Towards a Structured Approach to Qualitative Modelling

Bert Bredeweg[1], Paulo Salles[2], Anders Bouwer[1] and Jochem Liem[1]

[1]Human Computer Studies, University of Amsterdam
Kruislaan 419 (matrix I), 1098 VA Amsterdam, The Netherlands, E-mail: {bredeweg,bouwer,jliem}@science.uva.nl
[2]Institute of Biological Sciences, University of Brasília,
Campus Darcy Ribeiro, 70.919-900 Brasília, DF, Brazil, E-mail: {psalles@unb.br}

Abstract

Creating qualitative models is often considered an art rather than a scientific undertaking, lacking a structured methodology that supports formalisation of ideas. This hampers the already difficult process of building qualitative models. This paper presents a methodology that structures and supports the capture of conceptual knowledge about system behaviour using a qualitative approach. The framework defines a protocol for representing content that supports the development of a conceptual understanding of systems and how they behave. The methodology supports modellers in two ways. It structures and explicates the work involved in building models. It also facilitates easier comparison and evaluation of intermediate and final results of modelling efforts.

Introduction

Building a model is a process during which potentially vague and general ideas become detailed and formally specified. The goal of this paper is to support developers of qualitative models and simulations in performing such a task, particularly to structure the work carried out by the modellers. The presented framework is part of a larger effort to support users in actually using Qualitative Reasoning (QR) technology to develop and organise their understanding of system behaviour, particularly concerning issues relevant to the notion of 'sustainable development'. The support effort emphasizes three aspects[1].

- *Easy to use software*. A new workbench has been developed (Garp3) that provides users with a seamless environment for creating and simulating qualitative models (Bredeweg et al., 2006b).
- *Curriculum for learning about QR*. A curriculum has been developed for potential modellers to learn about the essentials of Qualitative Reasoning and Modelling (Bredeweg et al., 2006a).
- *Structured approach to modelling*. A structured methodology has been developed concerning *how* to capture qualitative knowledge, particularly focussing on the trajectory of developing a detailed model from a general idea (Bredeweg et al., 2005). The framework assumes that the target software supports a compositional approach to qualitative reasoning and that different kinds of knowledge, such as structural, causal, etc., are explicitly represented.

Building a qualitative model is a complex task. It consists of creating a library of model fragments and accompanying scenarios such that when simulating those scenarios they produce output that answers the questions specified in the modelling goals. We present a *structured approach* that defines a protocol to support the execution and management of this modelling task. The approach has six main steps, which are depicted in Figure 1 and enumerated below:

1. Orientation and initial specification: establishing what should be modelled, why and how.
2. System selection and structural model: identification of the target system structure and its constituents.
3. Global behaviour: general specification of the behaviour that the model should capture.
4. Detailed system structure and behaviour: detailed specification of the behaviour to be captured.
5. Implementation: creation of the model ingredients in the model-building software. Simulation and debugging in order to improve and optimize the model and obtain the required results.
6. Model documentation: documentation of the model and underlying argumentation.

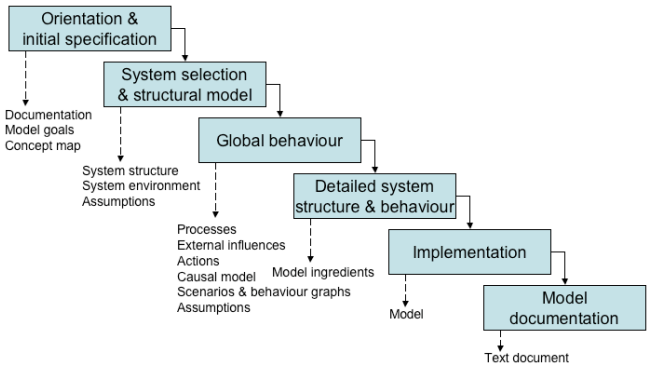

Figure 1: Structured approach to building qualitative models

The structured methodology presented in this paper is not the only way in which models can be built. However, creating a qualitative model is a difficult task, and

[1] Visit the QRM portal for details: http://hcs.science.uva.nl/QRM/

experiences in the past have shown that a structured approach, during which the model is step-wise clarified, defined, and documented, creates a momentum that makes success more likely (Salles and Bredeweg, 2003). Moreover, having all the intermediate representations and argumentations explicated significantly reduces the work that has to be done in order to establish a proper documentation of the end result. The steps in the methodology are detailed in the following sections. There is no space in this paper to illustrate all the intermediate representations and results with examples. For additional examples see Bredeweg et al. (2005).

