Assignment 1: Tree & Shade

The goal of this assignment is to create a qualitative model that describes the growth of a tree in time and the effects this growth has on the shaded area that this tree produces. The model is kept a simple as possible, as the aims are to be familiarised with the model ingredients from which a qualitative model is built, and to learn to use the qualitative reasoning and modelling software DynaLearn. The following model ingredient types are covered in this assignment:

- Entity
- Quantity (with value: magnitude and, derivative)
- Quantity space
- Scenario
- Model fragment types: Static & Process
- Influence (= direct influence)
- Proportionality (= indirect influence)
- Correspondence

Consider a growing tree that casts a shadow on the ground. Assume that the tree always grows (ignoring the need for water, sunlight, air and minerals). The size of the shadow depends on the size of the tree. This is a dynamic feature: the shaded area increases when the tree becomes bigger. Indirectly, the growth of the tree causes the shaded area to increase.

Step 1: Entity Hierarchy

The model implementation starts with the definition of the key objects within the system. In this case this is the tree. In order to keep the model simple, we ignore other possible objects in the system, like the sun, the forest the tree is located in, or the amount of minerals in the soil.

Do the following:

- Add an entity *Tree* to the entity hierarchy

Step 2: Quantities and Quantity Spaces

Quantities describe the changeable features of entities. The possible values a quantity can have are described in a quantity space. The *Tree* in our system, which grows and produces shade, has three important quantities, namely:

- the size of the *Tree* (*Size*);
- the area of shade caused by the *Tree* (*Shade*); and,
- the rate at which the *Tree* grows (*Growth rate*).

Each of the above quantities has to have an associated quantity space. For the *Size* of the *Tree*, three qualitative values are possible: *Small*, *Medium* and *Large*. Since the *Shade* depends on the *Size* of the *Tree*, it is logical to choose the same quantity space for that quantity. Finally, there is either no growth, or some positive amount of growth. Therefore, the possible qualitative values for the *Growth rate* are: *Zero* and *Plus*. The quantities and their quantity spaces are summarised in Table 1.

Table 1: The quantities in the Tree & Shade model and their quantity spaces

Quantity	**Quantity Space**
Size	{Small, Medium, Large}
Shade	{Small, Medium, Large}
Growth rate	{Zero, Plus}

Do the following:

- Define a quantity space called 'sml' with the values {Small, Medium, Large} (*Small* and *Large* should be intervals, and *Medium* a point).
- Define the quantity *Size* and associate it with the quantity space 'sml'.
- Define the quantity *Shade* and associate it with the quantity space 'sml'.
- Define a quantity space called 'zp' with the values {Zero, Plus} (*Zero* should be a point).
- Define the quantity *Growth rate* and associate it with the quantity space 'zp'.

Step 3: Creating a Scenario

A scenario is the starting point for a simulation. It is a description of a (typical) situation to which the knowledge captured in the model will be applied when simulating. A model usually has multiple scenarios that can be used to run simulations. But in this assignment we consider only one scenario, namely: ‘a tree with a small size’.

Do the following:

- Define a new scenario: 'A small growing tree'.
- Open the scenario in the scenario editor.
- Add the entity *Tree* to the scenario.
- Add the quantity *Size* attached to the entity *Tree* to the scenario.
- Set the value of the *Size* quantity to *Small*.
- Add the quantity *Shade* attached to the entity *Tree* to the scenario.
- Set the value of the *Shade* quantity to *Small*.

Step 4: Static Model Fragment with Knowledge about Size and Shade

In this step a model fragment will be created which represents the relationship between *Size* and *Shade*. Changes in the shaded area (*Shade*) are the consequence of changes in the *Size* of the *Tree*. In this case, when *Size* increases the *Shade* also increases, and when *Size* decreases the *Shade* also decreases. This relation can be modelled using a positive proportionality.

Do the following:

- Define a new static model fragment: ‘Tree with shade’.
- Open the model fragment in the model fragment editor.
- Add the entity *Tree* to the model fragment as a condition.
- Add the quantity *Size* attached to the entity *Tree* to the model fragment as a consequence.

- Add the quantity *Shade* attached to the entity *Tree* to the model fragment as a consequence.
- Add a positive proportionality from *Size* to *Shade* to the model fragment.

