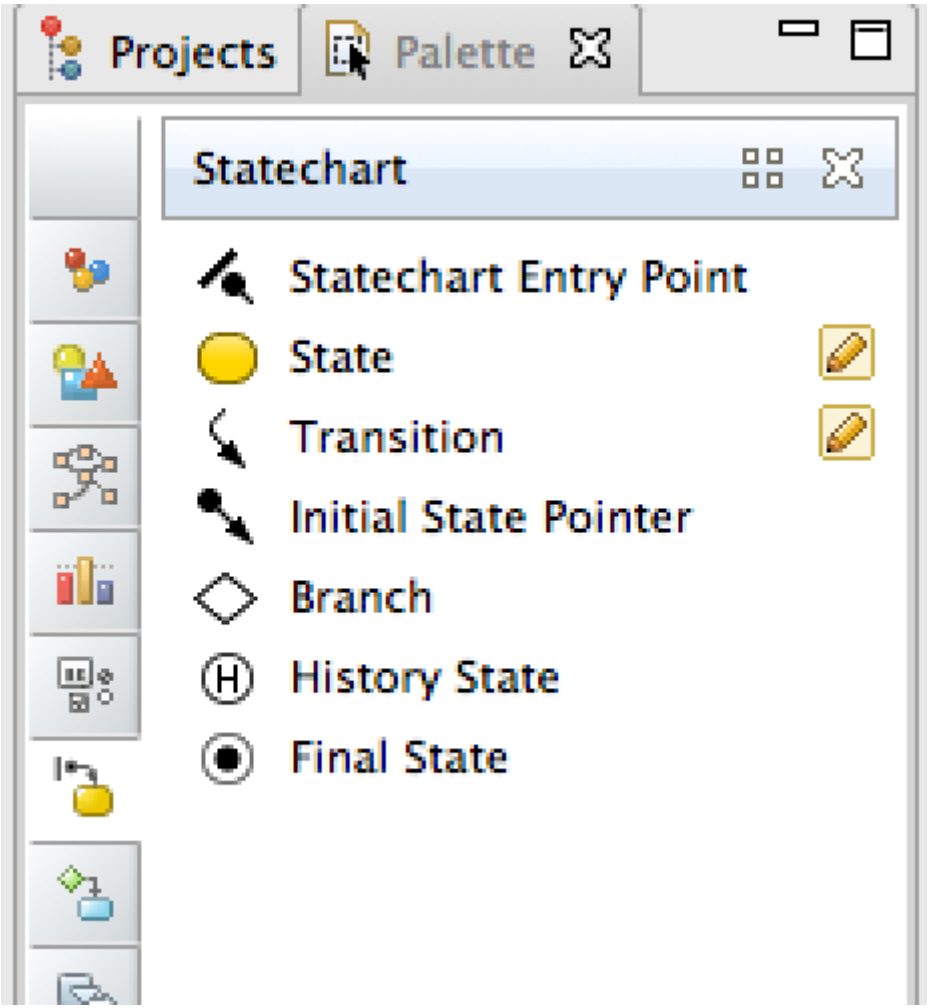
- The Anylogic Environment sets the parameters for the nature of the 2D landscape
  - Width
  - Breadth
  - Continuous vs. Discrete
  - Character of discrete neighbourhoods (cardinal directions vs. Euclidian { N,NE,E,SE,S,SW,W,NW}

Recording of Results

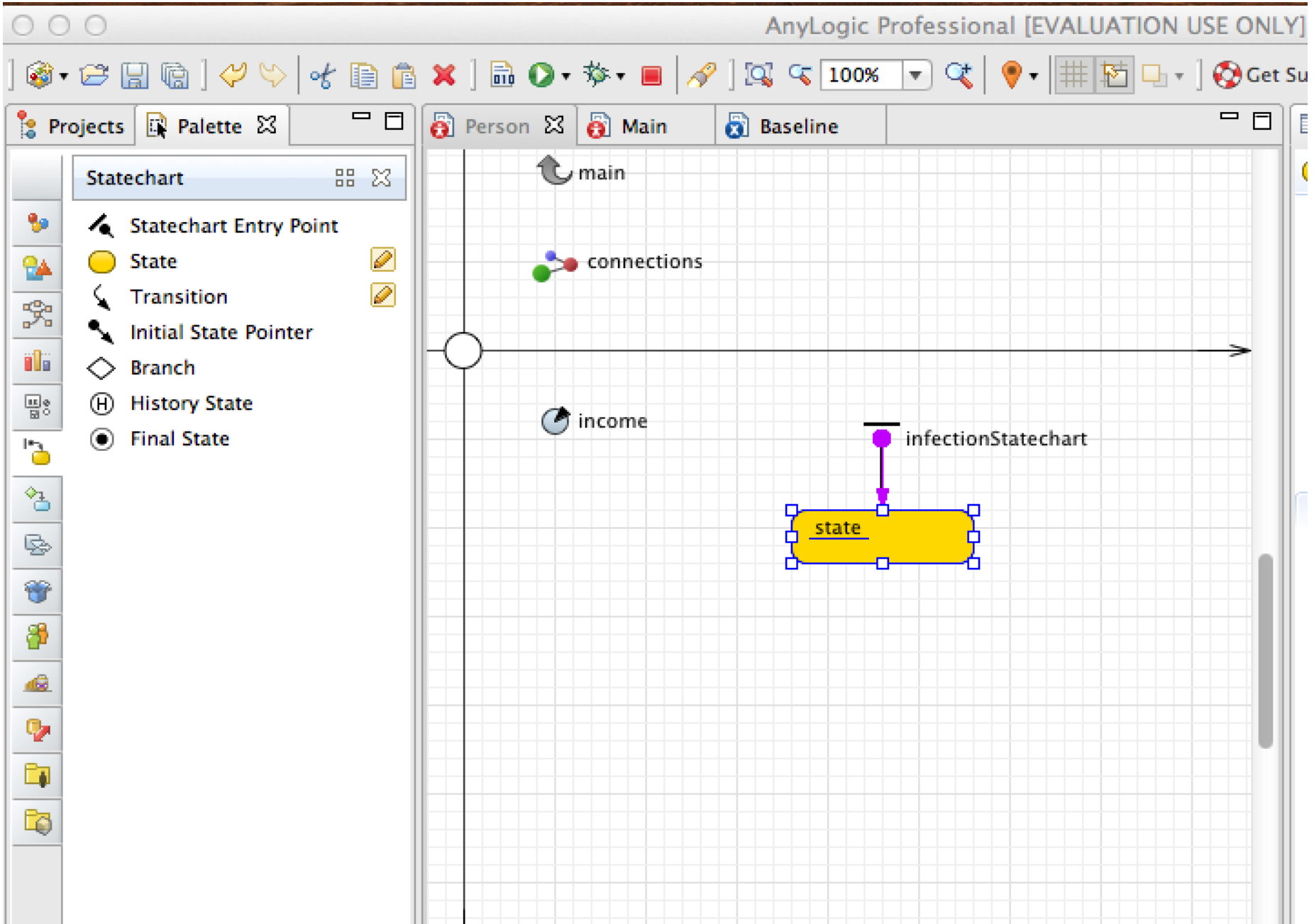
- A frequent modeler need is to record some components of model state over time
  - State variables (e.g. stocks)
  - States of agents
  - Summaries of model state
  - We informally term this a “trajectory file”
- *Trajectory recording is supported in higher AnyLogic versions*
- All versions of AnyLogic 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

Hands-On Components: Discrete Intra-Agent Interaction

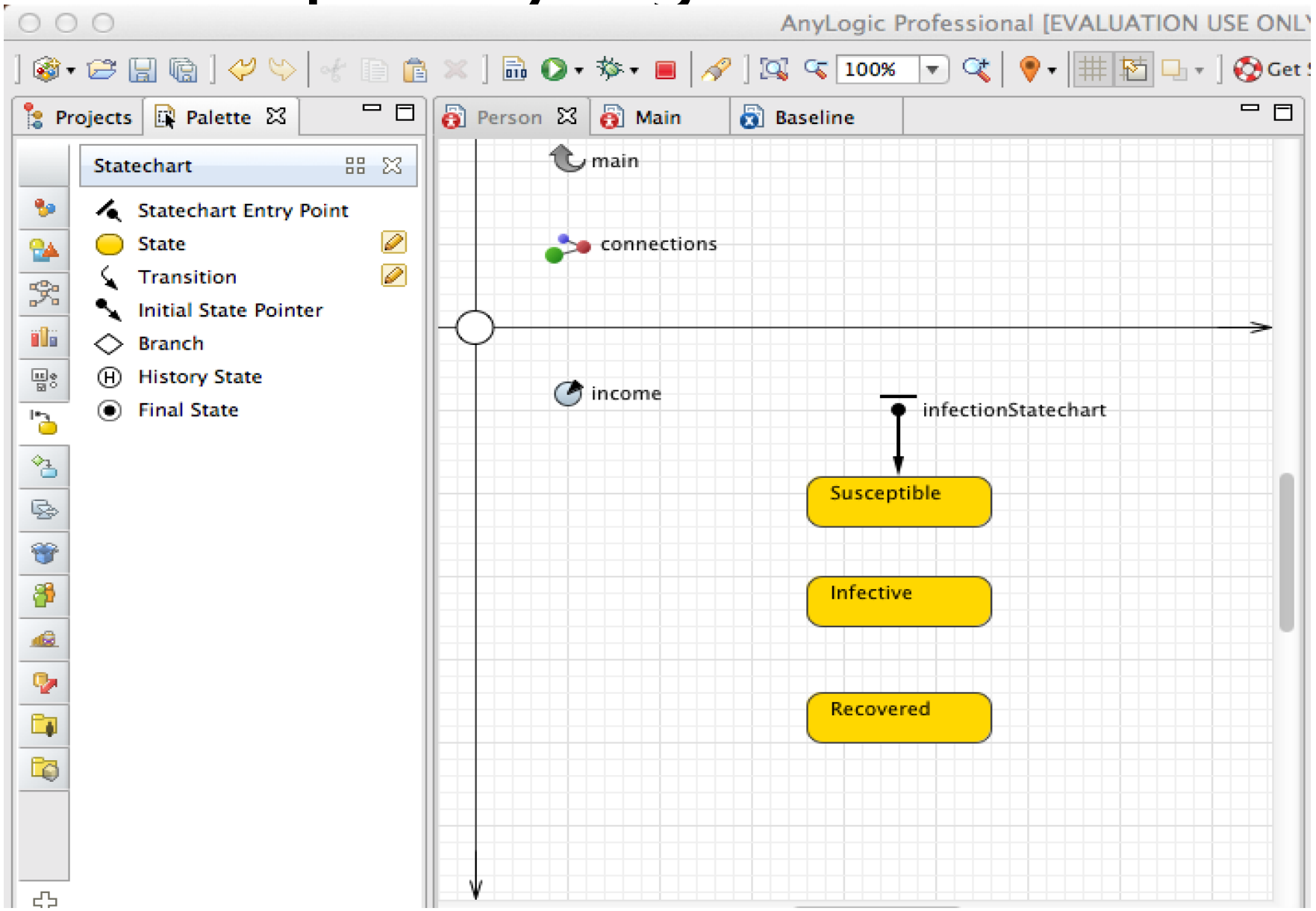
Adding Statechart Elements from the Palette



