Expressing Discrete Inter-Agent Dynamics via Messaging

Nathaniel Osgood

10-25-2009

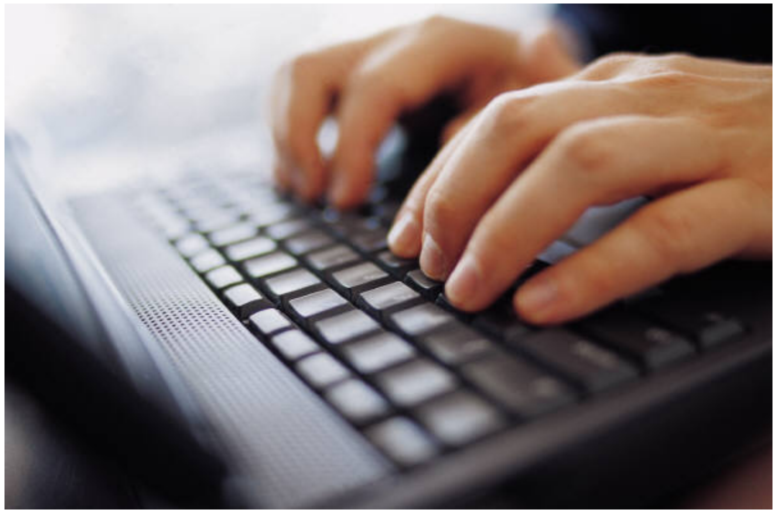
Discrete Agent Coupling via Messages

- 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)