Step 5: Process Model Fragment with Knowledge about Tree Growth

In this step the relationship between the *Growth rate* and the *Size* will be modelled. The *Growth rate* is assumed to be positive and stable and constantly causing the *Size* of the T*ree* to increase. This relation is represented using a positive influence between *Growth rate* and *Size*.

Do the following:

- Define a new process model fragment: ‘Growth of tree’.
- Open the model fragment in the model fragment editor.
- Add the entity *Tree* as a condition.
- Add the quantity *Size* as a consequence associated to the *Tree.*
- Add the quantity *Growth rate* as a consequence.
- Set the value (as a consequence) of the *Growth rate* to *Plus* (to indicate that a small, medium or large tree always grows).
- Set the value of the derivative of the *Growth rate* to stable (to indicate that the growth does not change).
- Create a positive influence from the *Growth rate* to the *Size.*

Step 6: Running and Inspecting the Model

When running the model, the goal is to have the simulator predict (simulate) the changes of the *Size* and the *Shade* of the Tree as time passes. Ideally, the resulting simulation consists of 3 states: a first state in which both *Size* and *Shade* are *Small*, a second state in which both *Size* and *Shade* are *Medium*, and a third state in which both *Size* and *Shade* are *Large*.

Do the following:

- Start the simulation of the model using the previously created scenario: 'A small growing tree'.
- Run a full simulation.
- Select a path from the start state to the end state (mind the order in which the states are selected).
- Look at the value history view.
- Do the previous two steps for each of the possible paths (if multiple paths exist).

Step 7: Debugging Suggestions

Debugging questions:

- Are the values in the first state in accordance with what has been defined in the initial scenario?
- Does the *Growth rate* have a value?
- Do both the *Size* and the *Shade* have a value?
- Do *Size* and the *Shade* have the same qualitative value?

- Is the number of states similar to the expected amount mentioned in Step 6?
- Do all quantities have their expected values in all states?

Suggestions:

- If *Shade* does get values in each of the states, but if those values are different compared to the values generated for *Size*, maybe a correspondence should be defined between the two quantities in the model fragment 'Tree with shade'.
- If *Size* does not increase in the first state, check whether the influence between the *Growth rate* and the *Size* is defined appropriately. Also check whether the value assigned to the *Growth rate* is actually positive (that is: *Growth rate* >0).
- If *Shade* does not increase in the first state, check whether the proportionality between *Size* and *Shade* is defined appropriately. Also check whether *Size* is actually increasing, because it should before *Shade* can increase. Also check whether *Shade* has a value, because it should have a value defined in the scenario before it can increase.

Do the following:

- Discuss which of the debugging options is needed to improve the model.

Step 8: Fixing the Model

Do the following:

- Open the 'Tree with shade' model fragment in the model fragment editor.
- Create a 'directed quantity space correspondence' from the quantity space of *Size* to the quantity space of *Shade.* Notice that a quantity space correspondence is created between the names of the involved quantity spaces.

Step 9: Interpreting the Results, Debugging, and further Issues

In each of the following step, try to understand what is shown, and discuss this with your partner.

- Select the Path from the start state to the end state (mind the order).
- Look at the Value history View.
- Choose one of the states.
- Open the Entity-Relations View, and look at the structural model.
- Open the Model Fragments View and look at the active model fragments.
- Open the Dependencies View and make it show only the quantities and the causal relations (I's and P's).
- While being in the Dependencies View: make all 'the context' appears.

Answer question:

- What is the relationship between the contents of the model fragments in the build environment, and the contents of the Dependencies View mentioned above?

Step 10: Additional Issues

- It is possible to create alternative scenarios by specifying another initial value for the quantity *Size*. Try to run these scenarios, and compare the results.
- Remove the influence from the 'Growth of tree' model fragment, and simulate the result. Look at the Value History. What is the reason only one state is generated? What is wrong with the Value History? Restore the influence.
- Remove the proportionality from the 'Tree with shade' model fragment, and simulate the result. Explain why only one state is generated. Restore the proportionality.

Assignment 2: Population Behaviour

The main goal of this assignment is to further familiarise the reader with the different kinds of building blocks that constitute a qualitative model.

Basic Population Growth

The goal is to create a model that captures the idea of a population and its growth due to birth. Essentially three ideas should be represented in the model:

- Existing populations grow.
- Growing means that the number of individuals increases.
- When the number of individuals increases, the biomass increases.