Orientation and initial specification

Before a model can be built, ideas need to be formulated concerning the contents and purpose of the model. Three aspects are important here. First, a global textual description of the target system and how it behaves. Second, the goals for which the model is being built, particularly addressing which characteristics of the target system will be captured in the model and how they will be observable in the simulation results. Also important is the audience for whom the model is created (the end-users). Third, a concept map that highlights the important concepts relevant to the system being modelled. These three aspects are important references during the model-building process as well as for using the model afterwards because:

- Having explicit documentation on content and goals will be of help for managing the modelling process.
- When a model is being built as a collaborative activity it is crucial to develop a mutual understanding of the target system and its behaviour. This is important before the modelling effort starts, but also during the process to avoid deviation and conflicts among the modellers.
- Model goals provide means to evaluate the model and simulation results during the construction process and when it is finished.
- Model goals provide means for potential users of the model to assess the applicability of the model for their purposes.

Documentation

Documentation may take on a number of forms, but it generally has two flavours: informative and normative. Informative documentation provides background information to help developers study and understand the target system; such documentation is generally a review of published information created by others. Normative documentation, on the other hand, is created by the model developers themselves, and is the first step to express their modelling intentions, and therefore is formative for the expected end-result. Typical examples of normative documentation include a summary and overview of the main features of the model, and a (short) slide show or text presentation of the target system, which can be used to explain aspects of the system to others.

Main model goals

A model is created to serve a purpose. The purpose usually includes one or more specific features that the model exhibits and an identification of the target audience (those who will use the model). During the 'orientation and initial specification' step, the formulation of this purpose is by definition general. Also, we do not yet want to restrict ourselves to a particular vocabulary to express these goals. Hence, the goals are expressed in natural language. For a qualitative model typical goals include:

- Developing a knowledge structure that can be used to explain phenomena (to teach learners, or to support stakeholders in developing an argumentation, etc.).
- Determining all possible behaviours of a system.
- Investigating the possibility of phenomena to occur (proof that phenomena can or cannot occur).
- Investigating interactions between phenomena (Are there interactions? What kinds of interactions are there? Why are there no interactions?).
- Investigating the coexistence of phenomena (proof that coexistence is possible or impossible).

Concept map

A concept map (sometimes also referred to as an entity-relation graph) is a graphical representation of ideas that a person believes to be true (Novak and Gowan, 1984), and as such represents knowledge. A concept map consists of two primitives: nodes and arcs. Nodes reflect important concepts, while arcs show the relationships between those concepts. Figure 2 depicts an example of a concept map.

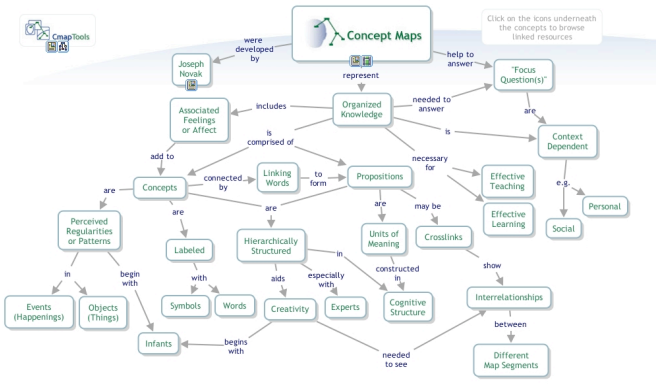

Figure 2: Concept map about concept maps (by Joseph D. Novak) (http://cmap.ihmc.us/Documentation/ - visited August 31st, 2005)

Once developed, concept maps can be used to discuss the content captured with other model builders. During this process the map can be further adjusted to accommodate alternative ideas, and may ultimately reflect a shared understanding among model builders. By making concept maps authors not only externalise information, but by

doing so they also further specify and organise their own knowledge. Concept maps can also be used to question domain experts about the correctness of the content captured and based on that input modify the map where needed.