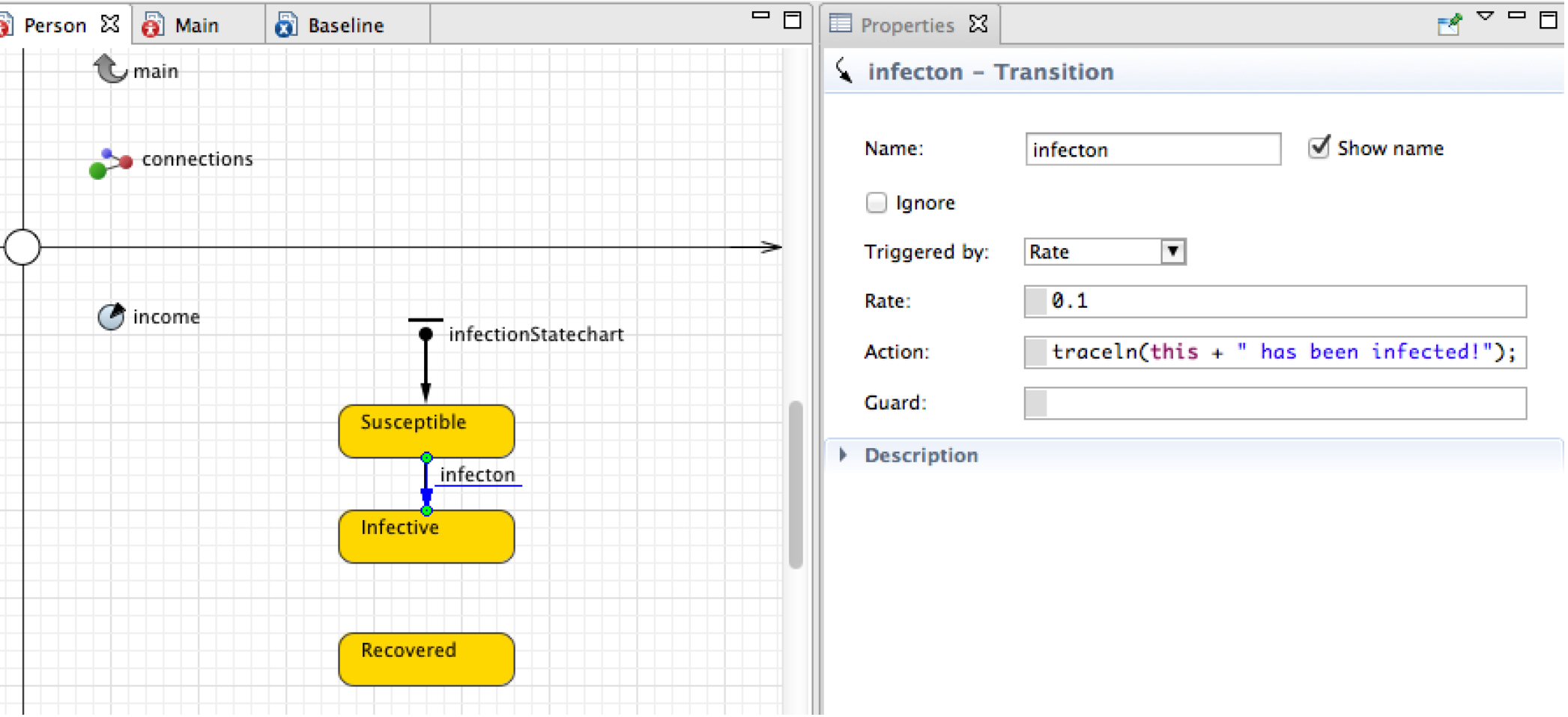
Starting the Statechart Construction



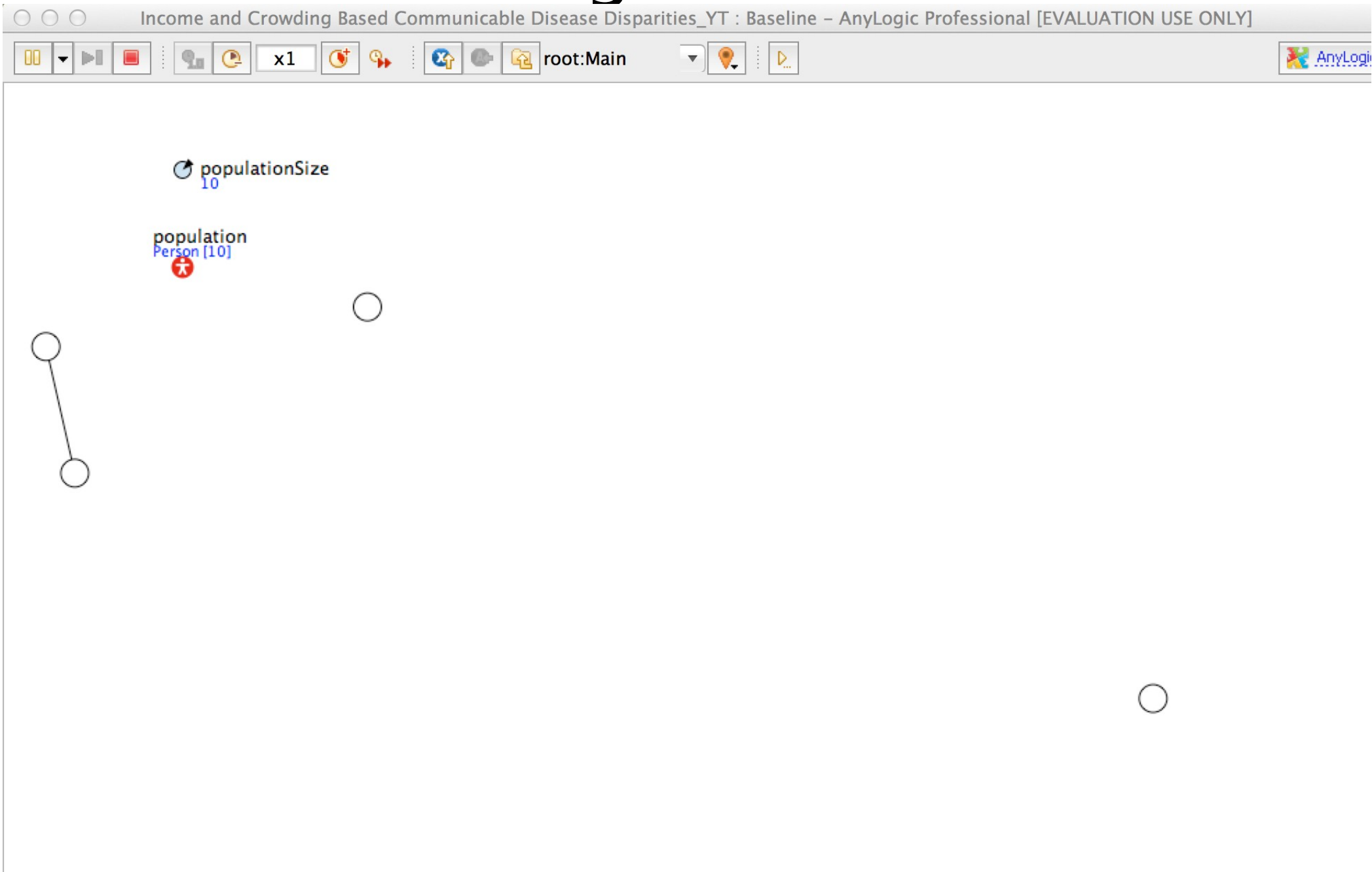
Specifying the States



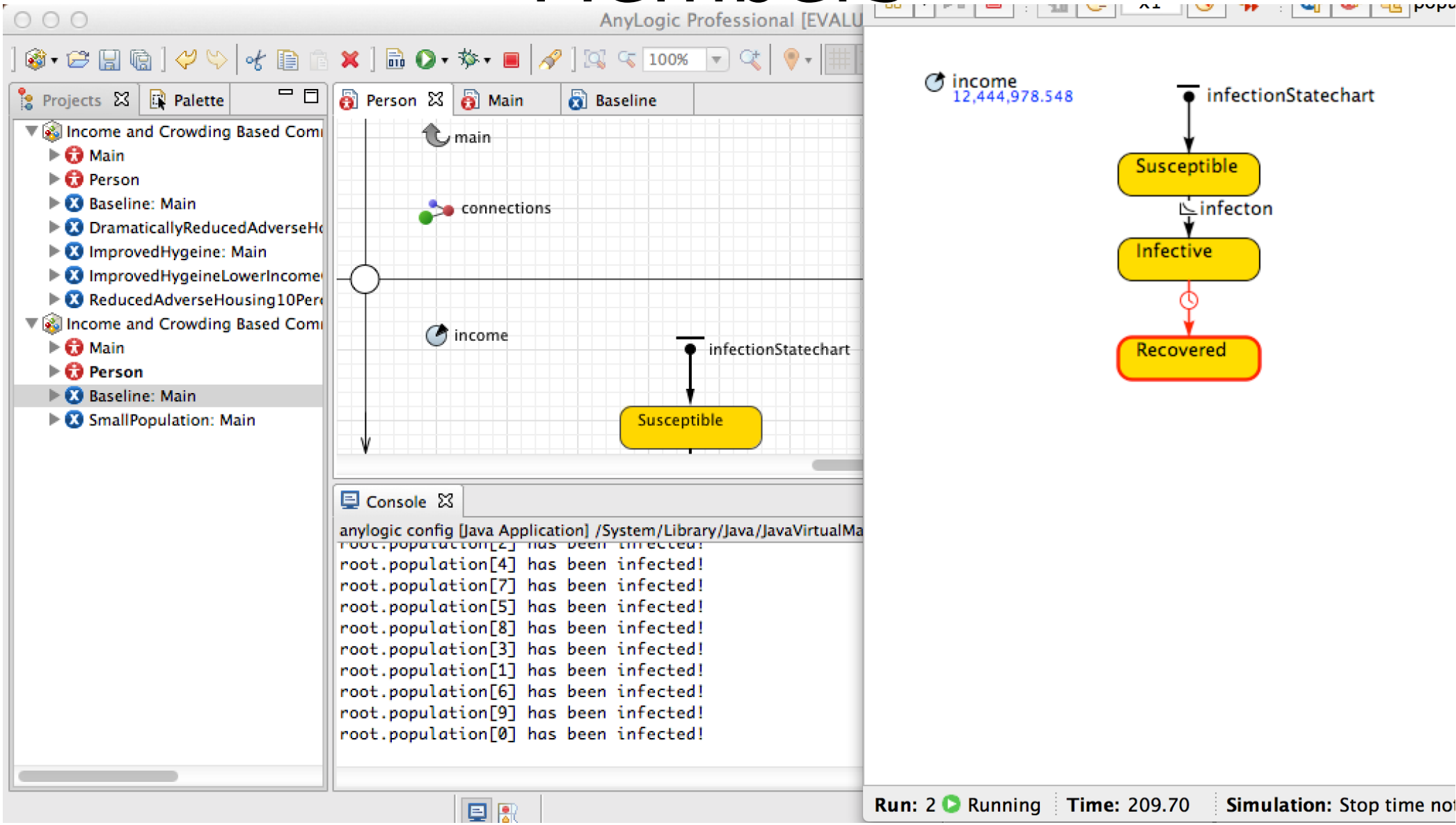
Specifying an Initial Transition



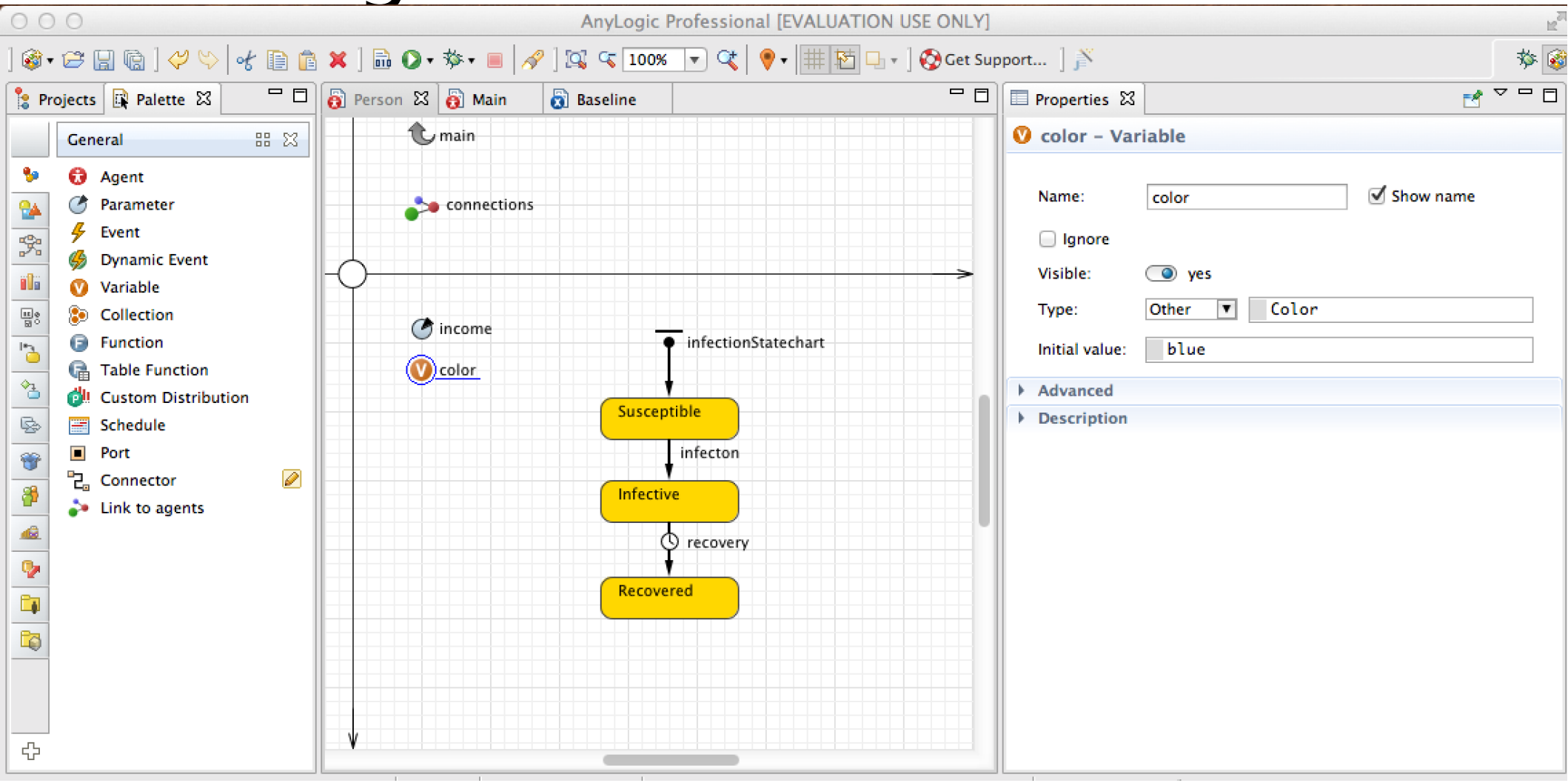
Running the Model



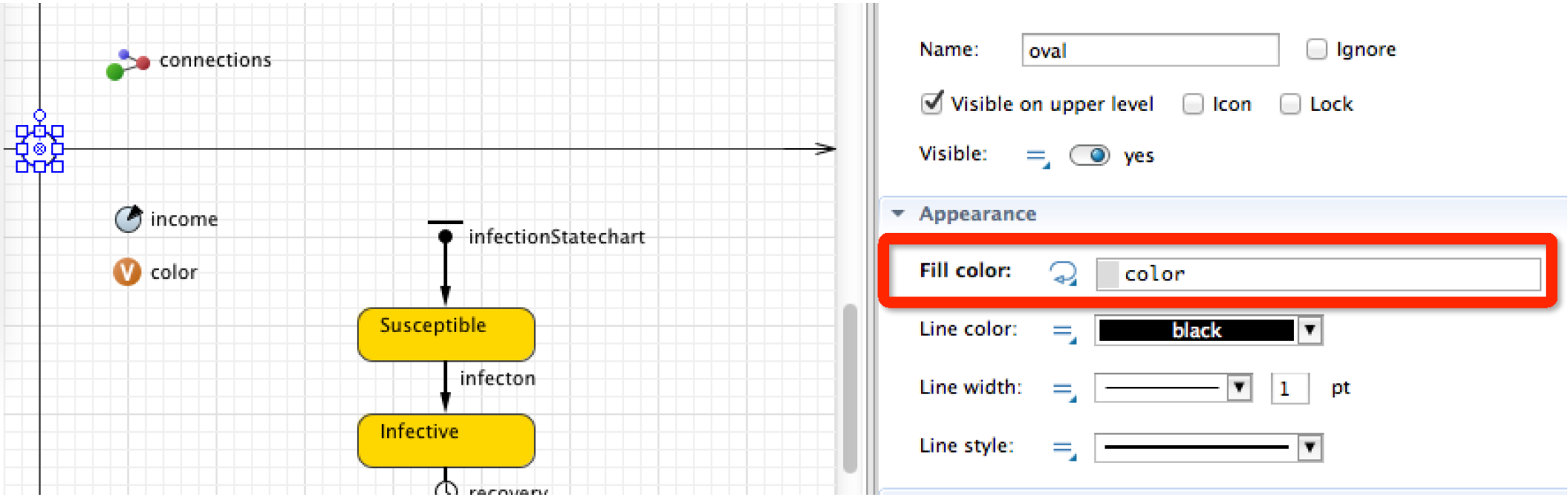
Drilling down to Population Members



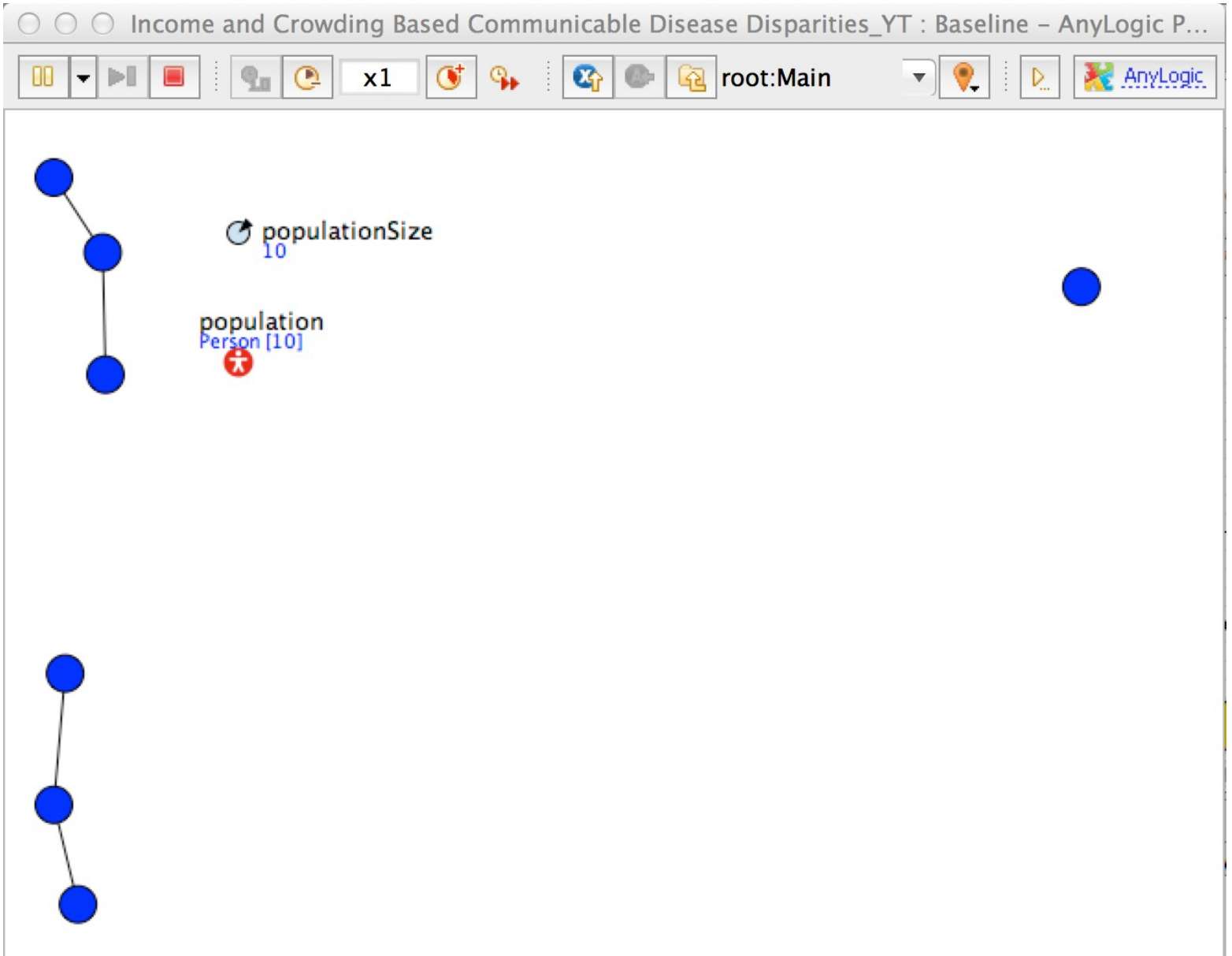