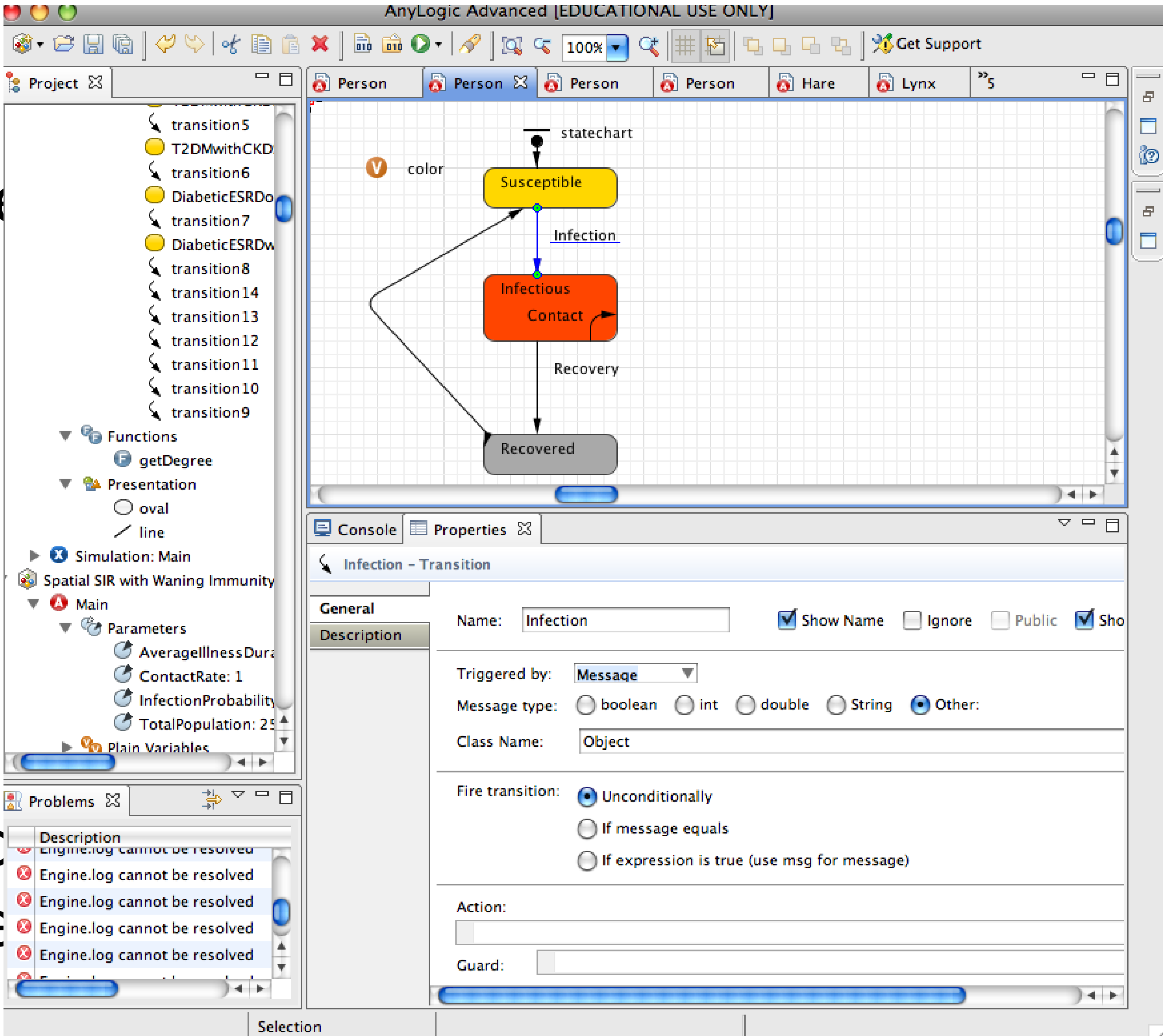
Hands on Model Use Ahead



Load model:Spatial SIR with Waning Immunity.alp

Receiving a Message

- In this case, only 1 messag type exists, so the simple fact that a message has been received is sufficient; there is no need to inspec message conte


AnyLogic Advanced [EDUCATIONAL USE ONLY]

Project: transition5, T2DMwithCKD, transition6, DiabeticESRDo, transition7, DiabeticESRDw, transition8, transition14, transition13, transition12, transition11, transition10, transition9, Functions, getDegree, Presentation, oval, line, Simulation: Main, Spatial SIR with Waning Immunity, Main, Parameters, AverageIllnessDura, ContactRate: 1, InfectionProbability, TotalPopulation: 25, Plain Variables

Problems

Description
Engine.log cannot be resolved
Engine.log cannot be resolved
Engine.log cannot be resolved
Engine.log cannot be resolved
Engine.log cannot be resolved

Infection – Transition

Name: Infection [✓ Show Name] Ignore Public

Triggered by: Message

Message type: boolean int double String [✓ Other:]