System selection and structural model

In the 'orientation and initial specification' step the goal was to establish a broad view on the system. The purpose of the 'system selection and structural model' step is to make the initial choices concerning the structure of the system. Which entities will be included in the model and which will not? And what are the assumptions made when imposing a certain structural organisation on the system?

System structure

System structure refers to the physical world as perceived by humans. It refers to those parts of the system that in principle do not change due to the behaviour of the system. When building a model, an important step is thus to determine the structure of the system, particularly to determine the entities from which the system is built. For those things that are part of the system further details need to be specified, namely:

- Type (what kind of entity is it?) – For each entity a description must be made of what it is. Often some entities are of the same type. In that case one description of the general description of the type suffices.
- Structural relationship (how do entities relate to each other?) – For each pair of entities consider whether the entities are structurally related. If they are, provide a description of that relationship.
- Decomposition (does the entity consist of subparts?) – Often objects can be decomposed into a set of other parts (the parts from which the bigger object is made). Identify and describe these decompositions and for the subparts, describe their type, structural relationship, and possible decomposition.

System environment and external influences

When creating a model certain aspects will not be included, because they are outside of the system boundary. In general, there is a huge collection of characteristics that is not included in a model. The goal is (of course) not to enumerate everything that is not part of the model. However, for certain aspects it is sometimes relevant to explicitly mention their exclusion. These include aspects that are:

- Irrelevant – When building the model, irrelevant aspects can be ignored. There appears to be no behavioural aspect of the system that is affected by it. They are thus fully irrelevant. Ignoring usually means that they are not included in the model at all. However, it may be worthwhile to describe such aspects in the context of the 'system selection and structural model' step, particularly when the choices about what is considered irrelevant deviate from what is considered normal practice.
- Important under certain conditions – Some details can be ignored until certain boundaries are reached. After that, they need to be taken into account. As a QR engine dynamically builds the model from which it calculates the simulation results, these aspects can be made available when needed (i.e., when the conditions become true). As an example consider the possibility of an iron container melting. Only under specific conditions, for instance in the case of exceptionally hot heaters, reasoning about that possibility should be considered. In all other cases it should be ignored.
- External influences – These influences refer to impacts enforced upon a system by aspects that are in principle outside the system as such. Consider a person managing a certain vegetation, e.g. by providing water on a regular basis. In a QR model the vegetation could be considered the system, and the manager could be presented as an external factor (exogenous) that has a certain impact on the system, but that itself is not a part of the system as such.

Assumptions concerning structure

Throughout the 'system selection and structural model' step, choices have to be made concerning what to include, what to leave out, and how to include it. All these choices must be enumerated, because they help outsiders to assess the model and its simulation results. They are also important for placing a model into perspective with other models. Some of these assumptions will eventually become model ingredients and play an active role helping the engine to decide what aspects to include in a model and what to exclude. Other assumptions will not be included in the model. They remain textual descriptions of choices made and what impact those choices have on the model and its results.

Global behaviour

The purpose of the 'global behaviour' step is to make the *initial choices* concerning the behaviour of the system. There are still few restrictions on the format in this step, which allows modellers to express their ideas as much as possible. However, some focus is required in order to gradually arrive at specific model details. Hence, the global behaviour is specified using the notion of processes, scenarios and behaviour graphs, causal model, and assumptions, which reflect choices and limitations regarding the global behaviour specified.