Setting up a Variable to Specify Agent Color over Time



Putting in Place Logic to Make the Oval Use the Specified Color

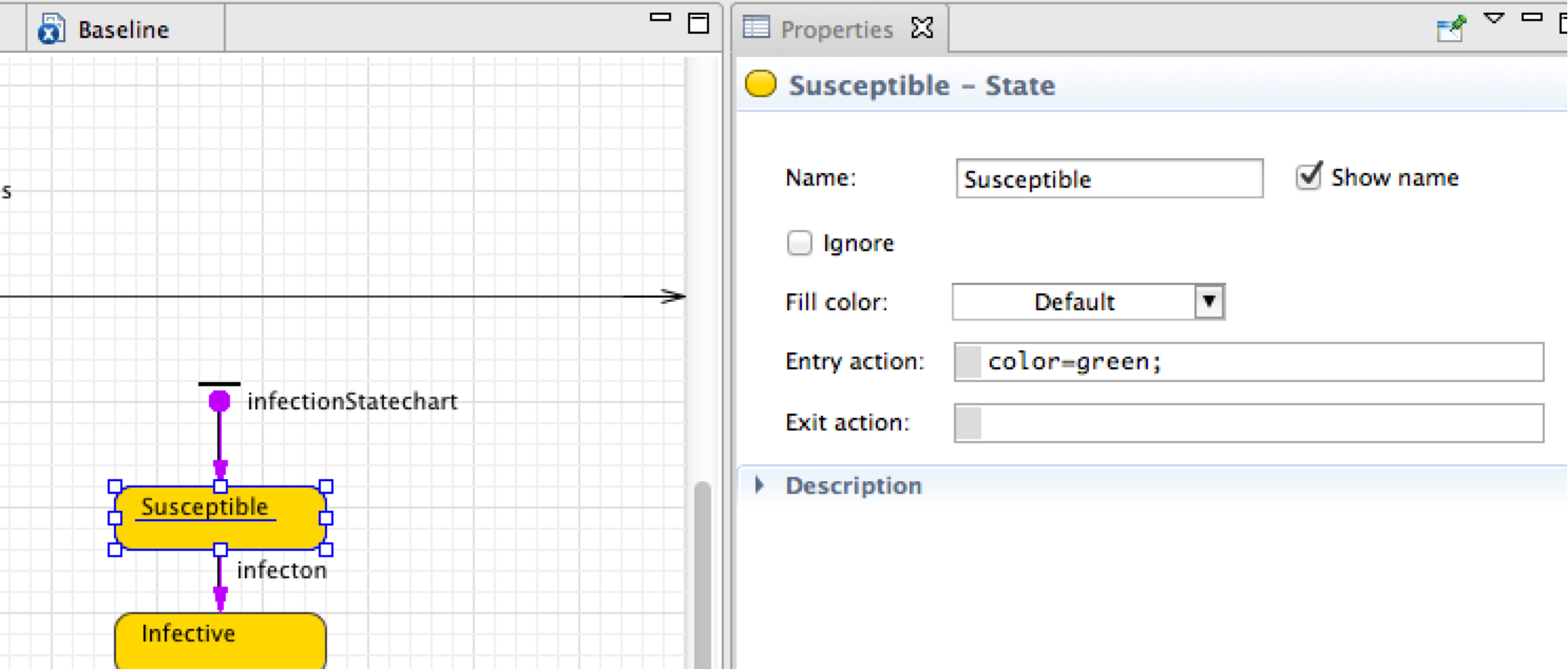


Result

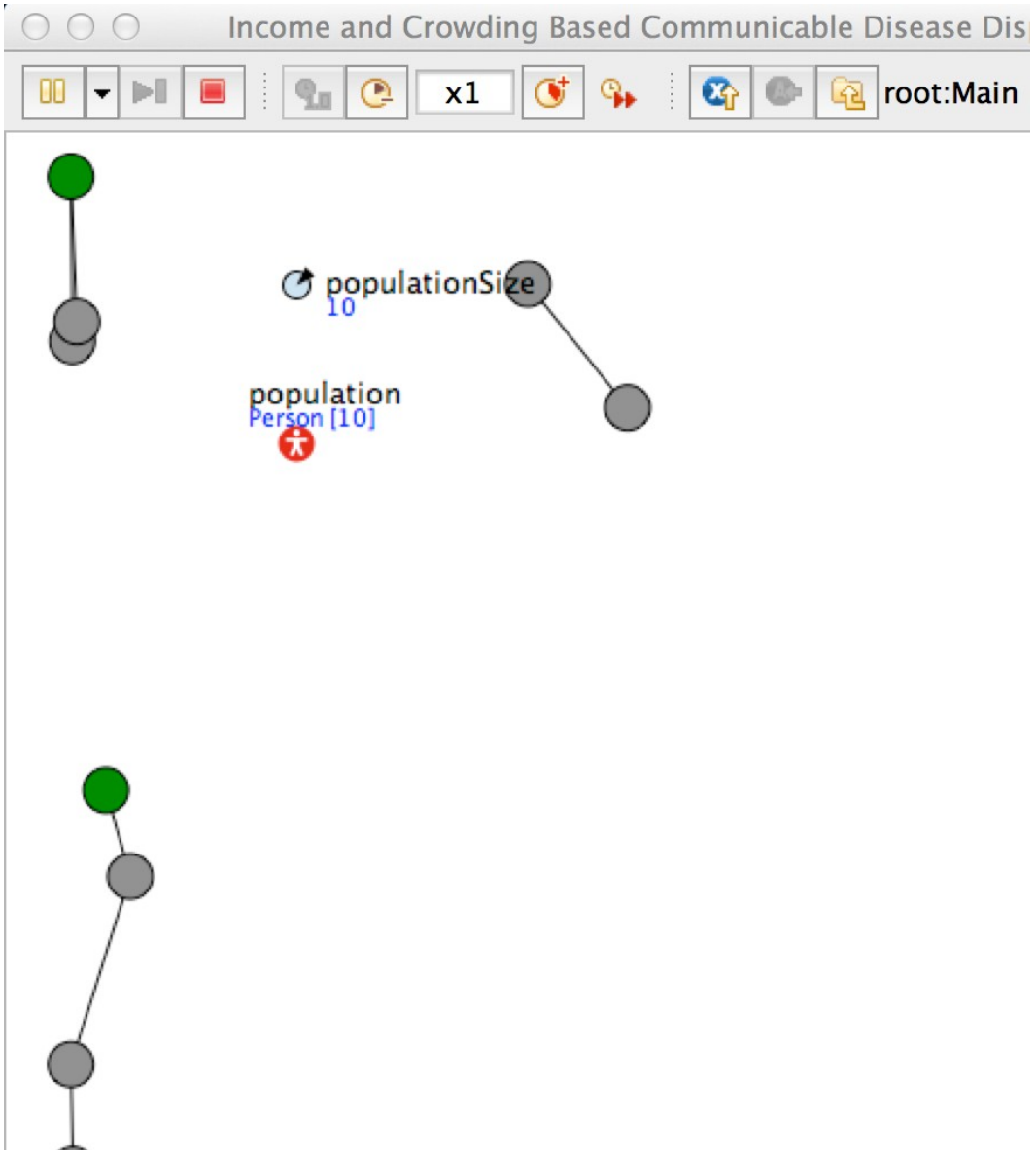

Income and Crowding Based Communicable Disease Disparities_YT : Baseline – AnyLogic P...
x1
root:Main
AnyLogic
populationSize
10
population
Person [10]