The population is thus characterised by three quantities: *Number_of*, *Biomass* and *Birth* rate. Notice, the close resemblance between these three quantities and the quantities involved in the 'Tree & Shade' system discussed above. For both systems there is a central variable (*Size* and *Number_of*) that designates the typical situations in which the system can be. Such a quantity is sometimes called the *state variable*. Second, there is a *rate* that directly (in this case positively) influences the state variable (*Growth* and *Birth*). Finally, there is a dependent variable that follows the changes in the state variable (*Shade* en *Biomass*). In fact, the two systems can be represented following the same general principles. As a result, the 'Basic population growth' model can be obtained by 'editing' the names of the model ingredients in the 'Tree & Shade' model. The following changes are required to accomplish this:

- Entity hierarchy:
    - Entity type (1 change).
- Scenario:
    - Name (1 change), entity instance (1 change), quantities (2 changes).
- Model fragments:
    - Name (2 changes), entity type (2 changes), quantities (4 changes).

The model should consist of two model fragments: a static fragment that describes the relationship between *Number_of* and *Biomass*, and a process fragment implementing the influence of *Birth*. There can be multiple scenarios with varying initial values for the quantity *Number_of*. Simulating the model should produce results similar to those of the 'Tree & Shade' model. Notice that the exact number of states that will be generated depends upon the initial value set for the quantity *Number_of*.

Competing Processes

Populations not only grow. They may also shrink and become extinct due to dying. The goal of this assignment is to include 'Death' as a second process affecting the population. This process is kind of analogous to the 'Birth' process. A quantity *Death* is introduced, with a negative influence on *Number_of*. Whilst running the simulation take notice the following:

- 'Death' and 'Birth' are competing processes resulting in ambiguity. Given a certain value for *Number_of* the population may increase, stay steady, or decrease.

- The population may become extinct. *Number_of* and *Biomass* should thus go to zero in one of the behaviour paths.
- How many states are generated? And how many behaviours are there?

An Alternative Representation

There are different ways in which 'the same' idea can be represented in a model. The goal of this assignment is to take an alternative representation, by applying the following principles:

- Introduce the quantities of interest as early as possible. Thus in addition to *Number_of* and *Biomass*, also introduce *Birth* and *Death* as quantities in the static model fragment describing population features.
- Make the processes conditional: the 'Birth' and 'Death' processes should only become active when:
    - the static model fragment describing the population features has been found *and*
    - the population exists, that is: *Number_of* > *zero*.

In addition to the alternative representation, also add feedback to this model. The idea is that the *Birth* and *Death* rates increase (or decrease) following an increase (or decrease) in the *Number_of*. This requires positive proportionalities between those quantities in the static model fragment describing the population features. One of the results of this approach is that the rates will change, and among others, *Birth* and *Death* will also go to zero when the *Number_of* goes to zero.

An Agent for Immigration

In the models created so far, a population cannot 'recover' from zero - from being extinct. The goal of this assignment is to add immigration. Consider the population discussed in the previous assignments as the system we are dealing with. Immigration can then be seen as an external influence affecting that system, namely, individuals from a different area entering the area in which this population lives. External influences are modelled using model fragments of type agent. Thus: create an agent that causes immigration to the system (population):

- Define an agent (external influence) using the 'agent hierarchy editor' (e.g. *Neighbour)*.
- Define the quantity *Immigration*.
- Adapt the structural details in the scenario. Introduce the idea of an area (e.g. an *Ecosystem*) in which a population lives while another population (e.g. the *Neighbour*) lives outside of this area.
- Create a model fragment (type: agent) in which (for the structural situation mentioned above) the quantity *Immigration* has a positive influence on the quantity *Number_of* and as such implements the idea of individuals immigrating into the ecosystem. (Ignore the details of the neighbour population in terms it becoming smaller due to individuals leaving).

When simulating the model, run a simulation in which the *Number_of* starts at *Zero*. This should result in a state-graph in which this population starts growing from *Zero* and reaches it maximum size. Also run a simulation in which *Number_of* starts at a value higher than *Zero*. Does this situation lead to the population becoming extinct (thus *Number_of* going to *Zero*)? Why not? An additional assignment could be to adapt the model such that it does go to *Zero*, e.g. by including an emigration process.