Processes

Forbus (1984) defines processes as "… something that acts through time changing the parameters of objects in a situation. Examples of processes include fluid and heat flow, boiling, motion, stretching and compressing." These

examples originate from physics. For other domains, such as 'sustainable development', different processes are relevant (e.g., pollution, agricultural production, erosion, photosynthesis, etc.).

For the specification of the global behaviour, identification and description of the processes that govern the system behaviour is a key issue. For each process the following aspects should be clarified (textual description):

- Name and type – A process has a name and can be classified as belonging to a certain class of processes.
- Collection of entities (partial system structure) – The set of structurally related entities that provide the context in which the process is active. Obviously, these entities should be subsets of those entities identified during the 'system selection and structural model' step.
- Quantities involved – The dynamic features of the entities (of the previously mentioned collection of entities) that somehow relate to the changes caused by the process. Particularly, those features that play a role in the start and stop conditions, and the effects of the process.
- Start (triggering) conditions – Processes usually do not occur unexpectedly, but happen because some enabling condition is satisfied.
- Effects (what does it change) – The process changes features of entities. Which features are changed and how does this happen?
- Stop (ending) conditions – What are the conditions under which the process becomes inactive again?

External influences and deliberate actions

Consider for instance a hunter who shoots certain animals, or a pump that puts water from one reservoir into another. Changes such as these are usually perceived as 'external influences' affecting a system, or deliberate 'actions' carried out by agents.

Although humans perceive these changes as being different from processes, their textual descriptions are largely similar to describing processes. The main difference is the added agent or actor. That is, the person or thing that performs the action. Consider for example hunting:

- Name and type – Hunting, relates to deliberate human actions of changing the size and structure of biological populations.
- Agent (actor) – The hunter (a human being).
- Collection of entities – The hunted population (some kind of animals).
- Quantities involved – Size of the population.
- Start conditions – Current size of the population greater than the desired population size.
- Effects – Reduces the size of the population.
- Stop conditions – Current size of the population equal to, or smaller than the desired population size.

Causal model

The processes discussed above provide isolated views on the changes happening to the system. When specifying the causal model, the goal is to create an overview of how the effects of processes propagate to other features (quantities) of the system and how processes interact. The representational means used are direct influences (I's) and proportionalities (P's, indirect influences) (Forbus, 1984). Influences refer to flows initiated by processes (e.g. a *flow* of water from a tap increases the amount of water in the bathtub). Proportionalities refer to propagation of these influences to other system features (e.g. an increasing amount of water in the bathtub causes the level of the water to increase as well). Influences and Proportionalities can be positive or negative.

A causal model thus becomes an interconnected graph (potentially large) in which the nodes represent quantities and the arcs represent direct (I+/I–) and indirect (P+/P–) influences (see e.g. Figure 3). Notice that such a graph is in principle a kind of concept map as discussed before. It differs from this general notion of a concept map in that the nodes and arcs are now of a specific type (namely, quantities and I/P's, respectively) and thus have specific meaning.

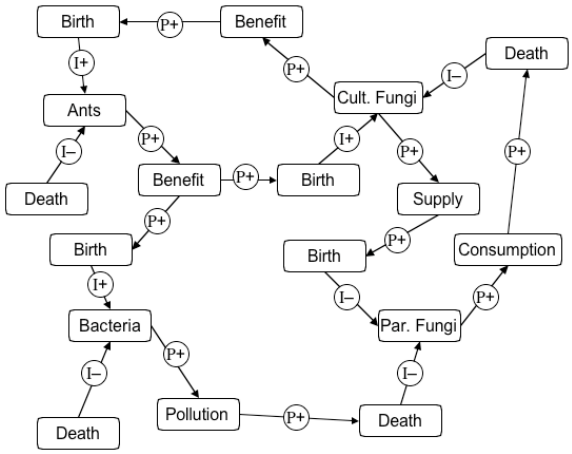

Figure 3. Causal model for The Ants' Garden (example taken from Bredeweg et al., 2005)

Scenarios and behaviour graphs

An important step in specifying the global behaviour is the identification of typical situations and behaviours. Scenarios refer to initial situations of the system. Such scenarios may trigger processes and include deliberate actions. Hence the system specified in a scenario may change. Behaviour graphs refer to series of continuous changes that the system will go through following such an initial situation. In a way, describing the typical situations and behaviours concerns the specification of the *expected* simulation output. That is, the results the QR engine should produce. Notice that the expected simulation output should somehow reflect the 'model goals' mentioned before.