Hands-On Components: Declarative Specification of Appearance

Incorporating Logic to Update Agent Colors as Agent Evolves



Result of Running the Model



Discrete Agent Dynamics

- Frequently we can represent agent behaviour using as transitioning among a set of mutually exclusive and collectively exhaustive 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

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

Key result: Statecharts are *modular:* You can add a new statechart without modify all the existing

Modularity Disparities

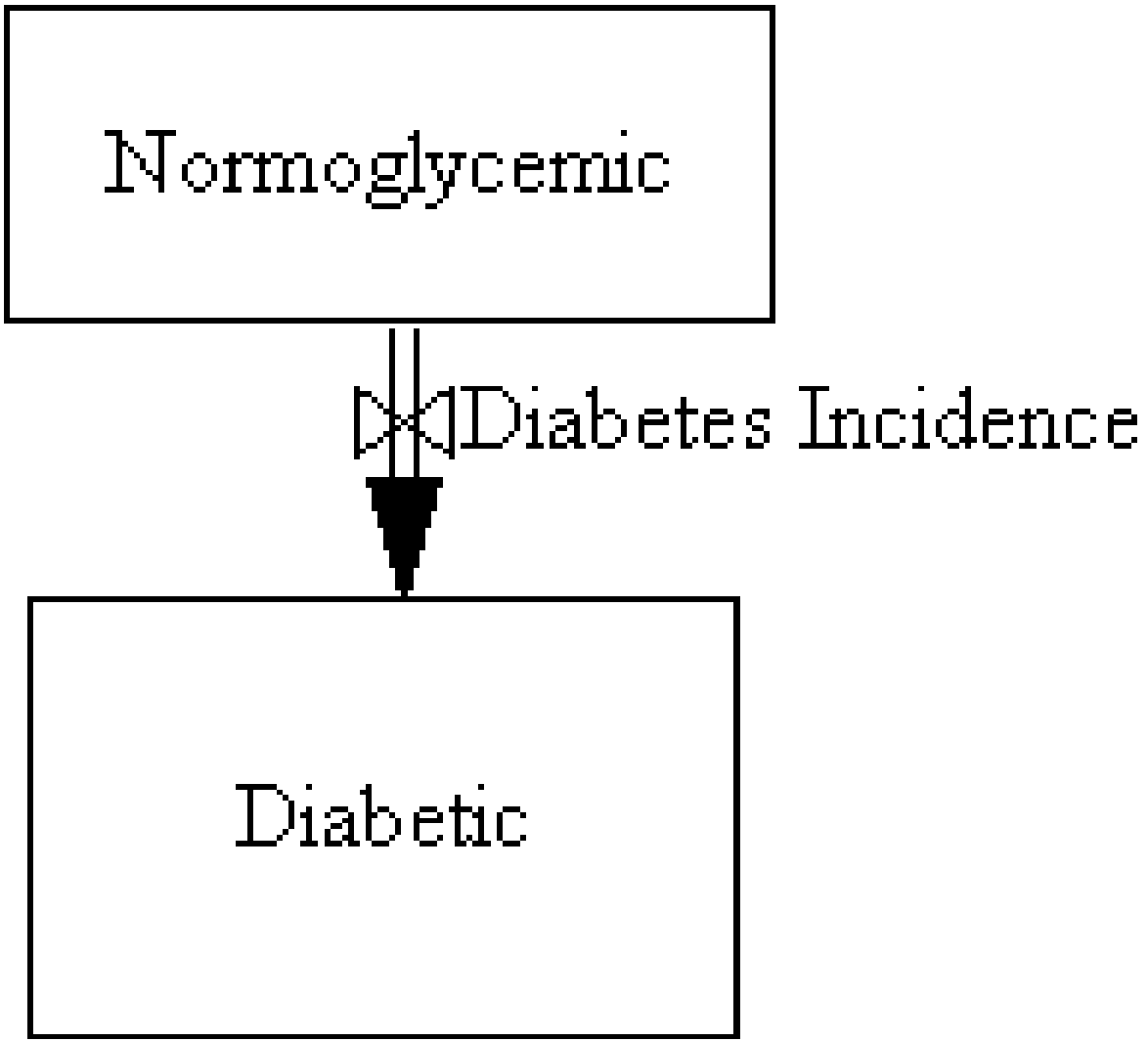
- A consequence of the previous points is that there are vastly different implications for representing new taxonomies in ABM & aggregate models
- Aggregate model:
  - Require representing all state combinations
  - Adding a new division (e.g., to represent an additional comorbidity) entails updating the entire structure
- Individual based model
  - Each statechart is largely orthogonal
  - Statecharts are *modular:* You can add a new statechart without modify all the existing statecharts

Aggregate Non-Solution: Maintain Marginals

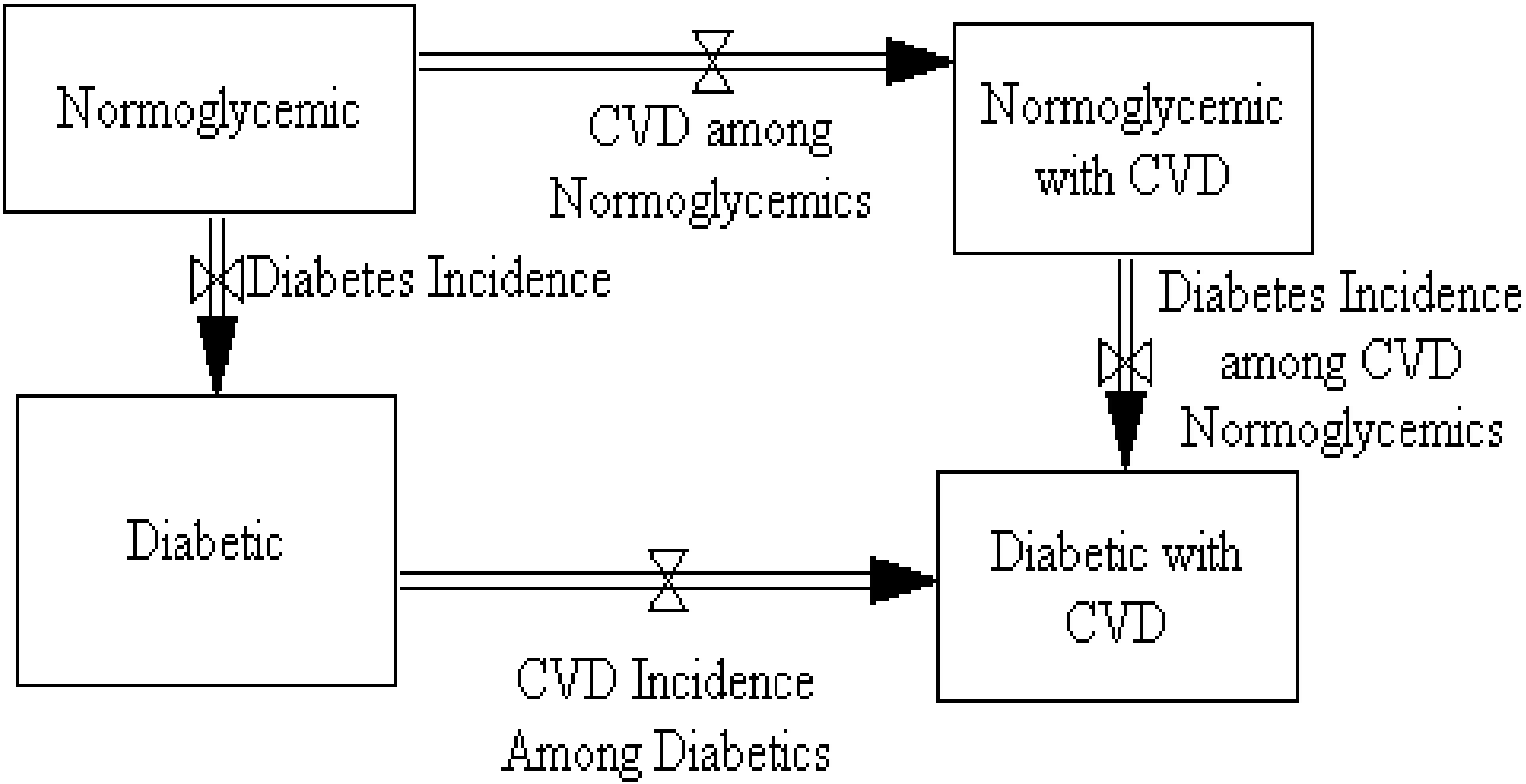
- Maintain a total count of people with each condition (the marginals)
- Maintain some prevalence information on occurrence of co-morbidities

- Problem: This doesn't capture the dynamics of co-morbidities
  - Prevalence will change in
    - Baseline
    - Induced by interventions
  - Because of differential mortality, intervention and other effects, anticipating how the prevalence of co-morbidities will change requires simulating them explicitly