Class Name: Object

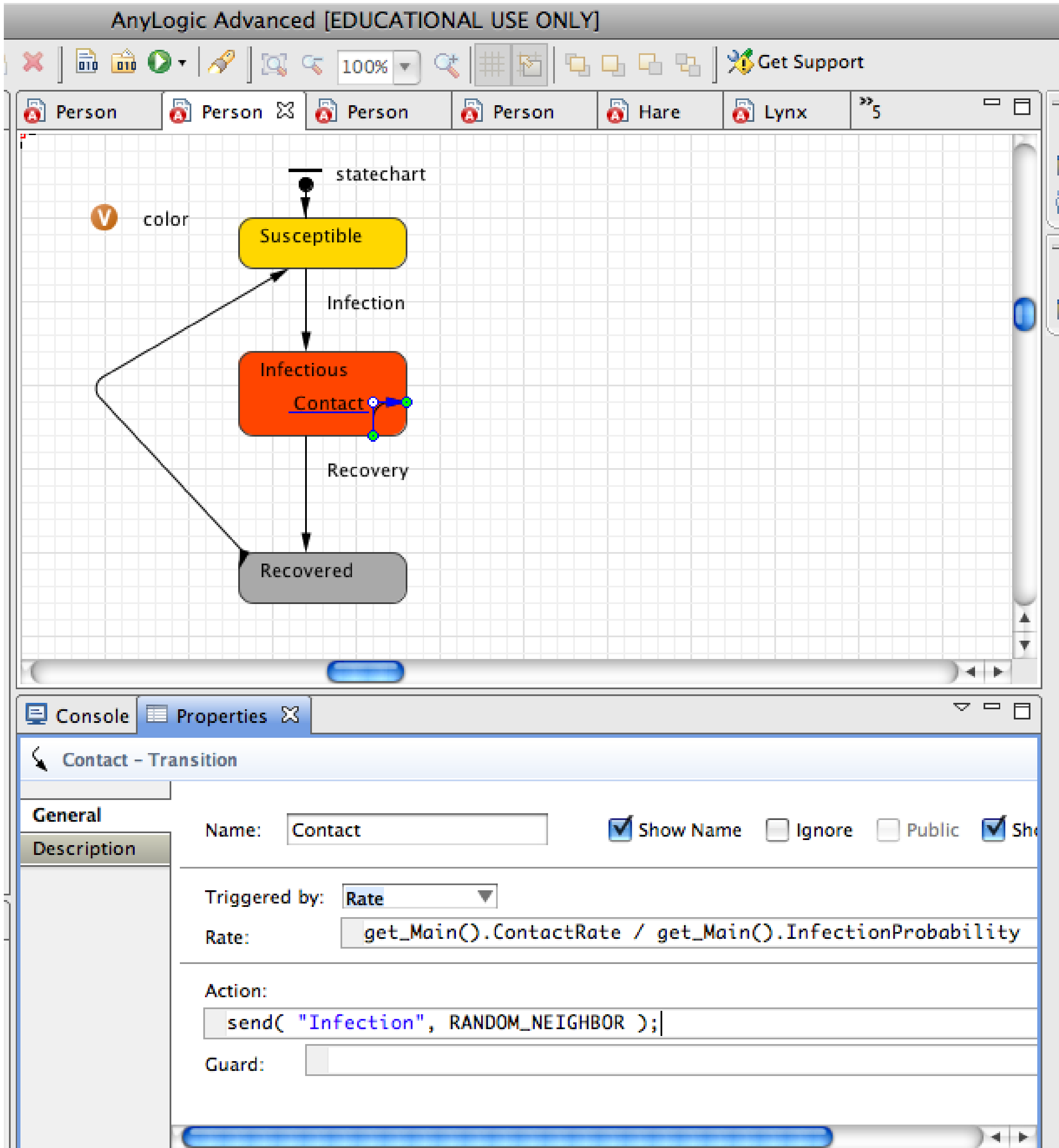
Fire transition: [✓ Unconditionally] If message equals If expression is true (use msg for message)

Action:

Guard:

Sending a Message

- (Self-transition because remains in state)



Name: Contact ☑ Show Name ☐ Ignore ☐ Public ☑ Sh

Triggered by: Rate

Rate: get_Main().ContactRate / get_Main().InfectionProbability

Action:

```
send( "Infection", RANDOM_NEIGHBOR );
```

Guard:

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*)