Descriptions of scenarios may include the following aspects:

- Name
- Collection of entities (partial system structure)
- Agents (if any)
- Quantities
- Initial values and in/equality statements

Descriptions of behaviour graphs typically consist of a chain of changing quantity values and/or in/equality statements. Consider for instance a scenario with communicating vessels. A typical behaviour for this situation is shown in Figure 4.

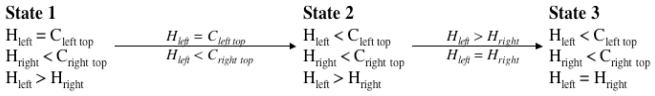

Figure 4: Example behaviour for communicating vessels (H: Column height, C: Container height)

When specifying the global behaviour, the goal is *not* to produce *the* perfect behaviour graph that includes all the details concerning the changes of the system behaviour. On the other hand, very specific scenarios and detailed behaviour graphs provide better means to evaluate the model and the simulation results it produces. They also help as a source of reference while building the model.

In addition to the sequence of states (reflecting changing system behaviour over time), three aspects are important to specifying behaviour graphs.

- Begin (start) state – These are the initial states at the start of a behaviour graph.
- End state – These are the states at which a change of behaviour stops. One may refer to these states as goal states, in the sense that the goal of the model is to predict the system behaviour and actually arrive at these end states. Hence, it is important to be explicit about what the intended end states of the simulation are.
- Branching – A state of behaviour lasts for a certain amount of time. After that, the behaviour of the system may change into a 'new state of behaviour'. However, often the transition to successor behaviours is not unique and branching may occur. Branching refers to the situation when multiple states of behaviour follow a previous state of behaviour.

A behaviour graph can have multiple begin and end states, and also multiple branches. Furthermore, branches may reunite and even produce cycles where behaviour repeats indefinitely.

Assumptions concerning behaviour

Throughout the 'global behaviour' step, choices have been made concerning which behavioural aspects to include, how to include them, and what to leave out. All these choices must be detailed, because they help outsiders to assess the model and its simulation results. They are also important for placing a model into perspective with other models.

As with the structural assumptions, some of the behaviour assumptions may eventually become model ingredients and play an active role helping the engine to decide on what aspects to include in a model and what to exclude. Other behavioural assumptions will not be included in the model. They remain textual descriptions of choices made.

Detailed system structure and behaviour

The 'detailed system structure and behaviour' step concerns the specification of *all* the model ingredients of the model using the Qualitative Reasoning vocabulary[1]. In principle QR modelling software could be used to do this specification. However, by not yet using the software model builders have more freedom. This freedom is considered essential at this stage of the model building process, because many facts still need conceptual detailing and adding implementation constraints may hamper this process. Advantages of ignoring implementation details at this point are:

- The order in which the ingredients are defined is unconstrained
- Small inconsistencies present no direct problems
- Modifying ingredients is relatively easy

When specifying the model ingredients it is worthwhile to not only enumerate them, but to also explain their purpose and meaning. Below we briefly summarise the main issues.

Structural details, Agents and Assumptions

For the structural details the following ingredients need to be specified:

- Entity types (refer to the kind of entities that may exist).
- Entity super-type relations (are used to organise the entity types in a type hierarchy)
- Attributes and their sets of attribute values (refer to static (non-changing) features of entities).
- Configurations (are binary relationships that specify structural relationships between entities).

External influences on a system are represented by:

- Agent types (refer to the kind of agents that may exist).
- Agent super-type relations (are used to organise the agent types in a type hierarchy).

Alternative viewpoints on structural or behavioural aspects can be represented by:

- Assumption types (refer to the kind of assumptions that may exist).

[1] We take a Garp3 perspective. For other engines, details may vary.

- Assumption super-type relations (are used to organise the assumption types in a type hierarchy).
- Configurations (can be used to assign an assumption to specific Entities or Agents).

Quantities and quantity spaces

Changeable features are specified by means of:

- Quantities (represent the changeable features of entities and agents, how these features can change is determined by their quantity spaces).
- Quantity spaces (are ordered sets of alternating points and intervals, critical entries in quantity spaces are sometimes called landmarks).

Detailed description of scenarios

Scenarios describe initial situations. The following ingredients can be specified:

- Name (a unique identifier).
- Assumptions (labels that may facilitate the inclusion or exclusion of model fragments).
- Agents (to represent external influences).
- Entities (represent the structural details, possibly with attribute/value pairs and configurations).
- Attributes (and specific values)
- Configurations (relations between entities)
- Quantities (represent changeable features of entities and agents, and can be given initial values).
- Initial values (magnitudes and/or derivatives).
- In/equality statements (can be used to further differentiate between quantities and values).

Detailed description of model fragments

Model fragments describe chunks of knowledge that may apply to scenarios. For a model fragment the following ingredients can be specified:

- Name
- Super-type (from which knowledge is inherited)
- Conditions
    - Assumptions
    - Agents (only in 'agent' model fragments)
    - Entities
    - Attributes (and specific values)
    - Configurations
    - Quantities
    - Values
    - In/equality statements
    - Model fragments
- Consequences
    - Quantities
    - Values
    - In/equality statements
    - Correspondences
    - Influences
        - Direct (I-/I+, only in 'process' or 'agent' model fragments)
        - Indirect (P+/P-, proportionalities)

Conditions refer to ingredients to which a model fragment applies. When the conditions are true the consequences provide additional details that apply to the situation. Name is a unique identifier. A model fragment can be a subtype of 'static', 'process', 'agent', or of an already defined model fragment. Assumptions represent viewpoints and further refine the applicability of a model fragment. Agents represent external influences. Instances of entities, possibly with attribute/value pairs and configurations, represent the structural details. Quantities represent changeable features of entities and agents, and can be assigned values. In/equality statements can be used to further differentiate between quantities and values. Model fragments refer to chunks of knowledge. Correspondences refer to coexisting values (or sets of values). Influences specify changes and propagation of changes.

Implementation

Now that in principle all the model ingredients have been specified, the next step is to actually create the model using the QR software. By creating the model, errors and inconsistencies may become apparent and will have to be fixed. In addition, running the model will generate results that are not always as expected, and modifications may be needed to improve the model.

Enumerating all these details for a certain model is beyond the scope and purpose of this paper. Figure 5 illustrates scenario results of the 'implementation step' for the Ant's Garden model using the Garp3 workbench. For details see Bredeweg and Salles (2005).

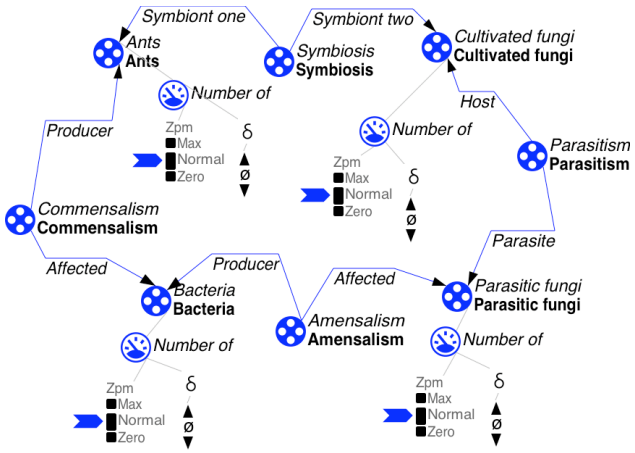

Figure 5: Ants' Garden: A typical scenario created with Garp3

Model debugging

As the modelling effort progresses and the model becomes more complex, it becomes more difficult to find the cause of problems that hamper the quality of simulations. That is why it is important to take a step-by-step approach to building the model, and to take care to understand how the implemented structure produces the behaviour shown in

the simulations. The following recommendations are considered important:

- Create the scenarios first and after that start implementing the model fragments. Scenarios should express the general ideas the modeller wants to convey and as such are important starting points for the behaviour generation.
- Be careful to correctly represent the system structure (entities, attributes, and configurations) in the scenario, and to ensure that this structure is correctly repeated in the model fragments.
- For each model fragment implemented, run a simulation and check the results. This way, undesired behaviour can be spotted and fixed as soon as it occurs and, as the model grows in complexity, the modeller can be sure that previous steps are correct.
- Continuously check the behaviour produced by processes. Given that processes start changes in the system, it is wise to be sure about the propagation of these effects throughout the whole set of quantities.

Model documentation

The model documentation should give readers sufficiently detailed information, such that the reader in principle can redo the modelling effort and arrive at the same results. In addition, the model documentation should provide the reader with sufficient information concerning the details captured by the model, so that it can be fully understood. In many aspects, model documentation consists of an organised summary of the ideas previously presented in this paper (see also Figure 1). Below, the main parts of model documentation are discussed.

Context, model objectives, and use of the model

Model documentation starts with the name of the model. This should summarise what is in the model (for example, 'the communicating vessel model', 'the Ants' Garden model'). Avoid names such as 'test2' and 'general water4'. The 'orientation and initial specification' step has produced three outputs that can be used to introduce the model context, its objectives, and use:

- Documentation (regular text and/or presentation)
- Model goals (simulation results and target audience)
- Concept map

Although the modelling effort may have changed the view on some of the initially specified details, it is important to present an account of what was intended and to what extent that was realised. So, it is expected that the model documentation always begins with a (slightly rewritten) version of the output from the 'orientation and initial specification' step.

Global structure and behaviour

The second part of the model documentation consists of the outputs created during the 'system selection and structural model' step and the 'global behaviour' step. In principle the documentation should include previously developed details concerning:

- Global structure
    - System structure
    - System environment and external influences
    - Assumptions concerning structure
- Global behaviour
    - Processes
    - External influences and deliberate actions
    - Causal model
    - Typical situations and behaviours
    - Assumptions concerning behaviour

As with the previous section, the modelling effort may have changed the view on some of these details. Therefore, it is important to present an account of what was intended, to what extent that was realised, and why modifications were needed.

Implementation details

The third part of the model documentation consists of a detailed description of all the model ingredients that constitute the implemented model. In general this can be a listing of all the ingredients created in the software annotated with the details specified in the step 'detailed system structure and behaviour'. Specifically it concerns a full description of the:

- Entity hierarchy
- Attributes and values
- Configurations
- Agent hierarchy
- Assumptions hierarchy
- Quantities and Quantity spaces
- Scenarios
- Model fragments

For many of the ingredients, screenshots of the QR workbench should be included if possible (Garp3 allows users to print EPS documents of most diagrams). Notice that the model documentation should include a description of *each* model fragment and *each* scenario. When a model fragment has subtypes, it is not necessary to repeat all the issues originating from the super type in the subtype. In such cases the documentation can focus on the newly added features in the subtype model fragments.

Simulation results

The fourth part of the documentation concerns the simulations results, based on the model. A key result of running a simulation is the state graph, which is the central output of running a scenario. The following details should be described:

- State graph
    - Scenario
    - Begin states, End states, and Branching points

- State transitions (following the Transition, Value and Equation histories)
- Behaviour paths
- States (per state details on: Structural model, Quantities and values, Model fragments, and Dependency diagram)

For each simulation to be documented, the modeller should include the name of the initial scenario and initial values of quantities, the state graph obtained from that initial scenario, definitions of initial and end states and the most relevant behaviour paths. Analysis of branching in the behaviour graph is useful for understanding ambiguities and to identify where more knowledge is required for a better representation of the system. A discussion about these ambiguities and how to solve them must be included in the model documentation.

An overview of the transitions present in a state graph is shown in the transition history. The value history and equation history may show details that are also present in the transition history, but the details may also be different. In principle, the value and equation histories show all the values and equations present in the selected states and may also include values and equations that do not change. The modeller has to decide which of these options provide the best means to show the required details.

The state graph produced during simulation consists of a set of states. The details in each of these states may be different. Probably not all states can be described fully (because there are too many details). The modeller has to decide upon the subset of states that should be discussed, for instance because they have important features. Typically, this involves the begin states, end states, and branching points. For the selected states the modeller should again decide upon the information to be discussed. Typically, the dependency diagram should be discussed; it shows how the reasoning engine assembles the model fragments into a full causal model for a given state. Possibly, the model fragments active in the state, the structural details to which the state refers and a list of the important quantities, their quantity spaces and their current values (magnitude and derivative).

In general a modeller should keep in mind that readers interested in the model and its simulation results want to learn about as many details as possible. Writing proper model documentation on the simulation results is a difficult task and time consuming. However, it is important, and should not be neglected.

Conclusion

We have presented a framework to support and organize the capture of conceptual knowledge about systems and their behaviour using a qualitative reasoning approach. The use of a structured approach assures that modellers are supported throughout the modelling process. It also makes the intermediate and final results easier to communicate and understand.

The presented framework was initially used to structure a collaborative modelling course with students in artificial intelligence at the University of Amsterdam and students in ecology at the University of Brasilia (Salles and Bredeweg, 2003). Recently the approach is successfully used in the NaturNet-Redime project (www.naturnet.org) to support domain experts in capturing conceptual knowledge on sustainable development.

We are currently implementing dedicated editors to support the steps in the methodology. An added value of such editors will be that the representations can be automatically analysed, and procedures can be developed to further support the modeller in building models, e.g., by maintaining consistency between representations.

Acknowledgement

The research presented here is co-funded by the European Commission within the 6th Framework Programme for Research and Development (2002-2006) (project NaturNet-Redime, number 004074, www.naturnet.org). Peter Barz (Environmental Network Limited) and Tim Nuttle (University of Jena) provided helpful comments. We want to thank the anonymous reviewers and the participants of the NaturNet-Redime workshop in Amsterdam (October 11-14, 2005) for their feedback and suggestions concerning the methodology.

References

Bredeweg, B., Liem J., Bouwer, A., and Salles, P. (2006a). *Curriculum for learning about QR modelling, Naturnet-Redime*, STREP project co-funded by the EC within the 6th Framework Programme (2002-2006), Project no. 004074, Project Deliverable D6.9.1.

Bredeweg, B., Bouwer, A., and Liem, J. (2006b). Single-user QR model building and simulation workbench, Naturnet-Redime, STREP project co-funded by the EC within the 6th Framework Programme (2002-2006), Project no. 004074, Project Deliverable D4.1.

Bredeweg, B. and Salles, P. (2005) The Ants' Garden: Complex interactions between populations and the scalability of qualitative models, *AI Communications*, volume 18, issue 4, pages 305-317.

Bredeweg, B., Salles, P., Bouwer, A., and Liem J. (2005) *Framework for conceptual QR description of case studies*, Naturnet-Redime, STREP project co-funded by the EC within the 6th Framework Programme (2002-2006), Project no. 004074, Project Deliverable D6.1.

Forbus, K.D. (1984) Qualitative process theory. *Artificial Intelligence*, volume 24, number 1-3, pages 85-168.

Novak, J.D. and Gowin, D.B. (1984) *Learning how to learn*. Cambridge University Press, New York.

Salles, P. and Bredeweg, B. (2003) A case study of collaborative modelling: building qualitative models in ecology. *Artificial Intelligence in Education: Shaping the Future of Learning through Intelligent Technologies*, U. Hoppe, F. Verdejo, and J. Kay (eds.), pages 245-252, IOS-Press/Ohmsha, Japan, Osaka.