Modularity Disparities: Aggregate Model

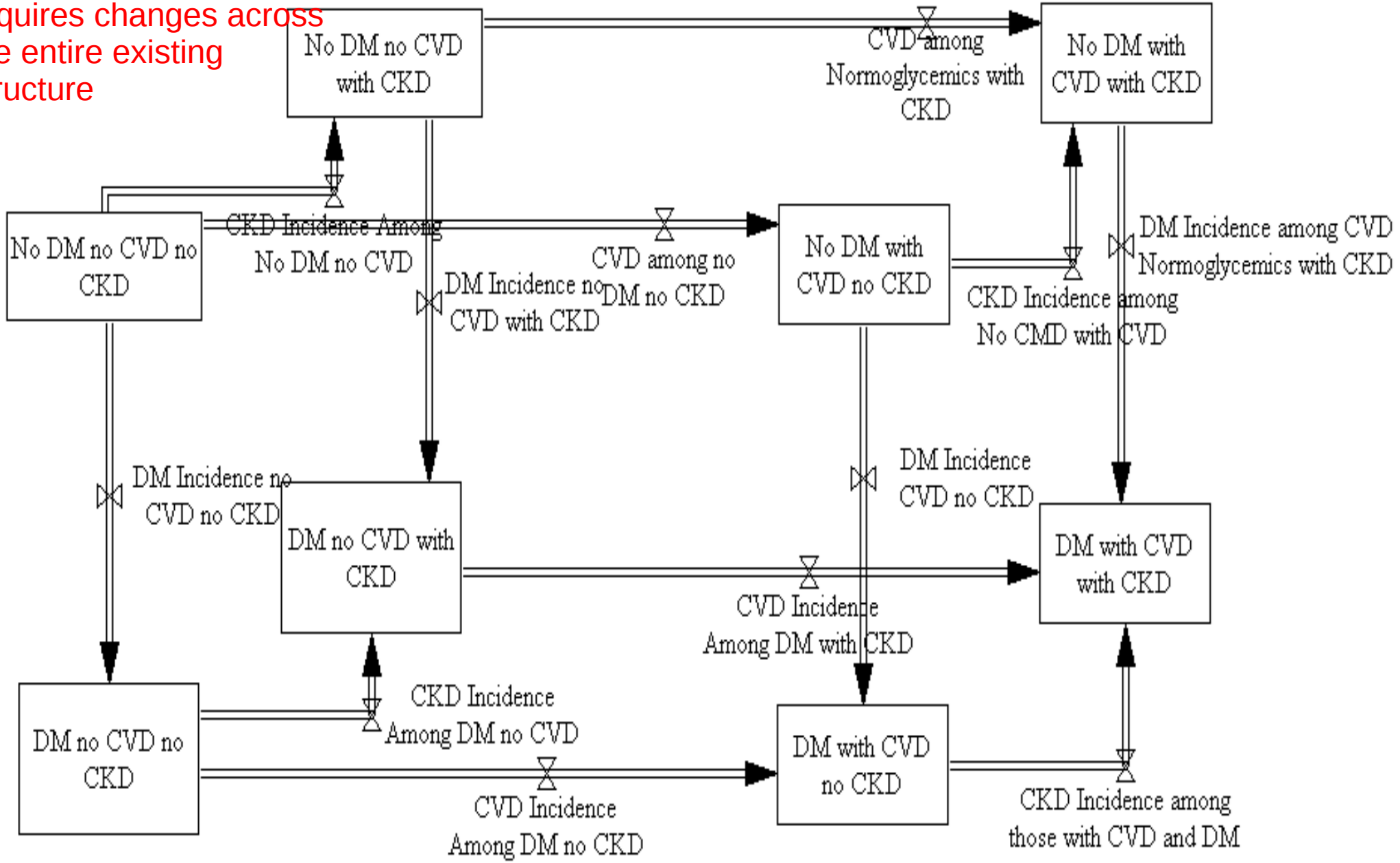


Modularity Disparities: Aggregate Model Adding a First Co-Morbid Condition

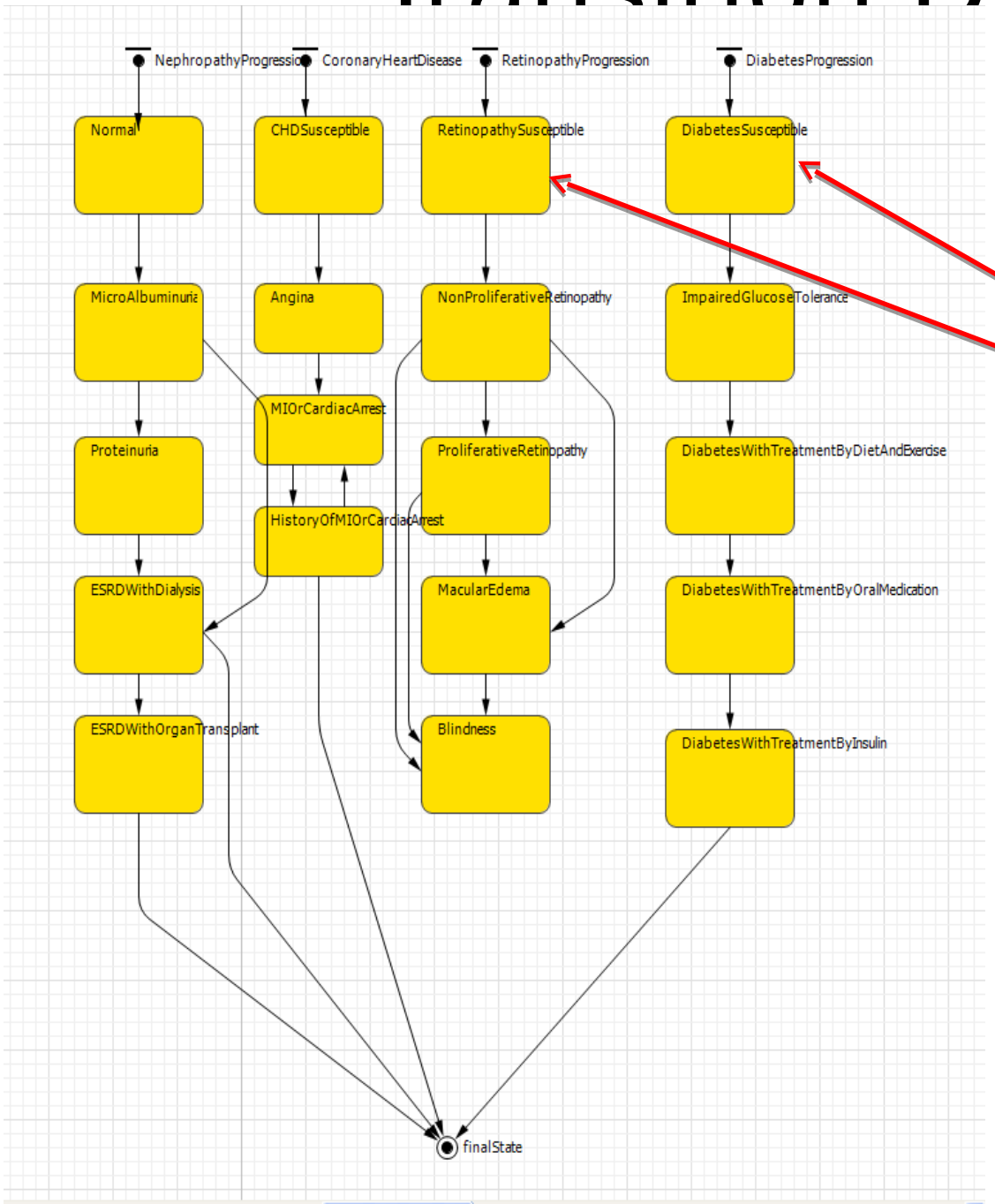


Modularity Disparities: Aggregate Model Adding a Second Co-Morbid Condition

Each new co-morbidity requires changes across the entire existing structure



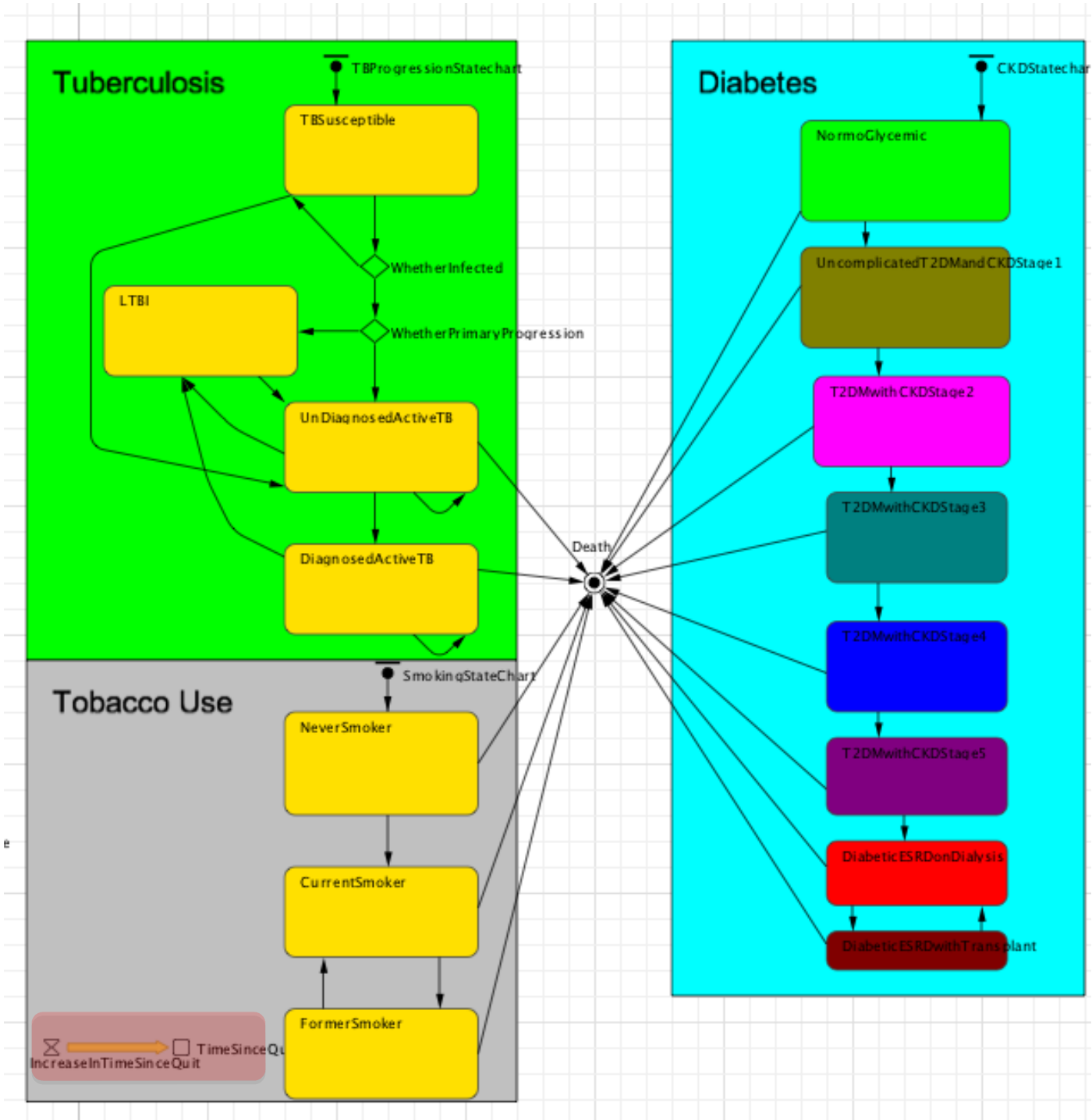
Individual Level: Parallel State Transition Diagrams



A person is in some particular state with respect to each of these (condition specific) state transition diagrams

This requires representing combinations of possibilities in an aggregate model

Parallel Transitions



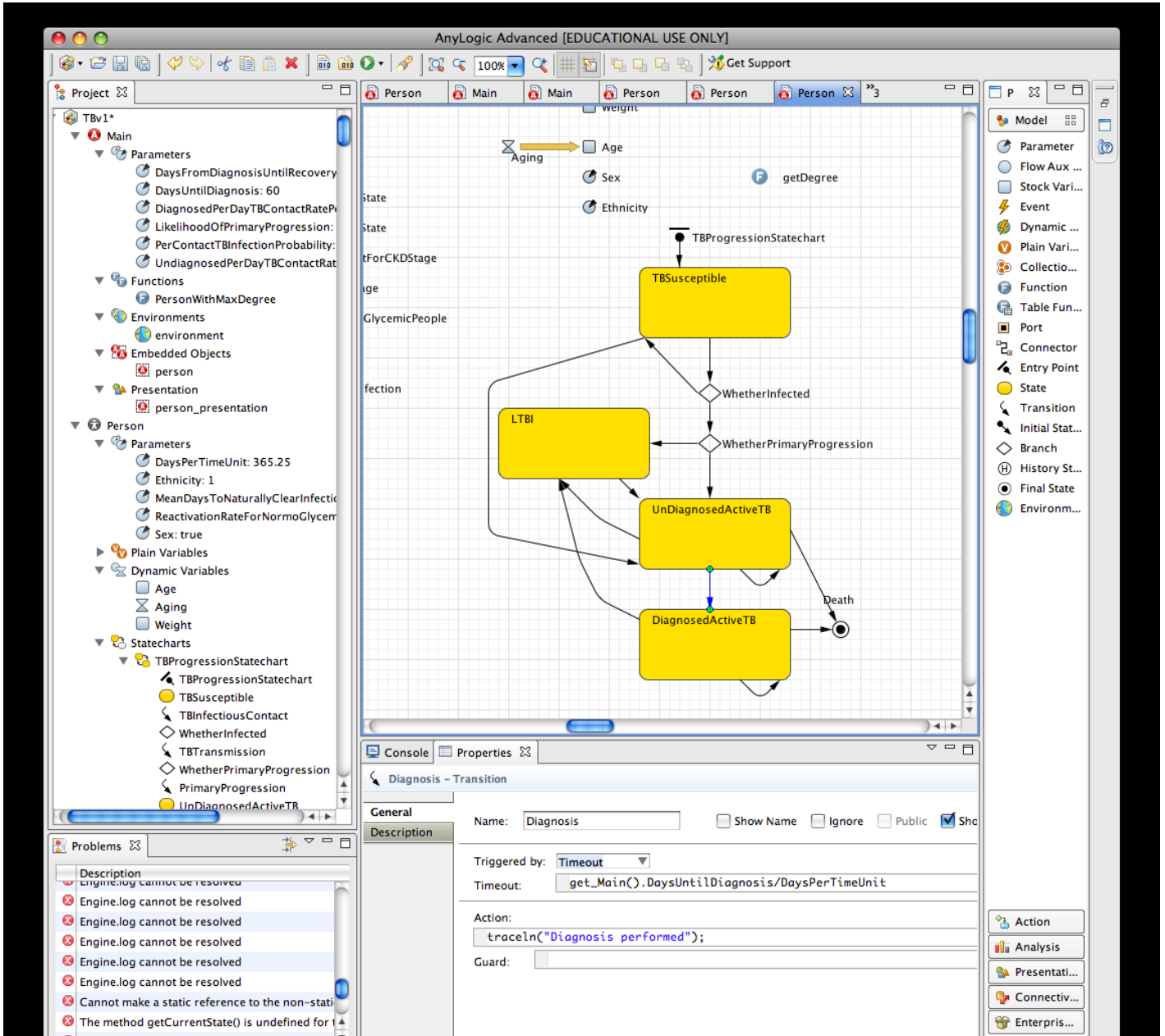
Discrete Agent Dynamics: Transitions

- 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
  - Arrival: Reach a location
- These transitions can be conditionally “routed” via branches
  - Conditions can determine to what destination state a particular transition will travel