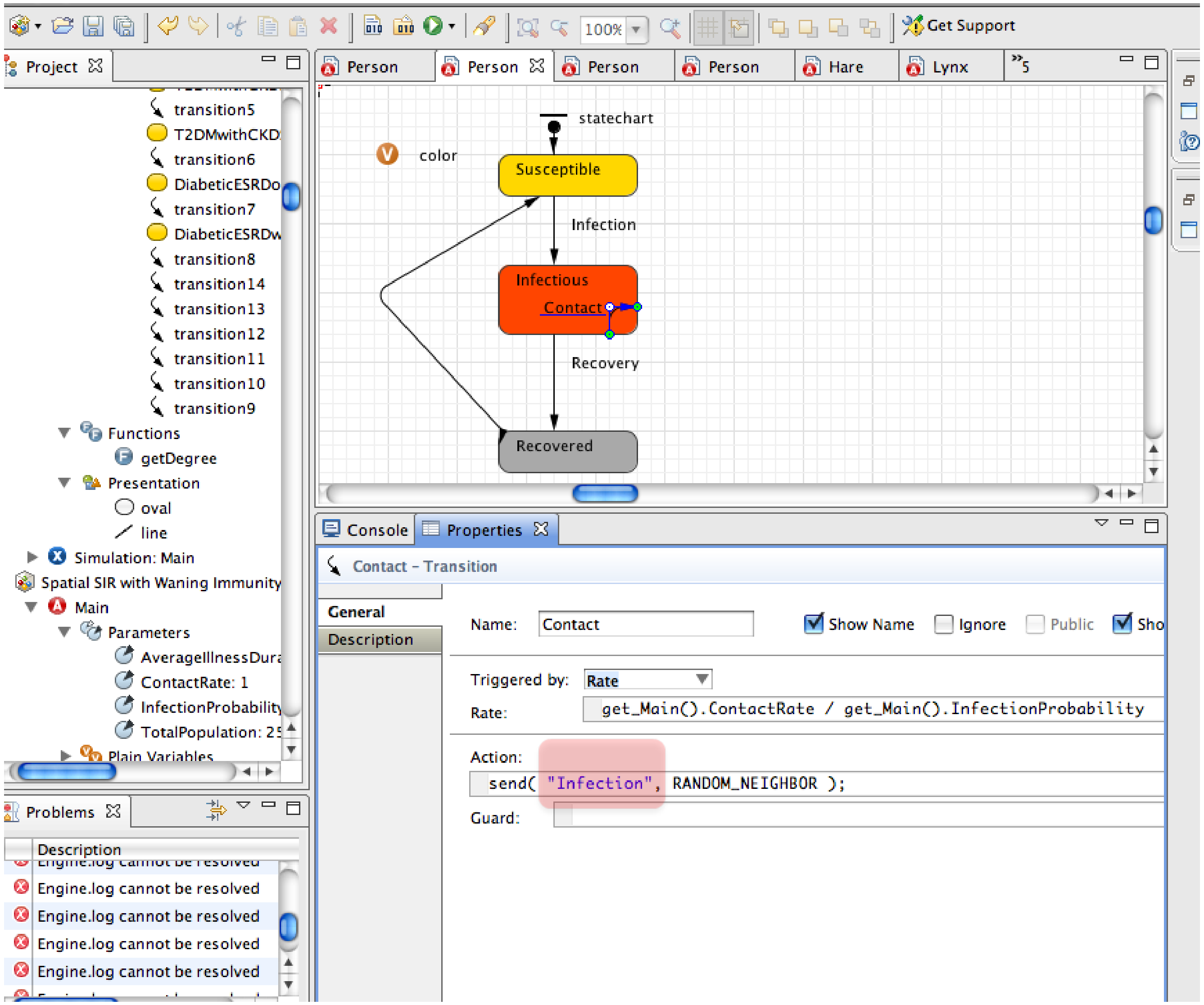
Synchronous vs. Asynchronous Delivery

- Messages may be sent in two ways
  - Via ***send***: Asynchronous
    - Delivery occurs sometime after call to send
  - Via ***deliver***
    - Synchronous
    - Risks infinite loops in delivery (if destination also calls deliver in the reverse direction)

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 multiple message types
  - Boolean/int/double/String/Other (can specify class type)

Sending a Message with a String Payload



Description
Engine.log cannot be resolved
Engine.log cannot be resolved
Engine.log cannot be resolved
Engine.log cannot be resolved
Engine.log cannot be resolved