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 Compartment models, these rates can change over time, but the statechart needs to be “made aware” of these changes (see later)
  - Leave & go back into current state (circular transition)
  - Trigger “change” event in Agent

Transition Type: Fixed Residence Time (Timeout)

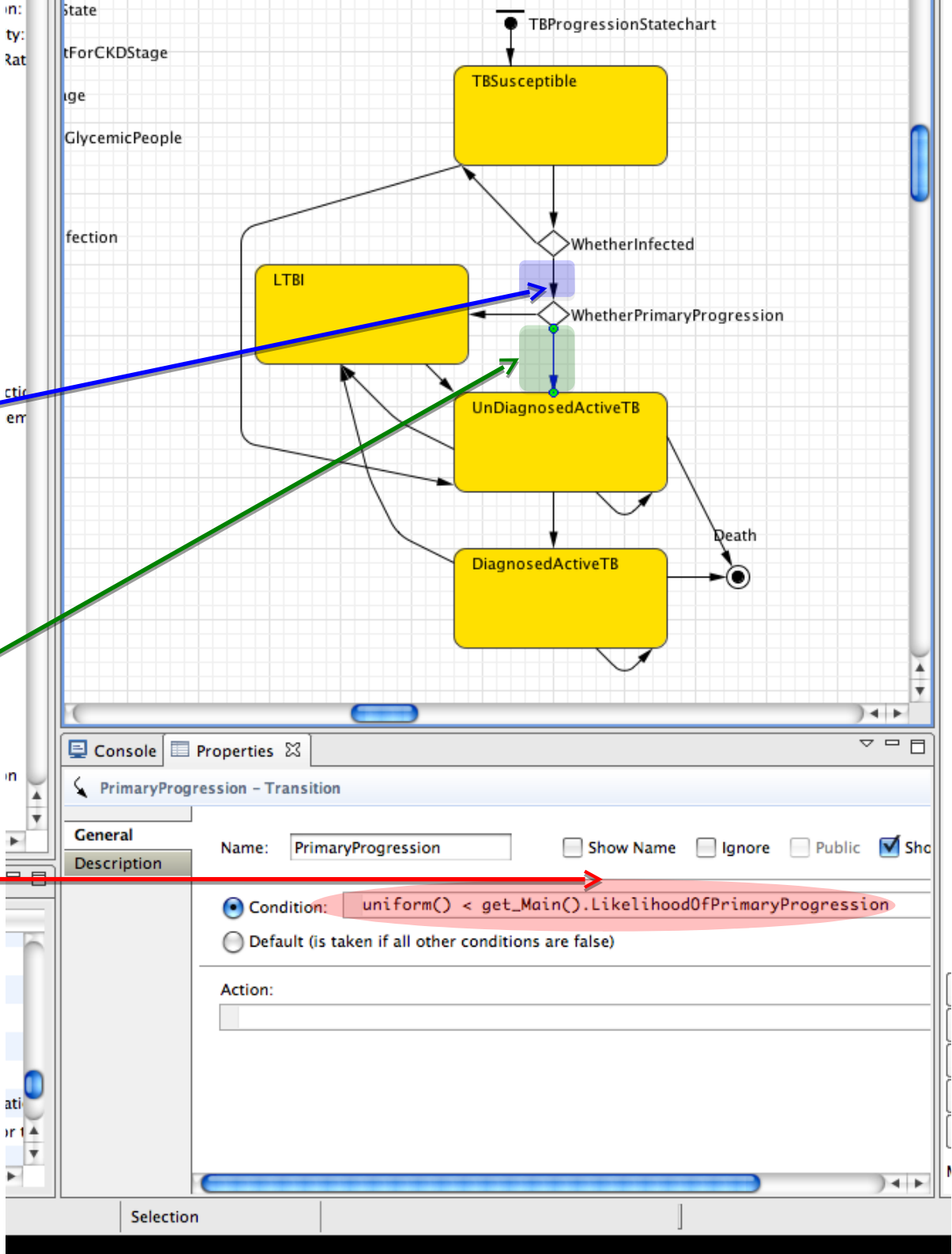


Example of Processes Associated with Fixed Timeouts

- Aging
- Tightly defined time constants associated with natural history
  - While these may be described as associated with a broad distribution (e.g. with a 1st or 2nd order delay), much of that variability may be due to heterogeneity
  - *For a given person, these may be quite specific in duration $\Rightarrow$ Can capture through a timeout*

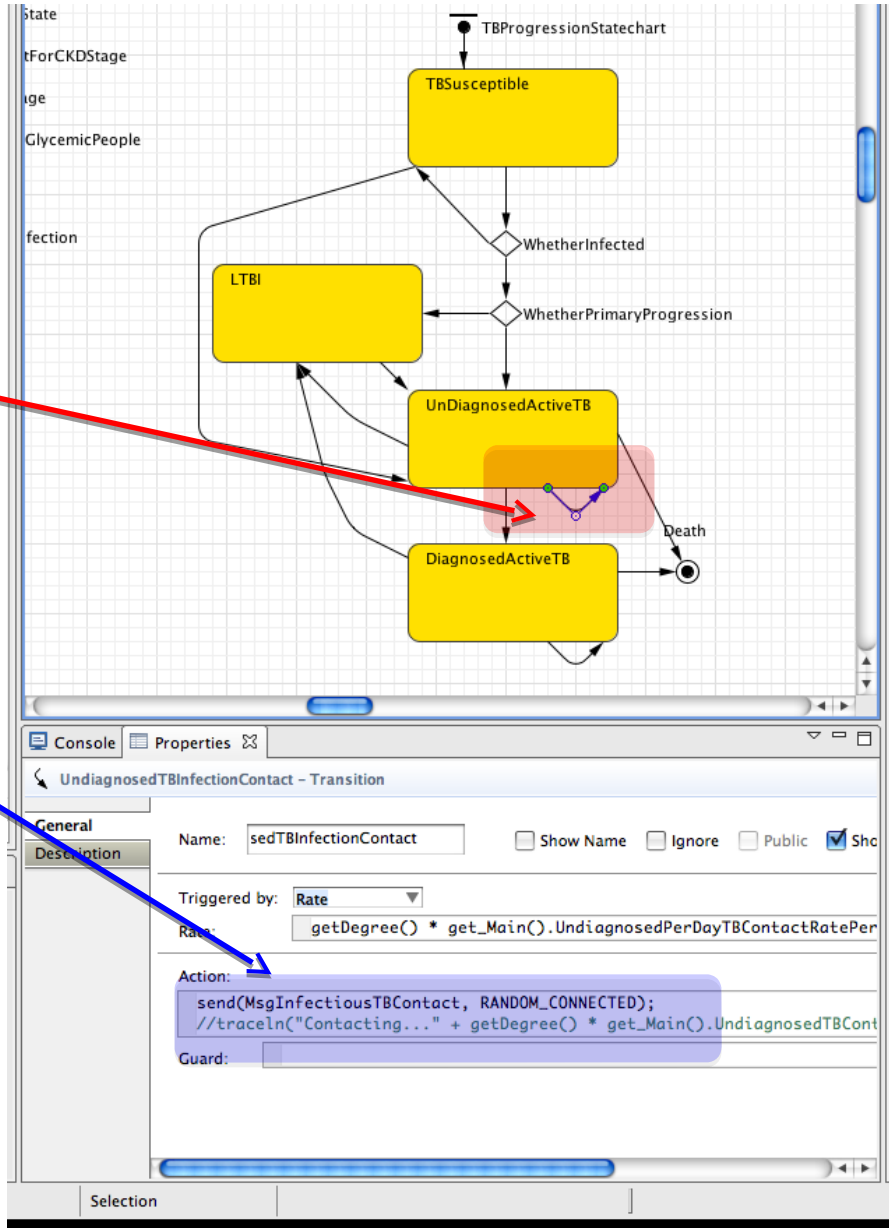
Example Conditional Transition

The incoming transition into “WhetherPrimaryProgression” will be routed to thisoutgoing transitionif this condition is true



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

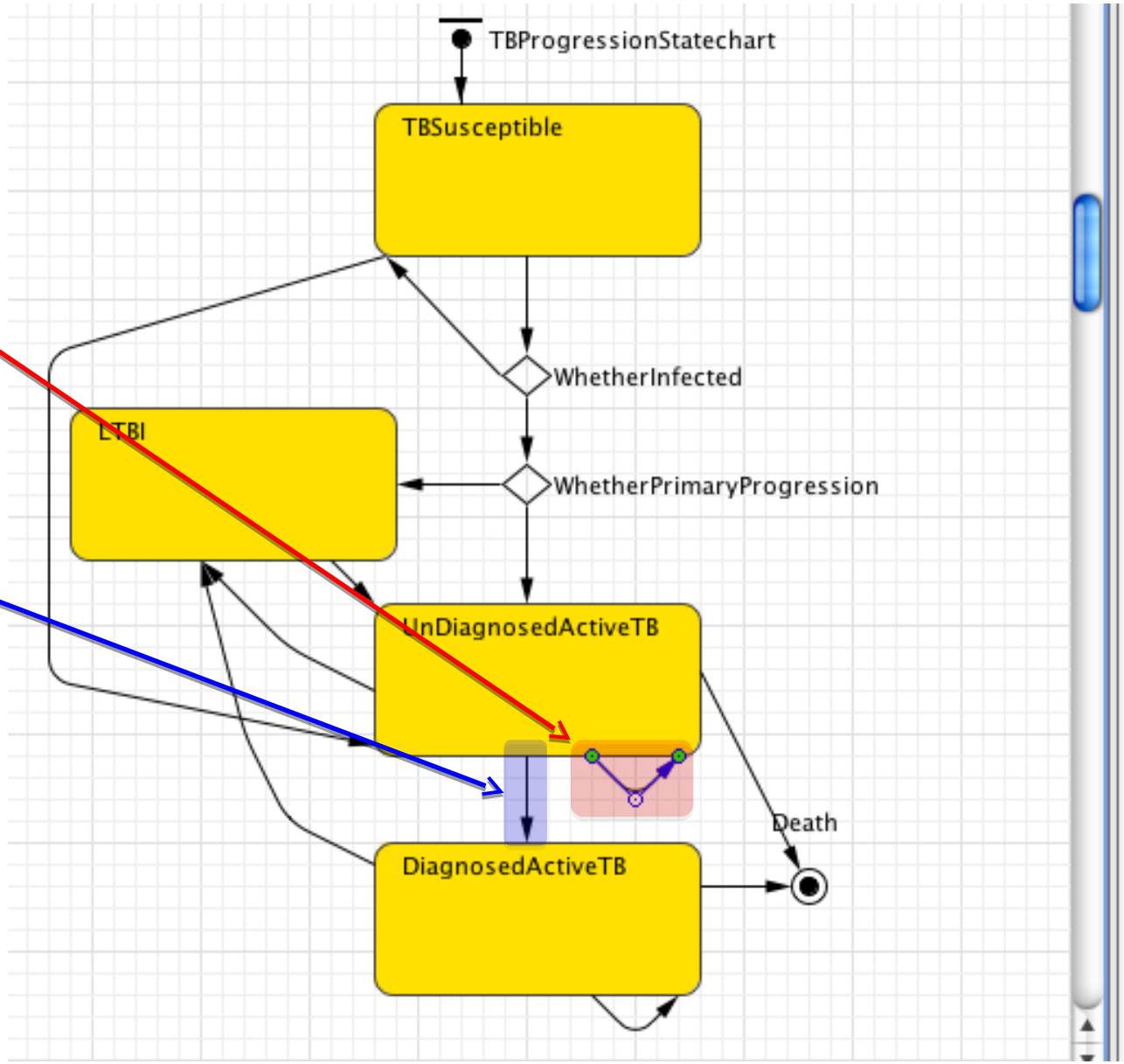
The self-transition will invoke this action when it occurs



Special Elements: Self-Transition

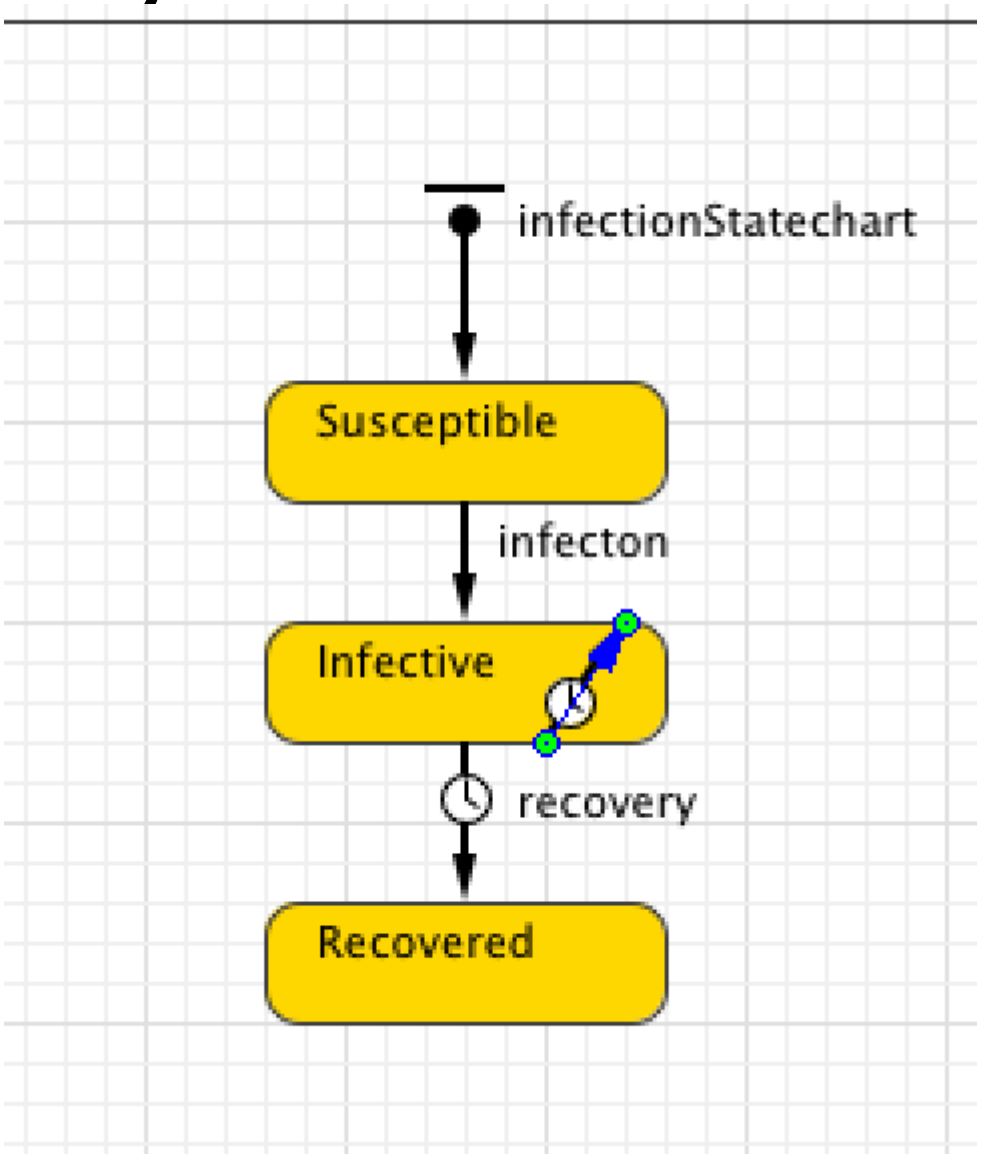
(Use if Wish To Have State Register Changing Out-transition rates)

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

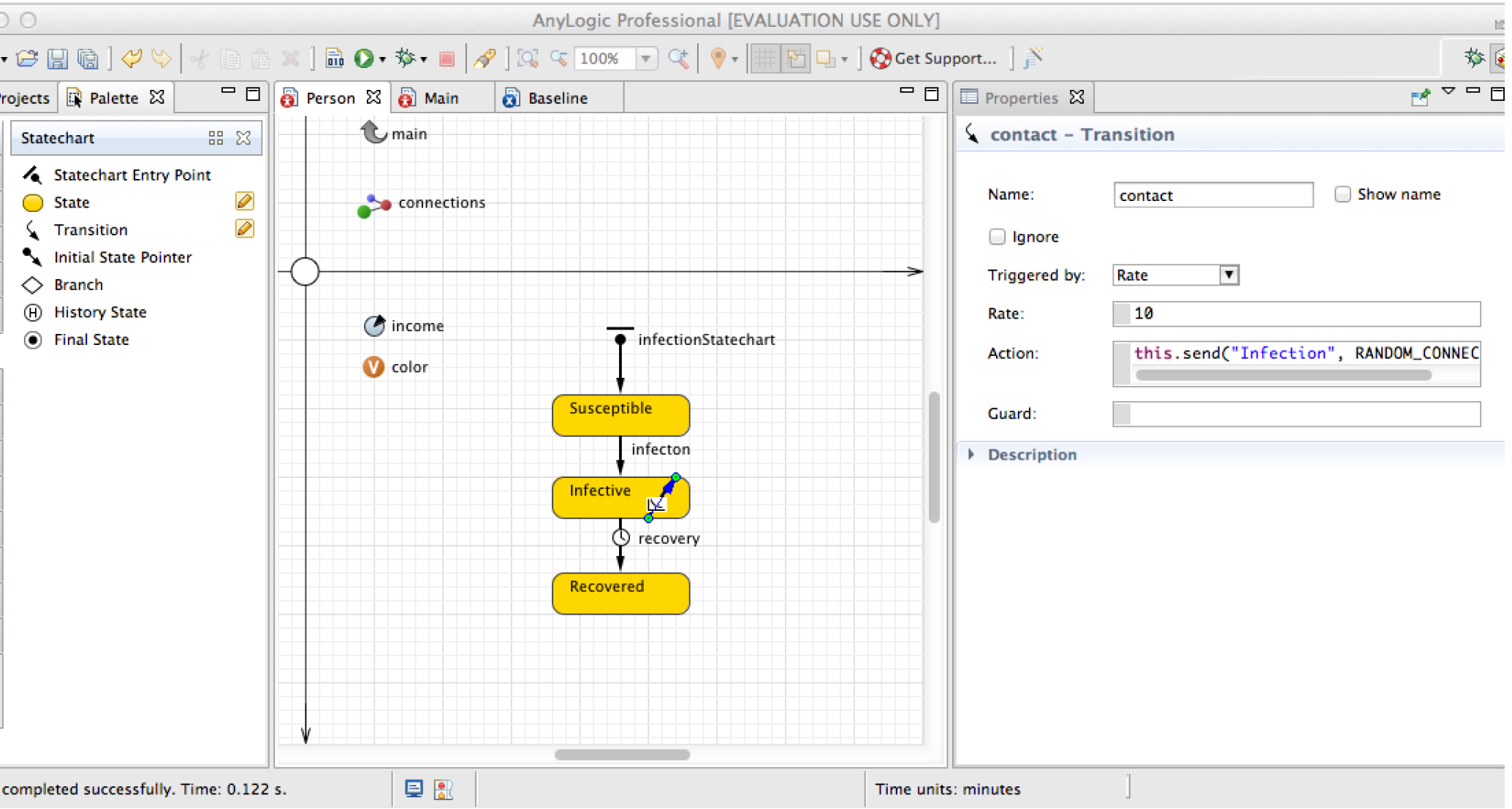


Hands-On Components: Discrete Inter-Agent Interaction

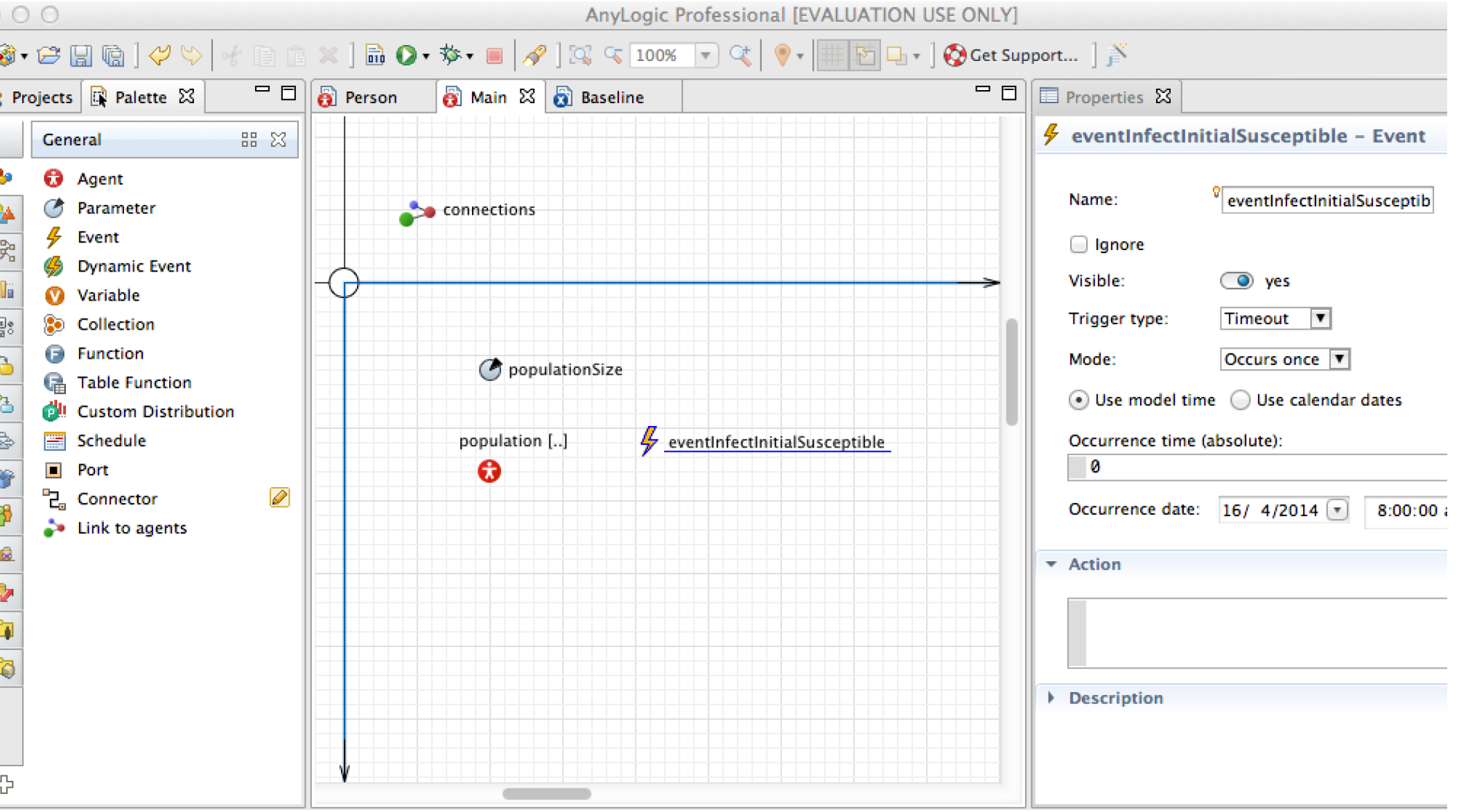
Adding a “Self Transition” to Periodically Undertake an Action



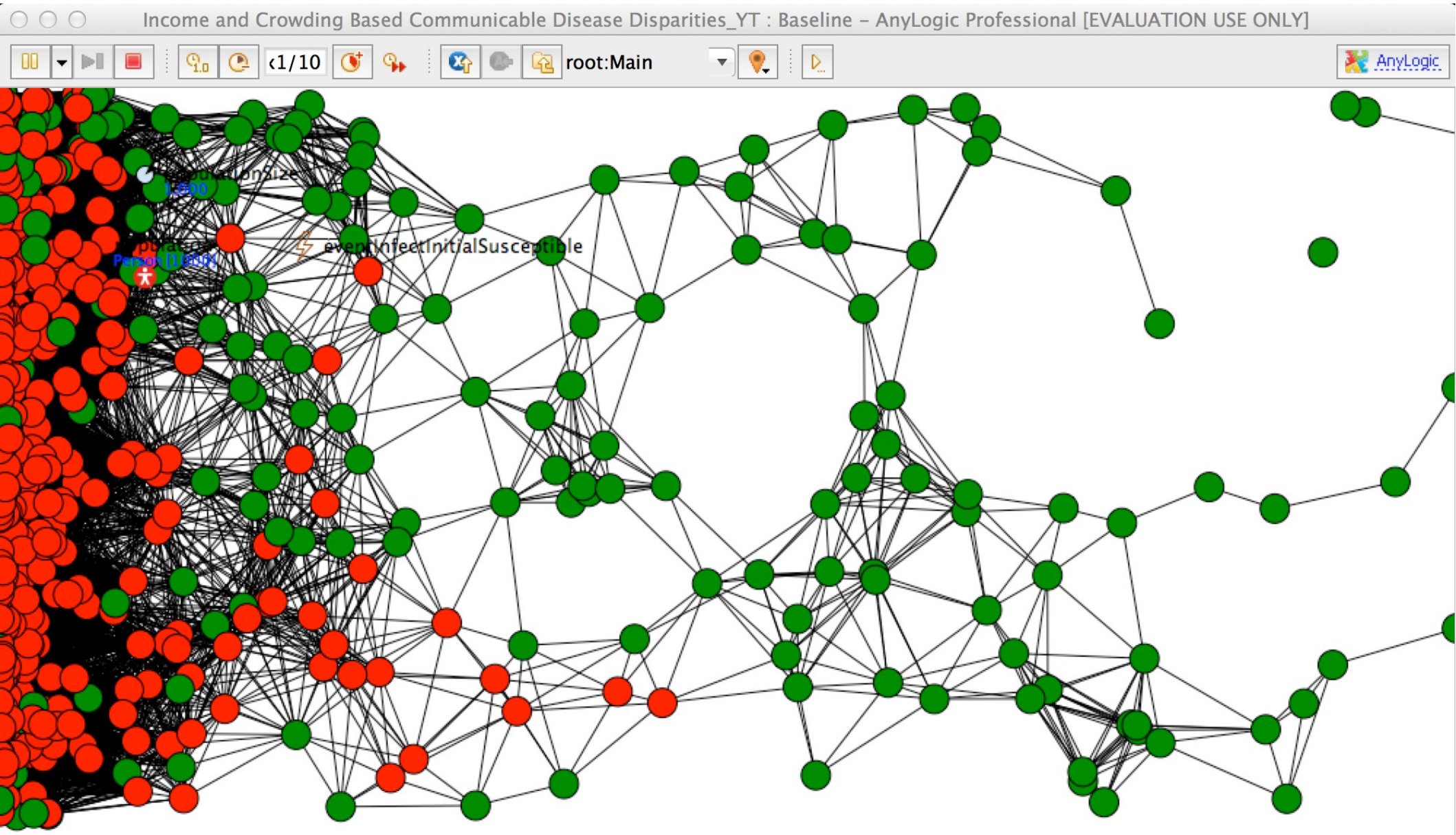
The Associated Code



Setting up an Event to Infect the Initial Person



Consequence



Discrete Agent Coupling via Messages

- Within AnyLogic, agents can be coupled by either discrete (instantaneous and individuated) or continuous (ongoing and gradual) coupling
- The preferred mechanism for discrete coupling is *messages* sent between agents
  - Many types of messages payloads are possible
  - An agent can send a given message to one or more agents
  - Frequently messages are sent locally to neighbors within the environment
    - Neighboring nodes on the network
    - Nearby neighbors in space

Messages & Statecharts

- Messages may be handled in many ways
- One of the most common ways in which messages are handled is by statecharts
  - A transition can be triggered (“guarded” or gated) by a message
  - A transition may be associated with an action that fires off a message to other agents (or to other statecharts within the agent)

Message Sending

- Messages may be sent to either
  - A particular, explicitly specified agent
  - An implicitly specified class of agents
    - Neighboring agents in the environment topology
    - Random agents
    - All agents
    - Any connected agents
    - All connected agents
- Mechanism:
  - send(*Message*, *DestinationObject*)
  - send(*Message*, *AgentClassId*)

Synchronous vs. Asynchronous Delivery

- Messages may be sent in two ways
  - Via ***send***: Asynchronous (scheduled)
    - Delivery occurs sometime after call to send
    - This is like sending a text message – it can be read later
  - Via ***deliver:*** Synchronous (immediately called)
    - Risks infinite loops in delivery (if destination also calls deliver in the reverse direction)
    - This is like calling the other person's phone – you demand their attention immediately

Message Payloads

- Sometimes just the fact that a message has been sent provides us with all of the information we need
- Sometimes just distinguishing different message types is sufficient
- We will sometimes send messages with payloads of data that provide extra information, e.g.
  - The agent that sent the message (eg that is infecting us)
  - Particular parameters
- Can send messages different payload types
  - Boolean/int/double/String/Other (can specify class type)

Hands-On Components: Events

Events & Scheduling in AnyLogic

- Reminder: In simulating stock & flow models, time advances in steps
  - Euler integration: Fixed-sized Steps
  - Runga-Kutta: Fixed or variable sized steps
  - For each timestep, we compute the flows & update the stocks
- AnyLogic jumps from “event” to “event”
  - The data structure that keeps track of such events is called the “schedule”
  - The associated process is called the “scheduler”

Implicit Events we've Seen

- Transitions
  - Fixed rate (Poisson arrival)
  - Timeout
  - Condition
  - Message transmission (schedules event for the receiver)
- Starting a model
- Stopping a model
- In this course, we term these *implicit events* because they are not reified as objects in the model
- To handle these events, code is inserted into certain handler areas for each of different sorts of classes

The Schedule

- At a given time, the schedule keeps track of a number of queued events
- Events may get added to the schedule (e.g. when we enter a new state)
- Events get deleted from the schedule
  - When they fire off and are complete
  - When another mutually exclusive event preempts them (e.g. a person dies before they recover from an infection)

History Information in Modeling

- Heterogeneity wrt individual history can be highly important for future health
  - Whether vaccinated
  - *in utero* exposure
  - Degree of glycemic control over the past decade
  - Exposure to adiposity
  - Previous exposure to a pathogen
- Such information can provide the basis for delivering interventions and treatments
- Inability to match such info can greatly undercut model value
- In some areas of health, we have access to longitudinal data that provides information on individual historical trajectories.

Example of Additional Information from Longitudinal Data

- Consider trying to distinguish pairs of situations
- e.g.: Smoking
  - Situation 1: One set of people quit & stay quit as former smokers, another set remain as current smokers
  - Situation 2: The entire set of people cycle through situations where they quit, relapse & repeat
- These two situations have very different health consequences
- We'd probably choose vary different sets of interventions for these two situations
- Similar examples are easy to imagine for obesity, STIs, TB, glycemic control & diabetes, etc.

Capturing History Information

- Individual based model
  - Both discrete & continuous history information can be readily captured
    - Categorical/discrete: State (in statechart) or variable
    - Continuous: Variable
  - Readily able to capture records of trajectories
- Aggregate model
  - Categorical/discrete: Limited discrete history information can be captured by disaggregating stocks
    - Curse of dimensionality provides tight limits on # of aspects of history can be recorded
  - Continuous: Almost always infeasible
  - Very complex to provide distributions of trajectories (via convolution of potentially changing PSFs of stocks)

Longitudinal Fidelity: Aggregate Models

- An aggregate model provides an ongoing series of *cross-sectional* descriptions of system state
  - In Calibration & validation, we can do rich comparison of these cross-sectional descriptions against available point or time-series data
  - Because the model does not track individuals, we generally cannot explicitly extract model longitudinal trajectories from the model for comparison with empirical giving longitudinal trajectories

Longitudinal Fidelity: Individual-Based Models

- An individual-based model provides easily accessible *cross-sectional* and longitudinal descrip. of system state
  - The system state at a particular moment in time is cross-sectional
  - By following & recording the trajectories of particular individuals, we can obtain longitudinal description
- In Calibration & validation, we can do rich comparison of both longitudinal and cross-sectional descriptions against available point or time-series data
  - It is in principle possible to have a model that accords with cross-sectional data, but which is at odds longitudinally

Stochastic Processes in AnyLogic

- In AnyLogic, ABM and Discrete Event Models ("Network-Based Modeling") are typically stochastic
  - Transitions between states
  - Event firing
  - Messages
    - (Frequent) timing of message send
    - Target of messages
  - Duration of a procedure
- As a result, there will be variation in the results from simulation to simulation

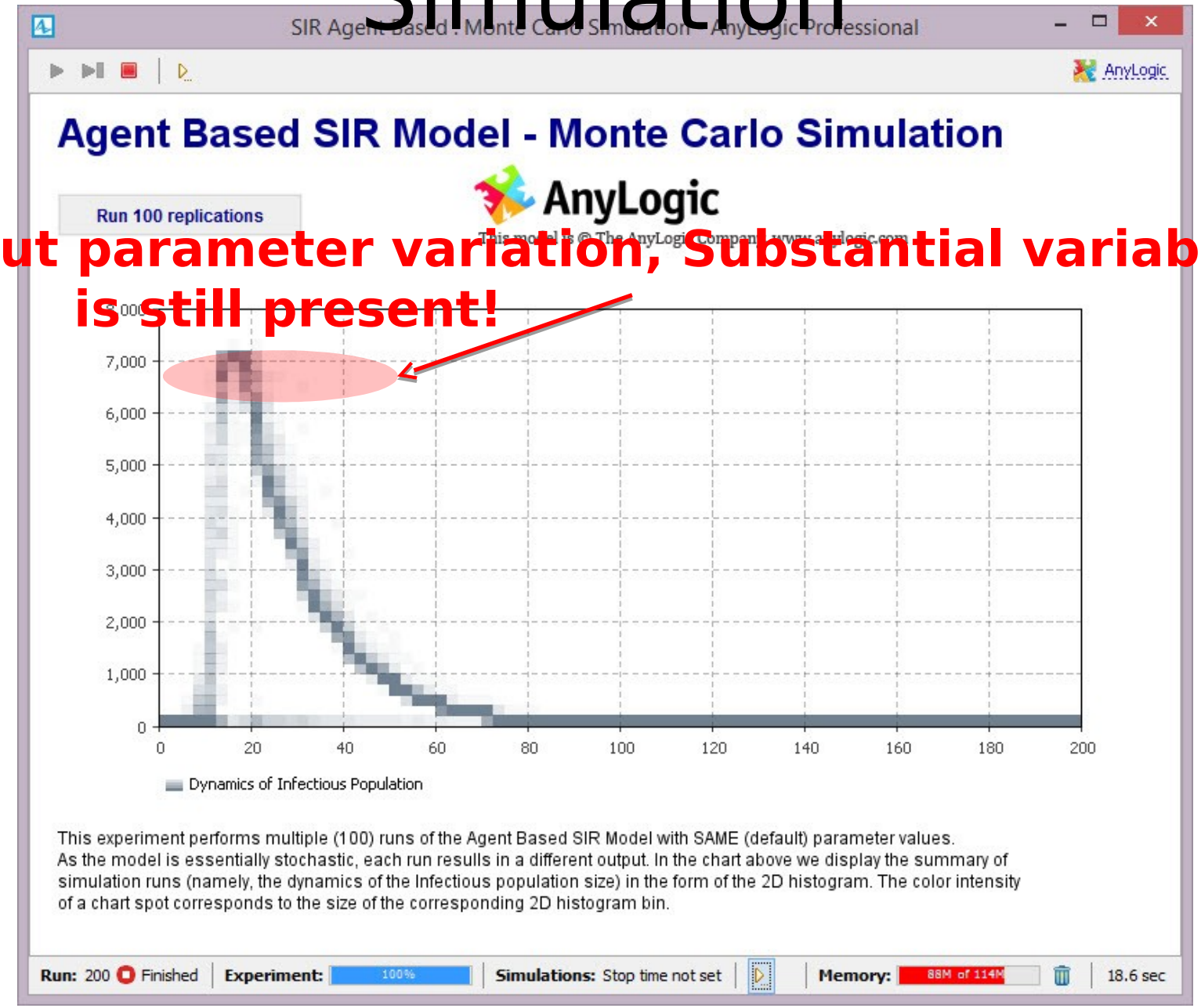
Summarizing Variability

- To gain confidence in model results, typically need to run a “Monte Carlo” ensemble of realizations
  - Deal with means, standard deviations, and empirical fractiles
  - As is seen here, there are typically still broad regularities between most runs (e.g. rise & fall)
- Need to reason over a population of realizations $\Rightarrow$ statistics are very valuable
  - Fractile within which historic value falls
  - Mean difference in results between interventions

Monte Carlo Methods in AnyLogic

- Monte Carlo methods draw repeated samples from distributions & stochastic processes of interest
- When running Monte Carlo method, we'd like to summarize the results of multiple runs
- One option would be to display each trajectory over time; downside: quickly gets messy
- AnyLogic's solution
  - Accumulate data regarding how many trajectories fall within given areas of value for a given interval of time using a "Histogram2D Data"
  - Display the Histogram2D Chart

Results of Monte Carlo Simulation



Agent-Based Modeling Workshop:

Themes for Weaving into the Interactive Example

Nathaniel Osgood

Using Modeling to Prepare for Changing Healthcare Needs

April 16, 2014