Contact – Transition

General
Description

Name: Contact [✓] Show Name [ ] Ignore [ ] Public [✓] Sho

Triggered by: Rate

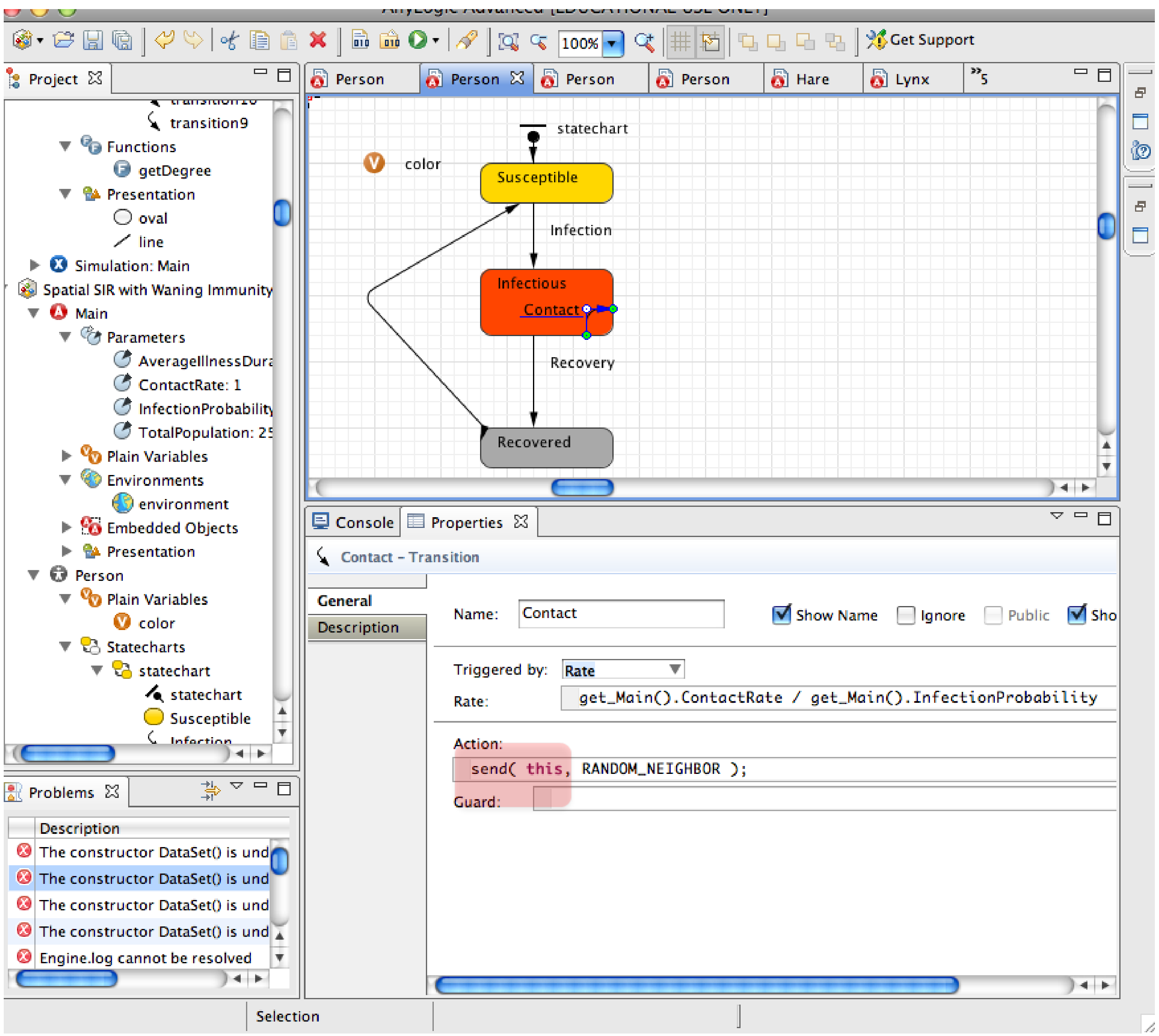
Rate: `get_Main().ContactRate / get_Main().InfectionProbability`

Action:

```
send( "Infection", RANDOM_NEIGHBOR );
```

Guard:

Sending a Message with Object Payload



Contact – Transition

Name: Contact ✓ Show Name ☐ Ignore ☐ Public ✓ Sho

Triggered by: Rate

Rate: get_Main().ContactRate / get_Main().InfectionProbability

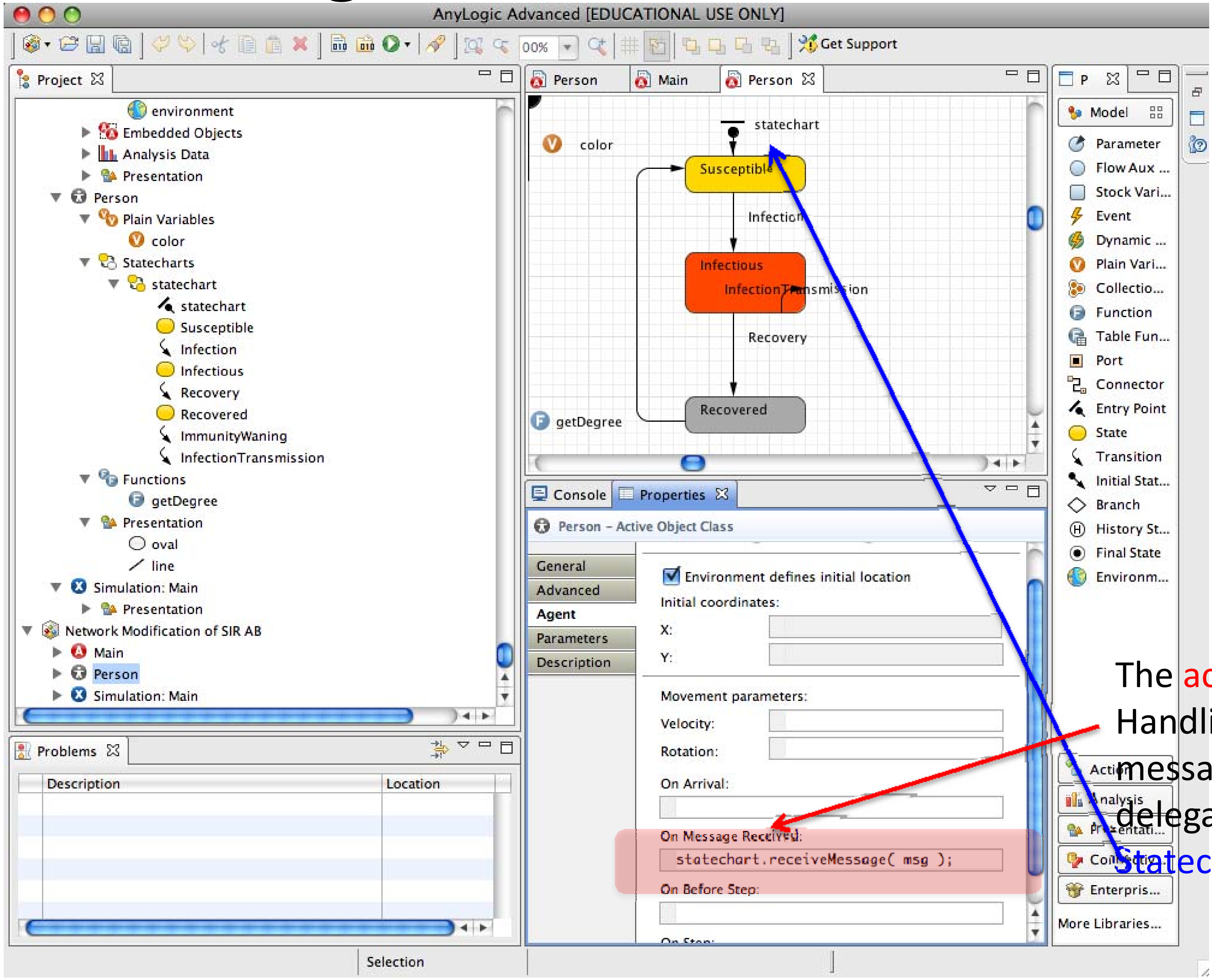
Action:

```
send( this, RANDOM_NEIGHBOR );
```

Guard:

Description
The constructor DataSet() is und
The constructor DataSet() is und
The constructor DataSet() is und
The constructor DataSet() is und
Engine.log cannot be resolved

Receiving a Message: Forwarding Messages on to the Statechart



Person – Active Object Class

☑ Environment defines initial location

Initial coordinates:

X:

Y:

Movement parameters:

Velocity:

Rotation:

On Arrival:

On Message Received:

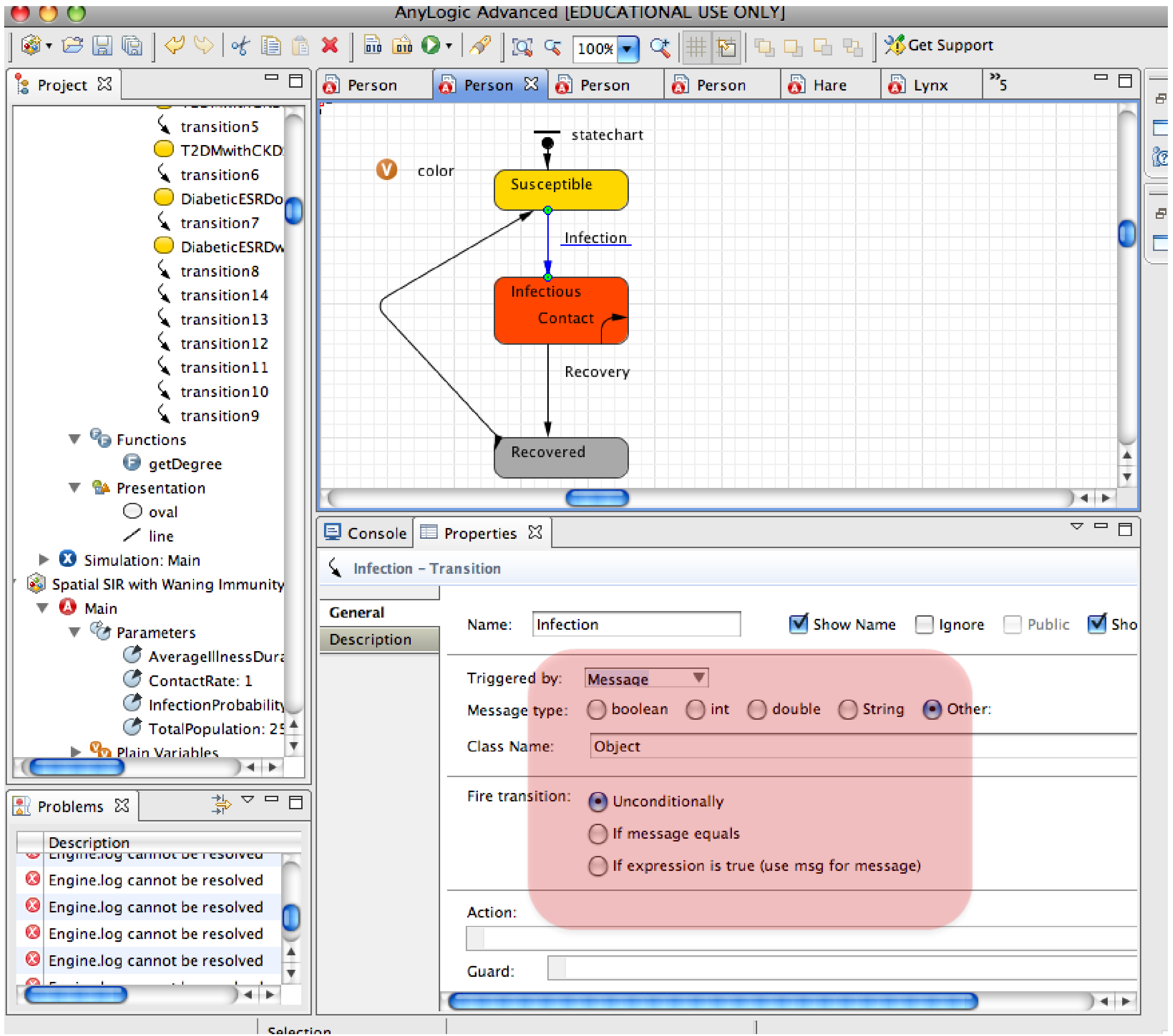
```
statechart.receiveMessage( msg );
```

On Before Step:

On Step:

The action for Handling received messages delegates to the Statechart object

Receiving a Message



Infection – Transition

General
Description

Name: Infection [✓ Show Name] Ignore Public [✓ Sho

Triggered by: Message

Message type: boolean int double String [✓ Other:]

Class Name: Object

Fire transition: [✓ Unconditionally] If message equals If expression is true (use msg for message)

Action:

Guard: