On Compositional Modelling

by

Jeroen Keppens, Qiang Shen

Informatics Research Report EDI-INF-RR-0132

Division of Informatics **March 2001**
http://www.informatics.ed.ac.uk/

On Compositional Modelling

Jeroen Keppens, Qiang Shen

Informatics Research Report EDI-INF-RR-0132

DIVISION *of* INFORMATICS
Centre for Intelligent Systems and their Applications
March 2001

Presented in: Knowledge Engineering Review

Abstract :
Many solutions to AI problems require the task to be represented in one of a multitude of rigorous mathematical formalisms. The construction of such mathematical models forms a difficult problem which is often left to the user of the problem-solver. This void between problem-solvers and their problems is studied by the eclectic field of automated modelling. Within this field, compositional modelling, a knowledge-based methodology for system-modelling, has established itself as a leading approach. In general, a compositional modeller organises knowledge in a structure of composable fragments that relate to particular system components or processes. Its embedded inference mechanism chooses the appropriate fragments with respect to a given problem, instantiates and assembles them into a consistent system model. Many different types of compositional modeller exist, however, with significant differences in their knowledge representation and approach to inference. This paper examines compositional modelling. It presents a general framework for building and analysing compositional modellers. Based on this framework, a number of influential compositional modellers are examined and compared. The paper also identifies the strengths and weaknesses of compositional modelling and discusses some typical applications.

Keywords : Compositional modelling, knowledge-based system-modelling, automated modelling, model fragments

The Knowledge Engineering Review, Vol. 16:2, 157–200. © 2001, Cambridge University Press
DOI: 10.1017/S0269888901000091 Printed in the United Kingdom

On compositional modelling*

JEROEN KEPPENS and QIANG SHEN

School of Artificial Intelligence, University of Edinburgh. Email: jeroen@dai.ed.ac.uk; qiangs@dai.ed.ac.uk

Abstract

Many solutions to AI problems require the task to be represented in one of a multitude of rigorous mathematical formalisms. The construction of such mathematical models forms a difficult problem which is often left to the user of the problem-solver. This void between problem-solvers and their problems is studied by the eclectic field of automated modelling. Within this field, compositional modelling, a knowledge-based methodology for system-modelling, has established itself as a leading approach. In general, a compositional modeller organises knowledge in a structure of composable fragments that relate to particular system components or processes. Its embedded inference mechanism chooses the appropriate fragments with respect to a given problem, instantiates and assembles them into a consistent system model. Many different types of compositional modeller exist, however, with significant differences in their knowledge representation and approach to inference. This paper examines compositional modelling. It presents a general framework for building and analysing compositional modellers. Based on this framework, a number of influential compositional modellers are examined and compared. The paper also identifies the strengths and weaknesses of compositional modelling and discusses some typical applications.

1 Introduction

It is generally recognised that modelling plays a central role in intelligent problem-solving. In fact, although more research focuses on building problem-solvers, theoretical developments in symbolic artificial intelligence as early as 1968 established that modelling is at least as important as problem-solving (Amarel, 1968). Before a problem-solver can be applied to a particular problem, one is first confronted with the task of identifying the most suitable problem-solver and reformulating the problem at hand, such that the chosen problem-solver can be applied.

More specifically, intelligent problem-solving tasks are usually accomplished through three steps. First, it is necessary to commit to the most efficient and effective problem-solver by choosing the problem representation formalism. An appropriate representation of the problem at hand can then be constructed by means of the chosen formalism. This step is called model construction and the resulting problem representation is usually termed the model of the problem. Finally, an adequate problem-solving algorithm is applied on the model. In the broad sense, modelling involves the first two issues: choice of representation and model construction. Research in automated modelling, to date, focuses on model construction, however, since this is a prerequisite for resolving modelling as a whole. Therefore the remainder of this paper focuses on model construction or modelling in the narrow sense.

Model construction is a complex problem in itself. Because there is no single correct model (Leitch *et al.*, 1999), model construction involves more than mere syntax transformation. Modelling also requires reasoning about the aims and intentions of the problem-solver, and the most effective means

* The first author is financially supported by a scholarship of the Faculty of Science and Engineering of the University of Edinburgh. Both authors would like to thank the anonymous referees, Ian Miguel and Lourival da Silva for their valuable comments on earlier versions of this paper.

of applying the decisions about these to model construction whilst being economical with the available resources. Consequently, modelling is a far from trivial task and forms an inherent part of intelligent problem-solving. Automating this process would provide considerable opportunities for the associated problem-solvers. The main advantage of such automation is the increased versatility of the resulting systems. First, they simplify coping with a greater variety of situations. Because an automated modeller takes over most of the model construction effort, the user can focus on the problem specification and alternative solutions instead of modelling. Second, these systems have a more gradual decrease in performance with problems beyond the original specifications because they may be able to adapt the model with minimal user intervention when circumstances change. This may involve recoding the model to reflect different user requirements or system environment, or revising the model itself with respect to changes in the original system configuration. Automated modelling may even enable new applications, such as tutoring system design in engineering.

Several different classes of automated modellers have been proposed, specialised in different classes of task and with different architectures. This paper reviews one of the most successful classes of knowledge-based systems for the construction of appropriate problem representations: compositional modellers. A compositional modeller is essentially a knowledge-based approach to model construction. Its input is some incomplete specification of the problem at hand, including a formal high-level description of the system and the task that needs to be solved with respect to that system. Its knowledge base consists of composable pieces of knowledge called model fragments. Each of these model fragments contains a model for a part of a system. This part may constitute a sub-system or component of the system, or a process that occurs within the system. Ideally, a model fragment represents some generic part that can be found in many systems within the domain of interest. Different model fragments may represent different perspectives on the same component or process, thus reflecting alternative model-design decisions, with respect to the possible decompositions, simplifications and abstractions that can be made. The inference engine instantiates these model fragments and searches for the most appropriate combination of them. The appropriateness of a particular combination of course depends on relevance and resource economics.

The remainder of this paper is organised as follows: Section 2 describes the modelling task in more detail and relates compositional modellers to other types of automated modelling systems. Section 3 presents a formal, and general, description of the knowledge representation formalisms underlying the methodology. This is followed by a more specific examination and classification of individual compositional modellers and a discussion of the advantages and disadvantages of these modellers in Section 4. The following section summarises the strengths and weaknesses of compositional modelling and proposes a number of guidelines on using compositional modelling techniques. Section 6 describes a number of problem areas in which compositional modelling has been successfully applied. Section 7 concludes the paper.

2 Framework for automated modelling tasks

Models are approximate conceptual representations of real-world systems that enable effective and efficient problem-solvers, as they formalise only the relevant relations between the concepts of interest. When modelling a given system, which may be an object or activity, the following issues must be addressed:

- a *representation formalism*: a (symbolic) language that allows the representation of relevant aspects of the system and their relations;
- a *problem*: a task that must be solved with respect to the system, such as design, diagnosis, explanation, monitoring or prediction; and
- a *problem-solver*: an inference procedure capable of generating one or more solutions for the problem given a description of the system in the appropriate representation formalism.

A model is therefore an instance of the representation formalism that exhibits certain aspects of the behaviour or properties possessed by the system it models and enables the problem-solver to apply its inference procedures. Consequently, modelling is a process that requires the means to perceive a system's properties and behaviour, to propose promising models and to validate the model by generating its properties and behaviour and comparing these with those of the system itself. From this description, it follows that modelling may involve both perception and symbolic reasoning. Unfortunately, such an integrated modelling process is beyond the scope of any existing automated modeller. In practice, automated modellers require the user to present a formalised perception of the system or its behaviour.

This section presents a brief overview of automated modelling and proposes a broad classification in order to relate compositional modelling to alternative approaches to model construction. A complete and exhaustive classification is beyond the scope of this work. In general, automated modellers differ with respect to the set of target systems they aim at modelling, the representation formalisms they employ in the process, the types of problem task specification they hope to tackle and the associated problem-solving procedures.

Compositional modelling research is not specific to a single class of target systems. Hence the present classification does not consider this criterion. The *representation formalisms* significantly affect the range of models a modeller can represent. This criterion is elaborated in Section 2.1. The set of problem tasks and problem-solvers for which an automated modeller can formulate models constitutes the basis of the second criterion. Although the problem settings differ amongst compositional modellers, they all provide a specific means of composing models from first principles. Hence this classification focuses on the deductive or inductive nature of modellers, which is discussed in Section 2.2. Finally, in Section 2.3, the important classes of automated modellers according to these two main criteria are identified.

2.1 Representations

Because models are approximate conceptual descriptions, they may differ between one another. Model-classification frameworks provide sets of dimensions, such as precision, resolution, scope, granularity and so on, that allow a categorisation of the features of different models (Coghill *et al.*, 1999). The purpose of model construction is to find a model with suitable features given the problem at hand. The expressive power that enabled the distinctions between alternative models according to various dimensions in the first place lies in the representation formalisms used during the model-construction process. Each class of representation formalisms has its own unique characteristics that allow it to express models with different features along a specific subset of model classification dimensions.

The present discussion aims at providing a general overview of automated modellers and hence a rather broad classification of representation formalisms is proposed, which could be extended and enhanced to suit more specific purposes. Formalisms are categorised according to three levels of abstraction, as inspired by (Breunese *et al.*, 1998) and illustrated in Figure 1: technical, conceptual and mathematical.

2.1.1 Technical-level representations

Representations at the *technical level* model a domain system by relating visually, or mentally (if the system in question can not be perceived visually, e.g. an economy), distinguishable elements (usually components or processes) of the system. Thus technical models explicitly represent the structure of a system of interest. Consequently, modelling choices made at this level primarily affect the topology and scope of the resulting model.

Many kinds of problem-solving, including those tasks within which compositional modelling is applied, require the conciseness and precision of a certain rigorous mathematical formalism whereas the domain expert's most basic understanding and intuition lies in technical models. In this respect, technical models are the closest to the real-world system in the mind of the domain expert and could be considered the least abstract. In other tasks, such as design, the modeller may have a better understanding of the (required) mathematical properties of the system whereas a technical-level model

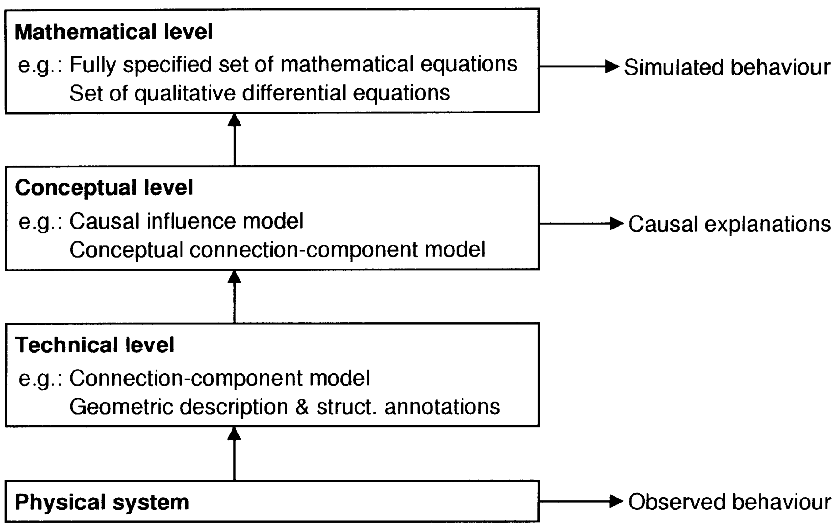

Mathematical level
e.g.: Fully specified set of mathematical equations
Set of qualitative differential equations
Simulated behaviour
Conceptual level
e.g.: Causal influence model
Conceptual connection-component model
Causal explanations
Technical level
e.g.: Connection-component model
Geometric description & struct. annotations
Physical system
Observed behaviour

Figure 1 Classification of representation formalisms

is required. In such cases, the technical-level representation is said to be the most abstract. Within the remainder of this text, abstraction will be considered from the perspective of compositional modelling, hence technical-level representations are considered to be the least abstract.

Technical models include *geometric models with structural annotations* and *component-connection models*. The former are simplified graphical representations of systems annotated with information such as properties of the elements in the graphical representation, how and by what means they are connected, and so on. For example, a simplified graphical representation of a U-Tube, together with an indication of the content of the tube being fluid, the tube being solid and rigid, the direction of gravity and the two position indicators of the fluid content's height, constitutes a geometric model with structural annotations. Component-connection models represent a system as a set of interconnected components. Each component is equivalent to a subsystem of the entire system and the connections model the input and output terminals between these subsystems.

Technical-level representations are well suited for many types of problem specification by the user because they should be relatively easily produced by people with a minimal amount of experience in the domain of discourse. For this reason they are often used as the input model to an automated modeller. Component-connection models require some domain-specific knowledge to recognise the components and how they are connected (see Biswas *et al.*, 1993, for a detailed discussion). However, this should not pose a problem for modellers with average knowledge of the domain of discourse, even though such a representation requires the user to resolve a number of the representational issues manually. For example, the level of detail at which a system's components are represented must be indicated by the user. As an alternative, geometric models on the other hand require fewer of these representational issues to be resolved, but any practical application relies on structural annotations to enrich the specifications (Amsterdam, 1992). Such annotations introduce new modelling decisions that need user intervention.

2.1.2 Conceptual-level representations

Models at the *conceptual level* represent the general notions underlying classes of subsystems or phenomena that enable the system being modelled to perform its function, including influences exerted between the subsystems. Modelling choices at this level primarily affect the granularity and order of the resulting model and have an impact upon the resolution and form of the model.

Concepts that represent a class of technical components that perform a certain function are called conceptual components and are the type of concepts used in *conceptual component models*. Similar to component-connection models, conceptual component models consist of a set of interconnected components, but instead of representing technically distinct subsystems (e.g. an electric motor), the components represent domain theory concepts (e.g. a gyrator). In the example, a gyrator is a specific type of conceptual means of transferring energy from one domain to another whereas an electric motor is one example of a piece of equipment that performs the function of gyrator. Since technical-level components and conceptual components are entirely different, the former will be referred to as “components” and the latter will be referred to as “conceptual components” hereafter. Alternatively, the concepts may be phenomena or properties of a domain. In *causal influence models*, these properties are represented by means of mathematical variables of the domain theory, such as current or velocity. The relations between these variables are causal influences representing how variables affect one another.

Bond graphs (Breedveld, 1984; Gawthrop, 1996) are an example of conceptual component models. The connections, called bonds, in these models represent transfer of energy. Energy E is itself modelled by two variables: flow f and effort e, such that $E \equiv f \times e$. The flow and effort variables and their derivatives have specific meanings and dimensions for each energy domain. Additionally, a small number of conceptual components can model specific constraints between effort, flow and their respective derivatives or model transfers between different energy domains. Again, these constraints represent specific functions for each energy domain. As such, bond graphs are able to represent a wide variety technical engineering systems within a uniform framework and by means of a small number of concepts.

Examples of causal influence models include models developed using domain theories such as qualitative process theory (QPT) (Forbus, 1984). In QPT, the relations between concepts are modelled by means of direct and indirect influences. A direct or differential influence $I^{\pm}(X_i, Y)$ means that an increase of the influencer X_i tends to increase/decrease the rate of change of the influencee Y:

$$I^{+}(X_i, Y) \equiv \frac{dY}{dt} = f(X_1, \ldots, X_i, \ldots, X_n) \qquad \text{with } \frac{\partial f}{\partial X_i} > 0$$

Similarly, an indirect or functional influence, or qualitative proportionality, $Q^{\pm}(X_i, Y)$ means that an increase of X_i tends to increase/decrease Y:

$$Q^{+}(X_i, Y) \equiv Y = f(X_1, \ldots, X_i, \ldots, X_n) \qquad \text{with } \frac{\partial f}{\partial X_i} > 0$$

Extensions of QPT provide additional expressive power. Qualitative influence diagrams (Heller & Struss, 1996), for example, can handle more specific types of functional dependencies, more complex types of influence combination. This work also introduces operators to automatically generate abstraction and approximations which are very useful in automated modelling.

2.1.3 Mathematical-level representations

Mathematical-level representations describe a system’s behaviour by means of a set of equations. The behaviour captured by such a model can be reproduced by the use of a *simulator*. Numeric mathematical representations are the most precise type of model, in that they generate a single, uniquely specified, behaviour description per model and the description is presented in real values. For example, a continuous simulator, such as DYNAMO (Pugh, 1976), can produce a system’s behaviour from a set of *ordinary differential equations* (ODEs) by extrapolating initial values based on the continuity and differentiability assumptions. Modelling choices at this level primarily affect precision, uncertainty, form and resolution (Leitch *et al.*, 1999). As opposed to other types of model, numerical models require parameter value estimates. Parameter estimation is a problem that requires a data set of observations and the structure of the equations assumed to explain the underlying relations between the observed variables.

Qualitative differential equations (QDEs), on the other hand, encompass an entire class of behaviours (see Kuipers, 1994, for a detailed discussion). In this representation, each variable takes on a qualitative state, rather than a value. A qualitative state typically consists of a quantity and a rate of change. The quantities are taken from quantity spaces which are ordered sets of so-called landmark values representing mathematically odd points in physically significant values. Rates of change are often denoted by the sign of the derivative of the variable in question, thereby representing one of increasing, steady or decreasing. Relations between variables are modelled by qualitative constraints.

Given a number of variables, qualitative constraints on these variables and the initial qualitative states for all or some of the variables, a qualitative simulator, e.g. the QSIM algorithm (Kuipers, 1986), extrapolates all possible and qualitatively distinct behaviours from the initial state, also based on continuity and differentiability assumptions regarding the system's behaviour. Impossible behavioural descriptions generated are then removed by checking the consistency between the QDEs and the generated states. Discontinuities require a new model and therefore new QDEs valid within an operating region, which is a range of qualitative values for each of the variables. The set of lists of subsequent qualitative states produced in this way for each variable in the set of QDEs that models the system is called the *envisionment*.

The equations in mathematical models enable the domains of the variables to be constrained to anything between a single value or a range of values. Therefore mathematical models are more suitable for simulation, which is a prerequisite for many applications such as prediction, monitoring, diagnosis and so on. The conceptual counterparts of these mathematical models are not well equipped for this purpose. In general, conceptual relations (such as QPT influences) are not based on the closed-world assumptions, and hence no generic statements on their behaviour can be made since there may be interactions with unknown conceptual relations. Also, some conceptual formalisms (such as bond graphs) implicitly combine several variables in a single concept and hence their granularity is too low. However, the absence of a closed-world assumption makes the relations highly composable and useful as an intermediate representation formalism for some automated modellers (Farquhar, 1993). The relative simplicity of the conceptual formalisms whilst containing the essence of the mechanics of the underlying domain theory makes them suitable as a representation for supporting explanation generation.

It is worth noting that there are alternative approaches to the representation of QDEs for qualitative simulation. For instance, the Mycroft system (Scott & Coghill, 1998; Coghill & Chantler, 1999) represents and infers the behaviour information about a physical system using a layered notation such that qualitative descriptions of higher-order derivatives can be obtained by differentiating QDEs represented at lower levels. Nevertheless, at each layer the explicit information regarding the system's structure and behaviour is also expressed at the mathematical level.

2.2 *Modelling-task descriptions*

In addition to the use of the criteria regarding model representations, modellers can be classified according to the descriptions of modelling tasks. Although AI models have been applied to a variety of application domains, AI research into automated modellers focuses on constructing models for physical systems (or more generally on application of the natural sciences: chemical systems, environmental systems and so on). The modellers usually assist in a design, diagnosis, explanation, monitoring or prediction task. A modeller's success at assisting such a task partially depends on the representation it is capable of using. It also depends on how well the model may be embedded in the application architecture of the problem-solver. The interface that allows an automated modeller to be embedded in a problem-solver typically prescribes whether the modeller works by reasoning – deductive, inductive or hybrid. These approaches are discussed here.

2.2.1 *Deductive modellers*

The deductive approach to modelling transforms a model of one type into a model at another level of abstraction by instantiating generic laws of the domain theory. Deductive modellers differ with respect to the organisation of the knowledge base and the associated model deduction algorithms.

As mentioned earlier, *compositional modellers* use a knowledge base of composable and generic model fragments. The input usually consists of a technical (component-connection) model, although this is not required. This input allows compositional modellers to instantiate a large number of components for the elements of the input component. The allowed combinations of instantiated model fragments are restricted by various structural, operational and domain constraints. The task of a compositional modeller is to find such an allowed combination and possibly an optimal combination with respect to some performance indicator, such as a relevance or simplicity measure.

As an example, the function-sharing program presented in Ulrich (1988) shows a knowledge-based approach to simple design tasks. The input consists of a specification of *desired* or *required* behaviour. Based on this specification, it searches a component-connection model of a system that can produce the required behaviour. The actual search algorithm proceeds in two phases. First, a bond graph that exhibits the desired behaviour is searched. Next, the conceptual components of the bond graph are assembled into technical components. What makes this approach deductive rather than inductive (see below) is that the behavioural data are specifications rather than random observations and that the model is constructed by applying principles to specifications rather than discovering principles from a case.

Pure deductive modelling can be a difficult task. Little work exists to validate the generated deductive models by means of observed behaviour. Therefore a deductive modeller may need to be combined with other software to validate its results (see Section 2.2.3). However, certain problem settings, such as tutoring and design, are deductive by nature. As discussed in Section 6.1, deductive compositional modelling has been successfully applied to these domains. Deductive modelling may involve the use of large knowledge bases. This implies that the deductive process may produce a large space of possible models that would require pruning, for example, by considering expected behaviours. Again, in such cases, a hybrid modeller may be more appropriate.

2.2.2 Inductive modellers

Inductive modellers generalise the *observed* behaviour of a system into an actual model of the system that explains the observed behaviour. In literature, this task is often named *systems identification* (Kay *et al*., 2000). As with deductive modelling, purely inductive system identification is a very complex task because, in theory, the search space is infinite. This problem is particularly acute when the precise model of a complex behaviour must be determined, since that involves adding many new variables and the construction of complex equations. Also, parsimonious models only approximate observations and, therefore, these model constructors need a way to determine what approximations are adequate. Distinguishing noise from interesting behavioural characteristics without any underlying theory is likely to suffer from misjudgements (Say & Kuru, 1996).

The observations used by an inductive modeller can be as precise as real-valued time-series data or consist of more general time-dependent constraints over a number of mathematical properties of the system (e.g. a QSIM envisionment that represents the observed behaviour of the system). Conventional system-identifiers also need additional knowledge (e.g. the structure of the mathematical equations) to evaluate model quality. Recently, a number of qualitative system identifiers have been proposed that operate by focusing on more specific but important families of models. This *inductive bias* (Mitchell, 1997) is achieved by restricting the operators used in a model's equations and by handling incomplete knowledge on relevant parameters, parameter values and measurement errors with qualitative reasoning techniques.

In Ramachandran *et al*. (1994) an approach is proposed to learn models of QDEs over multiple operating regions from a QSIM envisionment. This is achieved by: (1) locating the discontinuities in the behaviour trace so as to determine the operating regions, (2) learning a set of QDEs for the behaviours within each operating region, (3) identifying the operating conditions of each region and (4) trying to unify operating regions with identical QDEs or operating conditions. Within a single postulated operating region, a set of QDEs can be learned (via step 2) by programs such as MISQ (Richards *et al*., 1992) and Qualitative System Identification (QSI) (Say & Kuru, 1996). QSI, for example, first runs a constraint-determination algorithm on the envisionment. This algorithm searches through all syntactically possible constraints for a set that applies to the envisionment. The set of constraints found in this way is then used for qualitative simulation (e.g. QSIM). This is called the depth test. If the model that consists of this set of constraints can reproduce the exact envisionment used as input, it is sufficiently "deep" to accurately represent the internal mechanism of the system being identified. Otherwise, the model is said to be too "shallow" and must be deepened by extending it with new variables. These variables are defined by relating them to variables that are already in the model (e.g. a new variable could be the derivative of an existing variable). Using these constraints, and a set of heuristics that prefer minimal change in consecutive qualitative states, the qualitative states of these new variables are computed for the time points and intervals of the observation input. Each model extension phase is followed by a new constraint-determination phase and a depth test. This procedure is repeated until a sufficiently deep qualitative model is found.

In Washio and Motoda (1998) a system of two algorithms is proposed to discover basic mathematical laws that make up a domain theory, with the results represented as first-principle equations. In this approach, the first algorithm proposes a set of parameterised first-principle equation candidates using a set of variables and information on their scale and dimension. The second algorithm refines the findings of the first by fitting the parameters and testing the result.

A pure inductive modeller does not use any first principles to validate the models it produces. Therefore inductive models are sensitive to errors in the observations. Noise-filtering algorithms may reduce this problem but these risk eliminating significant features of the behaviour as well. Additionally, the underlying assumptions of pure inductive models can not be determined. As a result, evaluation of models in terms of user requirements, completeness with respect to relevant phenomena and components of interest are not possible. Finally, the computational complexity and usefulness of the machine-proposed variables and constraints, resulting from qualitative system identification, may be prohibitive in practical applications. In large systems, certain variables are naturally related because of the underlying laws of the domain theory or because of the way the system has been engineered. Observations do not contain such information, hence all variables are potentially related with one another via any number of intermediate variables.

2.2.3 Hybrid modellers

Being a compromise between inductive and deductive modellers, hybrid modellers use both an initial model and consequent data.

Deductive modellers that validate with intended behaviour. These approaches construct models using deductive techniques, but validate each constructed model against a specification of the expected behaviour provided by the user. For example, a minority of compositional modellers (e.g. Nayak & Joskowicz, 1996), though of a great significance, use an expected behaviour description as part of their input. Validation encompasses a comparison of the result of applying continuous simulation onto a candidate model, from one or more initial states, against the specification.

Deductive model construction and inductive validation. These methods deduce an intermediate model from a given scenario description but validate the model by means of actual observations. The MM program (Amsterdam, 1992), for example, produces QDEs from geometric or component-connection models through bond graphs. It uses a rule base to generate potential bond graphs that describe the presumed mechanism of the system. The generated bond graph is then translated into a set of QDEs. A variant of QSIM generates the model's envisionment. If the resulting (qualitative) behaviour matches the given observations, the model is retained. Otherwise, a number of model-correction rules are attempted or another model is constructed. This generate-and-test loop continues until a model is found that matches the observations.

Graph of models. The graph of models (GoM) (Addanki *et al.*, 1991) is an early and influential approach to automated modelling that selects a model from a graph relating alternative models in terms of their underlying assumptions. The modelling task tackled by GoM consists of searching the simplest model in the graph that sufficiently approximates observed behaviour.

The vertices in a graph of models represent the alternative models and assumptions that justify their use. These nodes are connected by directed edges that represent how one model can be replaced by an alternative. All edges are annotated by a set of assumptions that must be added, and a set of assumptions that must be retracted, when substituting the model from which the edge originates with the model at which the edge is directed.

The assumptions are grouped in so-called assumption classes. Each assumption class represents a dimension along which alternative models can be classified and from which a single assumption must be chosen. As such, the assumptions of an assumption class are mutually exclusive and, for the sake of completeness, always include an assumption that represents the irrelevance of the assumption class. This is similar to certain approaches of compositional modelling.

In GoM, any chosen model may be deemed invalid because of various conflicts. Internal conflicts arise when the parameter values predicted by the model are inconsistent with the constraints defined in the underlying assumptions of the model. For example, an internal conflict occurs in a model that considers $sin(\theta) = \theta$ under the assumption that $-1.9 < \theta < 1.9$ and that predicts $\theta = 2.4$. Empirical conflicts are the result of discrepancies between predicted and observed values beyond the allowed tolerance boundaries. Parameter change rules dictate the effect of assumption transitions on the various parameter values in a model and these are used to find the assumption transitions that rectify empirical conflicts.

Based on the conflicts detected in a model, a set of potential transitions can be derived that potentially reduce or alleviate one or more inconsistencies. Heuristics are proposed in Addanki *et al.* (1991) to choose the model transition that is likely to resolve most conflicts, and hence to focus the search. The task of model switching in a GoM based on discrepancies between observed and predicted behaviour is formalised by work on approximation reformulations (Weld, 1990; Weld & Addanki, 1991).

Induction from a model space. This class of modellers take a data set, a set of specifications of candidate models and methods to explicate the candidate models and evaluate them. In conventional system-identification systems, the data set is collected according to a purpose-built experimental design (Fedorov, 1972). The models in the candidates set are then fit to the collected observations and evaluated using standard statistical techniques (Neter *et al.*, 1996). A detailed overview of the techniques involved in conventional system identification is presented in (Ljung, 1999).

PRET (Bradley & Stolle, 1996) is an alternative approach based on the conventional system-identification problem setting, specialised in selecting ordinary differential equations (ODEs) from a set of possibly conflicting hypotheses. However, it uses general ODE theory and domain-specific knowledge to identify and instantiate the hypotheses that match the observations rather than statistical techniques.

Semi-QUantitative system IDentification (SQUID) (Kay *et al.*, 2000) also falls into this category. SQUID represents the space of models in multiple levels. At the structural level the form of the differential equations is described. Such structural differential equations are initially given. The qualitative level breaks each model variable of the structural level into an ordered set of landmarks and adds information on the nature of the functions concerned. The semi-quantitative level adds further information by assigning numeric boundaries to the landmarks and by specifying a pair of boundary functions within which the actual ODE must lie. The approach taken by SQUID is to refine models through the different levels. Refinement operators specific to each level further specify a (qualitative, semi-quantitative or ordinary, depending on the level in question) differential equation and hence refute parts of the model space that do not match the observations. Continuous simulators, again specific to each level, are used to produce the range of behaviours implied by a differential equation that must be matched with the observations. For instance, at the qualitative level, QSIM is used to simulate the model represented at the qualitative level.

2.3 Summary

This section has presented the different issues involved in modelling with respect to a problem-solving task. Model construction, or the transformation of one problem representation into another representation more suitable for problem-solving, has been identified as the sub-task of modelling solved by most automated modellers. An overview was presented of important classes of algorithm and

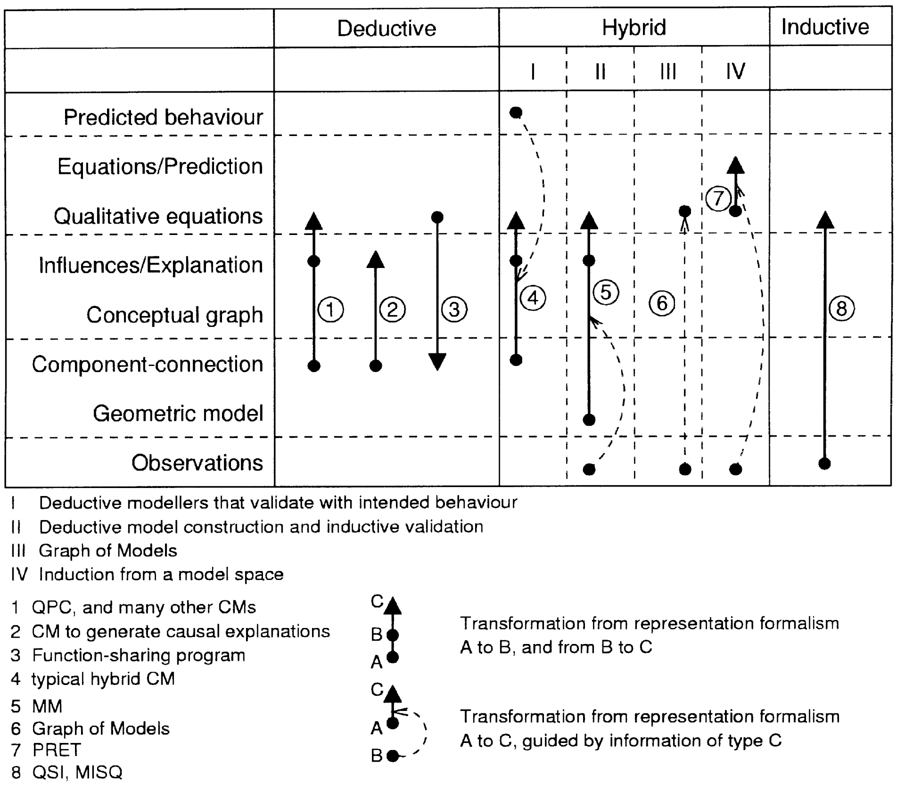

Deductive
Hybrid
Inductive
I
II
III
IV
Predicted behaviour
Equations/Prediction
Qualitative equations
Influences/Explanation
Conceptual graph
Component-connection
Geometric model
Observations
1
2
3
4
5
6
7
8
I Deductive modellers that validate with intended behaviour
II Deductive model construction and inductive validation
III Graph of Models
IV Induction from a model space
1 QPC, and many other CMs
2 CM to generate causal explanations
3 Function-sharing program
4 typical hybrid CM
5 MM
6 Graph of Models
7 PRET
8 QSI, MISQ
C
B
A
Transformation from representation formalism A to B, and from B to C
C
A
B
Transformation from representation formalism A to C, guided by information of type C

Figure 2 Classification of some important automated modellers

system for automated model construction. These automated modellers have been categorised with respect to two important criteria: the representation formalisms used for input, intermediate computations and output and the deductive or inductive nature of the model-construction process. Figure 2 illustrates this categorisation for a number of automated modellers discussed in this section. The arrows represent the transformations between representation formalisms that are performed by the different automated modellers. The classes of representation formalism in this figure are the same as those of Section 2.1. The columns indicate whether the automated modellers are deductive, inductive or hybrid.

Both criteria classify automated modellers with respect to certain properties of the types of problem these modellers can solve. The possibilities of adapting a problem specification such that it fits into another category are often limited. Therefore any comparison of automated modellers beyond the boundaries of a single category should be put into this perspective. It is advisable that any application of automated modelling be preceded by initially identifying the purpose of modelling, the available domain knowledge, the nature of available data and the required output. Such an initial analysis may be helpful in finding the appropriate automated modelling technique for a specific problem and, more importantly, may save the developer the excessive costs of choosing an unsuitable tool for the task at hand.

However, not all automated modellers fit into one single category. Compositional modelling, in particular, has been applied to a wide variety of problems. Although the technology is predominantly knowledge-based, both purely deductive and hybrid variants exist. As the next sections show, it has also been applied to transformations between different types of representation formalism. Consequently, compositional modelling constitutes an important and broad class of automated modellers.

3 Overview of compositional modelling

Compositional modellers are a class of knowledge-based modellers that, based on an initial model and a task specification (e.g. an initial state specification), instantiate a knowledge base of composable pieces of models and combine an appropriate subset of these pieces into an adequate model. The term compositional modelling was introduced in (Falkenhainer & Forbus, 1991), and is herein used to refer to a much wider class of similar modellers as is common in literature (Kuipers, 1994).

This section presents a general overview of compositional modelling. First, the modelling task of compositional modellers and the framework within which it operates is explained. Then, the most essential knowledge-representation formalisms that distinguish compositional modellers from other types of automated modeller are introduced in Section 3.2. Finally, Section 3.3 shows how model composition is achieved by this approach to automated modelling. The concepts that are defined in this discussion are illustrated by means of a small and simple example and are primarily based on the seminal work in the field presented in Falkenhainer and Forbus (1991).

3.1 The compositional modelling task

A compositional modeller is either a deductive or a hybrid modeller that constructs a mathematical or conceptual model based on a technical level input. Figure 3 presents a generic architecture for compositional modellers. A compositional modeller takes a scenario and a task specification as its input. The scenario constitutes the technical-level input, and the task specification is a formal description of the criteria imposed upon the model to be composed.

Model composition occurs through several stages. First, an *inference mechanism* instantiates the constructs of the knowledge base, such as model fragments and rules, that apply to the scenario. The model fragments describe how certain components, processes or concepts can be modelled. Different model fragments may have the same scope, representing the same component, process or concept.

Model fragment selection chooses a subset of the instantiated model fragments resulting from the inference phase by means of the task specification. Task specifications come in a variety of forms and are normally specific to each implementation since many compositional modelling-based systems are specialised to cope with one particular type of task (though the underlying ideas may well be generalised for other types of application). They are usually represented either as a *query* – request to describe the relationship between certain model components or to specify the value and direction/rate of change of a variable – or as an *initial state* of a model from which the problem-solver must extrapolate future behaviour. In compositional modellers that perform a hybrid modelling task, the task specification includes a description of the expected behaviour of the resulting model.

The selected instantiated model fragments are then composed into a model. This *model composition* phase may use various techniques. In particular, *consistency-checking* determines whether the underlying assumptions of the model fragments are compatible with one another and whether the set of equations can combined (i.e. the resulting set of equations is not overconstrained or underconstrained) (Levy *et al.*, 1997). *Causal ordering techniques* may be used to establish cause-and-effect relations over the variables in the models. Causal relations are typically required by certain

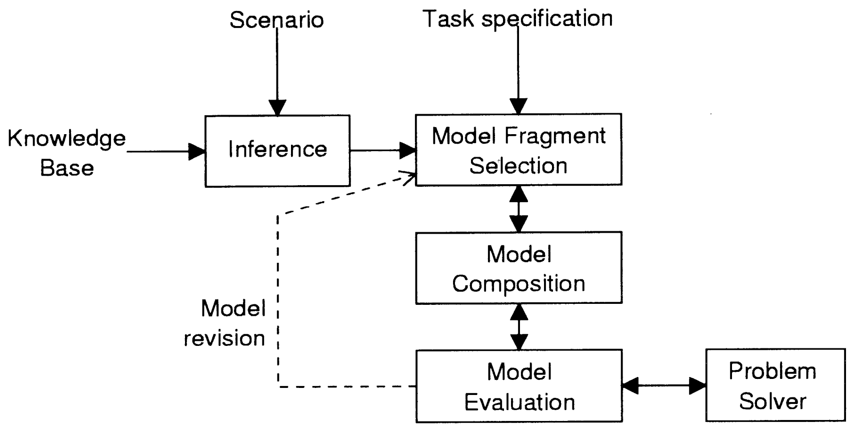


Figure 3 Generic architecture of compositional modellers

problem-solvers, such as explanation generators and diagnostic engines (Nayak, 1995). An *equation processor* transforms an intermediate conceptual model into a mathematical model (e.g. translating QPT influences into QSIM constraints (Farquhar, 1993)). Problem-solvers that perform simulation, for example, need a mathematical model of the scenario.

The models that are generated during the model composition phase are to be used by the problem-solver. However, not all models are equally suitable. The quality of a model depends upon the adequacy of the underlying assumptions, the necessity of the components, processes and/or concepts that are included in the model and the overall complexity of the model. In the *model-evaluation* phase, alternative models are assessed and the best alternative is passed on to the problem-solver.

During the model-evaluation and problem-solving stages, new information may be derived that contradicts earlier assumptions. For example, certain variables may not remain within the operating ranges assumed by the model (Farquhar, 1993). This information is fed back to the model fragment selection phase, which replaces the affected model fragment and hence revises the model accordingly.

Compositional modelling is conceptually close to certain formal theories of modelling that study how mathematical models can be derived from technical or conceptual representations. Compartmental modelling, for example, conceptually represents systems as compartments representing amounts of homogenous materials and flows between them. Formal theories have been developed to translate such descriptions into mathematical models and to analyse the represented systems (Walter, 1999). Incidentally, the compositional modelling techniques mentioned in (Struss & Heller, 1998, 1999) use compartmental modelling as the underlying paradigm for their ontology and model fragments. A related paradigm is system dynamics (Forrester, 1961), which organises a system as flows between stocks that are controlled by a causal network of influences. System dynamics is used as an underlying paradigm of the compositional modelling techniques presented in Keppens and Shen (2000).

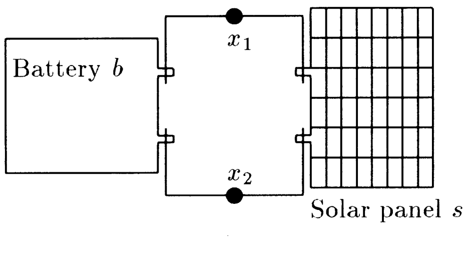
3.2 Essential knowledge-representation formalisms

3.2.1 Scenario and scenario models

The compositional modeller's task is to transform a general, user-friendly representation of a system, e.g. a component-connection model, into another more specific representation of the same system that can be manipulated by the problem-solver, e.g. a mathematical model. In the remainder of this section, a *model* is represented as a pair $\langle O, C\rangle$ where O is a set of object constants $\{o_1, \ldots, o_v\}$ and C is a set of relations over these object constants $\{c_1(o_1, \ldots, o_v), \ldots, c_w(o_1, \ldots, o_v)\}$.

A *scenario* is a model that is given as part of the task description. It is usually (but not necessarily) specified using a representation formalism at the technical abstraction label, e.g. via a component-connection representation. Note that a scenario is usually assumed to be *true* and *valid.* A scenario is *true* if it models a system that is not impossible in a given domain, otherwise it is false. A planet in a square orbit around a star without external forces at work is an example of a false scenario. A scenario is *valid* if it is an accurate representation of the real-world system.

An example scenario, which formalises the situation depicted in Figure 4(a), is given in Figure 4(b). This particular scenario represents a rechargeable battery that is hooked up to a solar panel.



(a) Physical system

battery(b), solar-panel(s),
rechargeable(b),
electrical-connection(b, x_2, x_1),
electrical-connection(s, x_1, x_2).

(b) Specification

Figure 4 A sample scenario

3.2.2 Model fragments

As Section 3.1 shows, a compositional modeller essentially performs the complex transformation of a very simple and abstract representation of a system (a scenario) into a conceptual or mathematical model that contains sufficient detail for problem-solving purposes. It is therefore necessary to differentiate between a source-model $M_s = \langle P_s, C_s \rangle$ and a target-model $M_t = \langle P_t, \mathrm{C}_t \rangle$. In this paper, the participants in the sets P_s and P_t are called source-participants and target-participants respectively, and the constraints in the sets C_s and C_t are called source-constraints and target-constraints respectively. As mentioned earlier, the knowledge base of a compositional modeller consists of composable pieces of sub-system models, called model fragments. Each of these model fragments relates a particular scenario of a subsystem to its equivalent representation in the target modelling language, under certain conditions. Definition 1 formalises the content of a model fragment.

Definition 1 (*Model fragment*). A model fragment μ is a tuple $\langle P^s, P^t, C^s, C^o, C^t, A \rangle$ *where*

- $P^s(\mu) = \{p_1^s, \ldots, p_m^s\}$ is a set of *source-participants,*
- $P^t(\mu) = \{p_1^t, \ldots, p_n^t\}$ is a set of *target-participants,*
- $C^s(\mu) \cup C^o(\mu)$ is a set of *preconditions,* where $C^s(\mu) = \{c_1^s, \ldots, c_v^s\}$ is the set of *structural conditions* and apply over the vector of source-participants $\vec{p_s}(\mu) = (p_1^s, \ldots, p_m^s)$ and $C^o(\mu) = \{c_1^o, \ldots, c_w^o\}$ is the set of *operating conditions* and apply over the vector of target-participants $\vec{p_t}(\mu) = (p_1^t, \ldots, p_n^t)$.
- $C^t(\mu) = \{c_1^t, \ldots, c_u^t\}$ is a set of *postconditions* which are constraints that apply over $\vec{P_s}(\mu)$, and
- $A(\mu) = \{a_1, \ldots, a_s\}$ is a set of *assumptions,*

such that for $i = 1, \ldots, u$

$$\forall p_1^s, \ldots, p_m^s, \exists p_1^t, \ldots, p_n^t$$

$$c_1^s \wedge \ldots \wedge c_v^s \rightarrow (a_1 \wedge \ldots \wedge a_s \rightarrow (c_1^o \wedge \ldots \wedge c_w^o \rightarrow c_i^t))$$

In most compositional modellers, the target modelling language consists of some type of (qualitative) constraints over variables. Therefore target-participants are usually called *quantities* and source-participants are known as *participants* or *individuals*.

An example of a model fragment is given in Figure 5. This model fragment applies to a battery B that is electrically connected to the connection points X_1 and X_2. If applied, it introduces four new target-participants to the scenario model: voltage V generated by the battery, a nominal voltage V_0 of the battery, the charge level CL of the battery and a charge level threshold C_0: for the battery. The implication that is formalised by this model fragment states that, if the B is electrically connected, and under the assumptions that B is not damaged, the binary voltage model for B is required and appropriate and the charge level CL of B is above its threshold CL_0, then $V = V_0$.

Source-participants: B, X_1, X_2

Structural conditions: battery(B), electrical-connection(B, X_1, X_2)

Target-participants: V, V_0, CL, CL_0

Assumptions: $\neg$ damaged(B), binary-voltage(B)

Operating conditions: $CL \geq CL_0$

Postconditions: voltage(B, V), nominal-voltage-level(B, V_0), charge-level(B, CL), charge-level-threshold(B, CL_0), $V = v_0$

Figure 5 Sample model fragment

3.3 *Model composition*

Model fragments are stored and organised in what is called a *domain theory* or a *model fragment library*. By using the standard operations of substitution and modus ponens, new knowledge can be derived from existing knowledge through the implication defined by the model fragments. In this respect, the usage of a model fragment can be defined as follows.

Definition 2 (*Usage of a model fragment*). A model fragment μ is *applicable* with respect to a scenario $\langle P_s, C_s\rangle$ if a substitution σ exists such that $P^s(\mu)\sigma \in P_s\sigma$, the structural conditions μ, are logically entailed by the source-constraints of the scenario: $C_s \models C^s(\mu)$. A model fragment is *applied* with respect to μ if it is applicable with respect to μ and its assumptions $A(\mu)$ are consistent and are assumed true. A model fragment is *active* with respect to μ if it is applied with respect to μ and its operating conditions hold and are consistent.

Determining the applicability of a model fragment is achieved by pattern-matching in much the same way as in production systems. However, a compositional modeller aims at constructing different models depending on the task at hand. The scenario model that is most appropriate for a given task must be discovered under incomplete knowledge, that is, made explicit by committing to the assumptions and operating conditions of certain model fragments. The eventual scenario model can be defined as follows.

Definition 3 (*Scenario model*). A set of model fragments $\Phi = \{\mu_1, \ldots, \mu_n\}$ defines a scenario model $M_t = \langle P_t, C_t\rangle$ of a scenario $M_s = \langle P_s, C_s\rangle$ in which $P_t = \cup_{i=1}^{n} P^t(\mu_i)$ and $C_t = [\cup_{i=1}^{n} C^o(\mu_i)] \cup [\cup_{i=1}^{n} C^t(\mu_i)]$ if $\forall \mu \in \Phi : \mu$ is applicable with respect to M_s, $P_s = \cup_{i=1}^{n} P^s(\mu_i)$ and $C_s = \cup_{i=1}^{n} C^s(\mu_i)$.

Consider, for example, the domain theory presented in Figure 6. Based on this domain theory, various scenario models can be constructed. The following model can be derived via the inference shown in Figure 7:

$$CL = \int I(x_1, x_2)\, dt \tag{1}$$

Model fragments μ_i with $P^s(\mu_i) = B, X_1, X_2$; $C^s(\mu_i) =$ battery(B), electrical-connection(B, X_1, X_2); $P^t(\mu_i) = V$ and $C^t(\mu_i) =$ voltage(B, V).
$\mathcal{C}$(constant-voltage(B)) $\rightarrow V = (0, \infty)$
$\mathcal{C}$(normal-degrading-voltage(B)) $\rightarrow \frac{d}{dt}V = (-\infty, 0)$)
$\mathcal{C}$(binary-voltage(B))$\wedge$ charge-level(B, CL) $\wedge$ charge-threshold(B, C_0) $\wedge$ $CL < C_0 \rightarrow V = 0$
$\mathcal{C}$(binary-voltage(B))$\wedge$ charge-level(B, CL) $\wedge$ charge-threshold(B, C_0) $\wedge$ $CL \geq C_0 \rightarrow V = (0, \infty)$
$\mathcal{C}$(charge-level-sensitive(B))$\wedge$ charge-level(B, CL) $\rightarrow V = M^+(CL)$
$\mathcal{C}$(temperature-sensitive(B))$\wedge$ charge-level(B, CL) $\wedge$ temperature-of(B, T) $\rightarrow V = M^+(CL, T)$
Model fragments μ_i with $P^s(\mu_i) = B, I, X_1, X_2$; $C^s(\mu_i) =$ rechargeable-battery(B), electrical-connection(B, X_1, X_2); $P^t(\mu_i) = CL$
$\mathcal{C}$(constant-charge-level(B)) $\rightarrow CL = (0, \infty)$
$\mathcal{C}$(normal-accumulation(B)) $\rightarrow CL = \int Idt$
$\mathcal{C}$(accumulation-with-aging(B)) $\rightarrow CL = \int Idt - (k \times DOD + l \times TSLC)$
Other model fragments
solar-panel(S) $\wedge$ electrical-connection(S, X_1, X_2) $\rightarrow$ voltage(S, V) $\wedge$ solar-energy(E_l) $\wedge$ $V = M^+(E_l)$
electrical-connection(C, X_1, X_2) $\wedge$ voltage(C, V) $\rightarrow V(X_1, X_2) = C^+(V)$
electrical-connection(C, X_2, X_1) $\wedge$ voltage(C, V) $\rightarrow V(X_1, X_2) = C^-(V)$
battery(B) $\wedge$ electrical-connection(X_1, X_2) $\wedge$ $V(X_1, X_2) < 0 \rightarrow$ resistance(B, R)
electrical-connection(C, X_1, X_2) $\wedge$ resistance(C, R) $\rightarrow \frac{1}{R(X_1, X_2)} = C^+(\frac{1}{R})$
$R(X_1, X_2) \wedge V(X_1, X_2) \rightarrow I(X_1, X_2) = \frac{V(X_1, X_2)}{R(X_1, X_2)}$

Figure 6 Model fragments of a sample domain theory

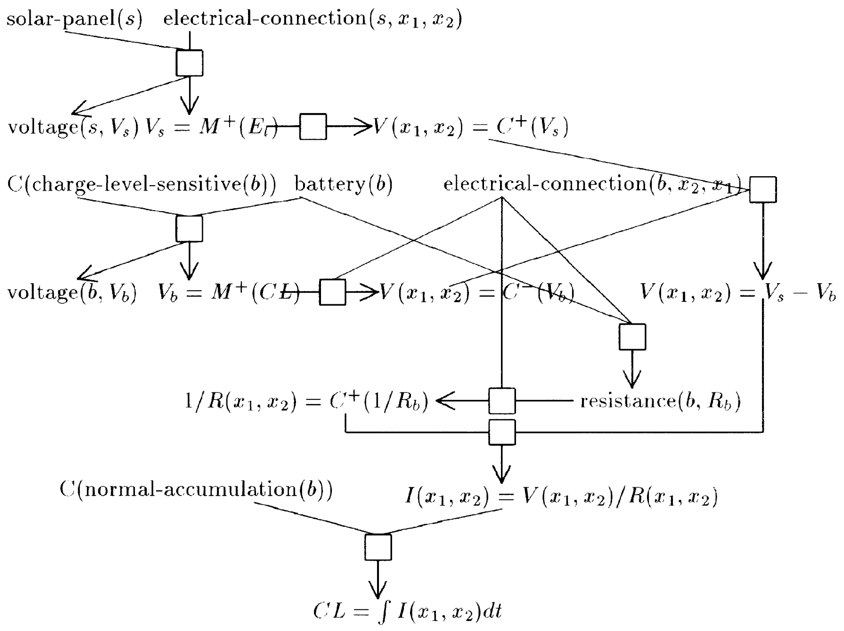

solar-panel(s)
electrical-connection(s, x_1, x_2)
voltage(s, V_s)
$V_s = M^+(E_l)$
$V(x_1, x_2) = C^+(V_s)$
C(charge-level-sensitive(b))
battery(b)
electrical-connection(b, x_2, x_1)
voltage(b, V_b)
$V_b = M^+(CL)$
$V(x_1, x_2) = C^-(V_b)$
$V(x_1, x_2) = V_s - V_b$
$1/R(x_1, x_2) = C^+(1/R_b)$
resistance(b, R_b)
C(normal-accumulation(b))
$I(x_1, x_2) = V(x_1, x_2)/R(x_1, x_2)$
$CL = \int I(x_1, x_2)dt$

Figure 7 Model inference example

$$V(x_1, x_2) = V_s - V_b \tag{2}$$

$$I(x_1, x_2) = \frac{V(x_1, x_2)}{R_b} \tag{3}$$

$$V_s = M^+(E_l) \tag{4}$$

The way in which the most appropriate assumptions and operating conditions are discovered is still an important research issue. Currently, compositional modellers tend to use techniques that are specific to a class of well-defined tasks. Therefore each approach should be examined with respect to the specific model construction task it tackles. Section 4 presents a survey of several important contributions to compositional modelling. In Section 5.1 the commonalities and differences between different approaches will then be analysed.

3.4 Summary

This section presented a general overview of compositional modellers, introduced the essential knowledge-representation formalisms used in compositional modelling and explained the type of model construction task that this approach attempts to solve. Compositional modelling relies on a knowledge base that contains alternative models of different components or processes of a system. It distinguishes between alternative models for the same component or process by means of operating conditions and assumptions. The main challenges of developing inference mechanisms for compositional modelling lie in finding ways to restrict attention to the most important components or processes of the system and to commit to consistent sets of operating conditions and assumptions such that the most appropriate model for the problem at hand can be derived.

4 A survey of compositional modelling approaches

The inference mechanisms of a compositional modeller search for consistent and complete models for a specific task description. Since most compositional modellers are designed with a specific model construction task in mind, there are considerable differences between the inference engines of specific compositional modellers. For example, some compositional modellers search for any model that meets basic consistency requirements (e.g. Farquhar, 1993) whereas others search for a sufficiently optimal, i.e. adequate, model (e.g. Rickel & Porter, 1997). In the latter case, the degree of optimisation of a model is dependent on the compositional modeller's notion of relevance. In this section, a number of important approaches to compositional modelling are examined and compared. All of these approaches have been implemented and tested on real-world or sample problems, as reported in the literature. This discussion will focus on the inference mechanisms, based on the specific implementation of the representational framework and the typical tasks that the target models are designed to accomplish.

4.1 CM: an ATMS-based compositional modeller

The compositional modeller presented in Falkenhainer and Forbus (1991) lends its name to the entire class of automated modellers that form the subject of this paper. This automated modeller is based on an approach to qualitative reasoning, called qualitative process theory (QPT). It extends the original work on GIZMO (Forbus, 1984) and QPE (Forbus, 1990). To avoid confusion between compositional modelling in general, and Falkenhainer and Forbus's latest extension of QPE, the latter will be identified as CM in the remainder of this paper.

The task CM is designed for is to construct a model of a particular subset of a scenario, to be identified based on a query, and to produce a total envisionment for that model. A total envisionment is a description of the behaviour of a (single) model under all possible operating conditions. CM uses three sets of algorithms to perform this task. The first represents the space of models that meet the structural conditions imposed by the scenario. The second identifies the subspace of models that are relevant to the query. The third searches for the simplest set of applicable model fragments. This set constitutes the output of QPE/CM since this compositional modeller produces total envisionments.

CM follows the general framework of Section 3 relatively closely, but its associated inference mechanisms are quite different from those employed by most compositional modellers. The scenario $M_s = \langle P_s, C_s \rangle$ is an instance of a component-connection formalism that allows "part-of" relations. These transitive part-of relations both imply that one component forms a subsystem of another and organise all components in a hierarchy. Before the actual problem-solving, this scenario is instantiated by the compositional modeller in its entirety. A model fragment in CM is a tuple $\langle P^s, P^t, C^s, C^o, C^m, C^t, A \rangle$ where P^s, P^t, C^s, C^o, C^m, C^t, and A follow definition 1 and C^m is a set of conditions on model fragment usage (e.g. "is applied"). The operating conditions $C^o(\mu)$ (the term has a slightly different meaning than in Falkenhainer & Forbus (1991)) are boundary conditions on variables and are not resolved by CM.

Assumptions are used in a slightly different way from other compositional modellers. Some assumptions are $\mathcal{C}(\text{exists}(p_i^s))$, p_i^s is a source-participant. Such assumptions represent the presumption that consideration of the source-participant is necessary to use the associated model fragment to model a query. Some assumptions that apply to source-participants are organised in a knowledge representation unique to CM: the assumption class. An assumption class is a tuple $\langle c, a_1, \ldots, a_n \rangle$ where c is a condition and the a_i, $i = 1 \ldots n$ represent individual assumptions. An assumption is logically equivalent to $c \rightarrow [(a_1 \vee \ldots \vee a_n) \wedge (\forall a_i, a_j, i \neq j, \neg a_i \vee \neg a_j)]$ (Falkenhainer & Forbus, 1991). This means that if the condition of the assumption class is true, then one and only one of its assumptions is true. The model fragment may also contain rules. Rules are typically used to instantiate the condition of an assumption class under well-specified circumstances or to constraint certain combinations of individual assumptions belonging to different assumption classes. Model fragments, assumption classes and rules are instantiated with respect to a scenario and stored in an assumption-based truth-maintenance system (ATMS).

The ATMS enables the compositional modeller to store economically all applicable model fragment instantiations simultaneously whilst maintaining consistency or inconsistency of all possible conjunctions of assumptions. All model fragments applicable to the scenario are represented in the ATMS. For each individual model fragment assumption, an ATMS assumption is instantiated. For each applicable model fragment μ an ATMS node is created for each different possible usage of μ (CM distinguishes between a model fragment that is applied with respect to the assumptions regarding source-participants and application with respect to all assumptions). These ATMS nodes are justified by ATMS assumptions and ATMS nodes (representing usages of model fragments) as is implied by the assumptions and model fragment usage conditions of these model fragments. Finally, an ATMS node is created for each target-participant and postcondition in each model fragment. The latter ATMS nodes are justified by an ATMS node indicating that the model fragment containing these target-participants and postconditions is applied. The resulting ATMS network enables CM to select a set of postconditions C_t and associated target-participants P_t by selecting a number of (ATMS) assumptions. The latter set of assumptions is called the modelling environment of $\langle P_t, C_t\rangle$.

Model selection in the compositional modeller is driven by the task specification, which consists of a query facility. A query is a question about specific properties of source- and/or target-participants. For example, a request to list the factors that influence the efficiency of a physical system's component consists of relating a number of energy inputs and outputs to and from that component. With the assistance of a domain-theory-specific and problem-solver-specific component, the query facility generates a set of expressions with respect to the instantiated domain theory. These expressions refer to a specific set of nodes in the ATMS that instantiates the domain theory with respect to the scenario. A node for the query is created with this set of expressions as its justification. The label of this query node is a set of candidate modelling environments.

These candidate environments imply consistent (guaranteed by the ATMS) but incomplete models and must therefore be extended. Additional target-participants and constraints are selected by adding assumptions to the candidate environments. First, all source-participants of relevance must be considered. The set of source-participants considered relevant by a candidate environment E_i is $P_s(E_i) = \{p_j \mid \exists \mathrm{e} \in E_i,\ e = C\}(\mathrm{exists}\}(\mathrm{p}_j))\}$. CM assumes that no component or subsystem (i.e. source-participant) can be modelled in isolation from other components or subsystems of the same covering system. A source-participant p is said to be a covering system of a set of source-participants P_s if $\forall$ $p_j \in P_s$, part-of(p_j, p). For each E_i, the smallest possible covering system of $P_s(E_i)$ and all source-participants that are part of it are added as assumptions to E_i.

Second, the partial candidate modelling environments produced by the previous computations are completed by applying domain-specific rules in order to account for all phenomena relevant to the completed object of the query. These domain-specific rules are the conditions of the assumption classes. A candidate modelling environment must contain an assumption from at least each assumption class whose condition is true as a result of the implications of this environment. This problem is resolved by the dynamic constraint satisfaction algorithm presented in Mittal and Falkenhainer (1990). The resulting candidate modelling environments are then ordered by means of some simple heuristics measuring their complexity. The model defined by the "simplest" candidate is used and validated. When inconsistencies are detected, an alternative candidate is chosen to define a new model, depending on the specific inconsistencies that occurred during previous trial runs.

The model produced by the compositional modeller is passed on to a simulator that produces a total envisionment of the model. One of the main problems with this compositional modeller and its predecessors is that these models consist of a set of instantiated constraints of the active model fragments. The representation of time employed by QPT does not require the resulting model fragments to hold at the boundaries of their operating conditions. Consequently, a variable continuously changing over time can cause the set of model fragments that holds at time instant t to be different from that which holds at $t-\varepsilon$ or $t+\varepsilon$. A model can therefore be valid for only a single instant of time. This prevents QPT from being expressed in terms of ODEs and gives rise to various problems with the associated qualitative simulator such as stutter: a sequence of situations which are represented by a model that is only valid for an infinitesimally small instant of time.

4.2 Qualitative physics compiler (QPC)

An alternative approach to compositional modelling is QPC (Crawford *et al.*, 1990; Farquhar, 1993, 1994). QPC's main task is to extrapolate the behaviour of the system described in a scenario, from a (possibly incomplete) set of initial conditions using a domain theory of model fragments. It aims at producing the *attainable envisionment* of a domain system from a certain state. An attainable envisionment is a description of all distinct sequences of states that could be observed from a system, from a limited set of possible initial states. To that end, a QPC task specification consists of a (partial) initial state of the system. QPC's goal contrasts with CM's purpose to find a *total envisionment* with respect to a given scenario and query. A scenario in QPC is specified by a set of source-participants and structural conditions which typically, though not necessarily, expresses a component-connection model.

The model fragments of a QPC model fragment library is a tuple $\mu = \langle P^s, P^t, C^s, C^o, C^t \rangle$. Note, that the model fragments do not contain a set of assumptions $A(\mu)$. Consequently, for a given structural setting, operating conditions are the only means by which to differentiate between models. In itself, this is not a major drawback since assumptions could be reformulated in terms of operating conditions. However, incompleteness of knowledge with respect to true operating conditions are not resolved. Instead, all possible models under operating conditions that are not disproved are constructed and simulated. This is achieved as follows. QPC first determines the set of model fragments that is applicable to the scenario. Model fragments that cannot be determined active or inactive are considered ambiguous. Suppose that there is a set $\Phi_{\text{ambiguous}}$ of n ambiguous model fragments. QPC resolves this by considering all so-called refinements of this set of model fragments. A refinement is computed as follows. QPC considers a way to partition the set of ambiguous model fragments in two sets Φ_{active} and Φ_{inactive} (there are 2^n ways of doing this and, in principle, all must be considered). Now, all ambiguous operating conditions of the model fragments in Φ_{active} are assumed to be true and one ambiguous operating condition of each model fragment in Φ_{inactive} must be assumed to be false (again, there is an exponential number of possible assignments). If one of these assignments is consistent with the unambiguous operating conditions, then the model resulting from this refinement must be considered and used for simulation.

This may seem a little excessive but the procedure tends to be less expensive than anticipated, as there are usually relatively few ambiguous conditions and the ambiguous operating conditions tend to be interdependent or even mutually contradictory. CM, on the other hand, resolves ambiguity by instantiating all possible, partial, candidate models into an ATMS. This approach involves the one-off cost of constructing the ATMS (disregarding the issue of model completion triggered by the query), the result of which is reused with each query. At any one time, CM searches for only one complete and coherent model. The selection procedures of the initial model deal with the relevance or irrelevance of the assumptions, which is more economical than reasoning with the full set of possible models. So far, CM is the only compositional modeller to use an ATMS and to simplify model selection to reasoning about the relevant assumptions.

The general architecture of QPC also differs from CM. In addition to a model processor responsible for model-fragment selection and instantiation, QPC also utilises an equation processor. The postconditions of the model fragments in QPC are QPT constraints as in the compositional modeller. A QPT constraint relates a small number of variables by means of one of a limited set of operators. Most types of QPT constraint formed in this way can be combined with others to form more complex mathematical relations, such as QSIM QDEs, and this makes them well suited for compositional modelling. However, as mentioned in Section 4.1, QPT constraints are not ideal for simulation. By having an equation processor to translate the set of QPT constraints (produced by the model processor) into QDEs, QPC resolves these problems. This translation is possible since, in this specific implementation, the influences are expected to hold at the limit of their operating conditions. The resulting QDEs can then be used for qualitative simulation via QSIM. This architecture is beneficial to QPC in at least two respects: it addresses issues, such as stutter, associated with the simulation in CM; it also allows QPC to make use of the latest improvements in qualitative simulation (and vice versa, of course).

4.3 Refining compositional modelling with quantitative information

The qualitative states considered by both CM and QPC tend to be very imprecise. This prevents them from exploiting whatever quantitative information that is available. However, quantitative information can enable modellers to focus the search for model fragments within narrower operating bounds. It also allows an associated simulator to produce more precise predictions which in turn narrow potential future operating conditions and enable more specific answers to the possible queries.

SIMGEN (Forbus & Falkenhainer, 1990) is a quantitative variant of QPC. Using a domain theory of fully specified numeric equations, it produces a total envisionment of the behaviour of a system when given a scenario and precise initial and boundary conditions. There are two obvious drawbacks, however, with this strict numeric approach. SIMGEN applications need to be able to construct a fully specified numeric procedure for every foreseeable combination of influences and this typically requires a large amount of model fragments. Also, it makes the system incapable of coping with imprecise knowledge.

Between these extremes lies the Semi-Quantitative Physics Compiler (SQPC) (Farquhar & Brajnik, 1994), a semi-quantitative extension to QPC. SQPC represents both the qualitative magnitudes of QPC and precise numeric intervals within the same framework. In so doing, SQPC can take advantage of quantitative information available in addition to information embedded in conventional qualitative QPC input. However, because the boundaries of the intervals are crisp and any value within an interval is treated equally, SQPC can not reason about the relative appropriateness of applied model fragments, nor can it give an indication of the relative likelihoods of different attainable behaviours.

4.4 Compositional modelling for generating causal explanations

In Nayak (1994, 1995) and Nayak and Joskowicz (1996) a compositional modeller is presented that searches for the simplest model of causal relations amongst the variables of a system which is sufficiently detailed to explain an expected behaviour of the system. Consequently, the task specification of this compositional modeller consists of an expected behaviour. This expected behaviour consists of a set of causal orientations between variables. The scenario is again a component-connection model. However, a rich representation may be used to express the source-participants (which model the components) and the structural conditions (which model the connections). This is possible because specific representation formalisms and inference techniques are available to process such knowledge. The target-participants of the model fragments are variables, and the postconditions are equations relating these variables to the allowed causal orientations between the variables within each equation.

The target model is the set of direct causal dependencies between variables (the target-participants) generated from the onto-causal mappings over the union of post-conditions of all selected model fragments. The onto-causal mappings are one-to-one functions from a set of equations to a set of causal orientations, such that all causal orientations are allowed by the model fragments and such that for each of the endogenous variables there is at least one equation which is causally oriented towards it. Such onto-causal mappings can be generated from a set of equations extracted from the postconditions of the selected model fragments, provided that the number of independent equations equals the number of endogenous variables, i.e. the set of equations is not overconstrained. In this case, the model defined by the set of selected model fragments is said to be complete.

The model fragments $\mu = \langle P^s, P^t, C^s, C^o, C^t, A \rangle$ represent classes of components of physical systems. Similar to object-oriented systems, these model fragments are organised in an inheritance hierarchy. A model fragment μ_i that specialises some other model fragment μ_j inherits all information contained in μ_i and may add to it, such that $P^s(\mu_j) \subseteq P^s(\mu_i) \wedge P^t(\mu_j) \subseteq P^t(\mu_i) \wedge C^s(\mu_j) \subseteq C^s(\mu_i) \wedge C^o(\mu_j) \subseteq C^o(\mu_i) \wedge C^t(\mu_j) \subseteq C^t(\mu_i) \wedge A(\mu_j) \subseteq A(\mu_i)$. Some of participants in the subclass μ_i may be physically the same as participants already in μ_j, but used for a different function. Such relations are modelled by so-called articulation rules. The components of the scenario are not necessarily instances of the classes of components described by the model fragments. The components represented by the model fragments are objects that perform a certain function (e.g. a magnetic coil) whereas the components of the scenario describe the physical appearance of the objects in the system (e.g. metallic wire coiled around a magnetic material). The structural conditions $C^s(\mu)$ represent preconditions on the structural setting that are required for a set of objects to be an instance of the component represented by the model fragment μ.

In addition to the structural conditions within model fragments, the knowledge base may also contain so-called structural coherence constraints. These are rules that impose a certain model fragment (or one of its subclasses) when given certain structural conditions. When these structural conditions are encountered, the structural coherence constraints prevent the compositional modeller from matching them with any of the superclasses of the imposed model fragment, thereby focusing the inference mechanisms. Similar to the aforementioned structural coherence constraints, the operating conditions $C^o(\mu)$ are enhanced by so-called behavioural coherence constraints. These are rules that impose a certain model fragment (or one of its subclasses) when given certain operating conditions, thereby preventing the inference mechanisms from searching through any of the superclasses of this imposed model model fragment.

In order to differentiate between different modelling perspectives, model fragments are also organised in so-called assumption classes (assumption class has a different meaning here than in CM). Here, an assumption class is defined by a set of binary approximation and contradictory relations between the model fragments involved. The *approximation* relation orders the model fragments with respect to their complexity. It is strictly domain-dependent and must, therefore, be imposed by the designer of the model fragment library. The approximation relation is irreflexive, anti-symmetric and transitive, and therefore it defines a partial ordering on a set of model fragments (Nayak, 1994). Such a partial ordering of model fragments is called an assumption class. Two model fragments are *contradictory* if their respective operating conditions or assumptions are inconsistent. This relation is also transitive. It is assumed that two model fragments are contradictory only if one is an approximation of the other. Hence all model fragments in an assumption class are contradictory with one another. Since the only source of inconsistency lies in the approximation relation, a model based on a set of model fragments Φ is inconsistent if and only if there exists an assumption class Φ^{AC} and two model fragments $\mu_i, \mu_j \in \Phi$ such that $\mu_i \in \Phi^{AC} \wedge \mu_j \in \Phi^{AC}$ (see definition 4).

The model fragment selection algorithm is based on a specific version of the approximation relation explained earlier, called causal approximation. A model fragment μ_i is an approximation of μ_j if $\langle P^s(\mu_i), C^o(\mu_i) \cup C^t(\mu_i)\rangle \subset P^s(\mu_j), C^o(\mu_j) \cup C^t(\mu_j)\rangle$. In other words, μ_i is an approximation of μ_j if μ_i contains a subset of the target-participants of μ_j and each constraint relating target-participants in μ_i has its equivalent in μ_j. It can be proven that, when a set of model fragments does not imply a consistent and complete causal model, a causally approximate set of model fragments does not imply a consistent and complete causal model either. This property is called the upward failure property and it enables a considerable reduction in the complexity of the search algorithm.

The search algorithm underlying this compositional modeller essentially works as follows. First, the most complex valid model is constructed. This is achieved by checking whether the structural conditions of the most complex model fragment of each assumption class matches the scenario and substituting the source-participants of the scenario if the match succeeds. The validity of the resulting is then checked. A model is valid if an onto-causal mapping can be constructed from it that matches the expected behaviour. An onto-causal mapping matches the expected behaviour if the causal relations implied by the expected behaviour are also implied by the onto-causal mapping. Additional constraints on the onto-causal mapping may be included in the operating conditions of certain model fragments. A subset of each $C^o(\mu)$ may consist of causal orderings between target-participants, representing the function of the component described by the model fragment. Next, the model constructed in this initial phase is subsequently simplified by searching through possible combinations of causal approximations of the originally selected model fragments. During this search, a set of model fragments Φ_1 defines a simpler model than a set of model fragments Φ_2, if $\forall \mu_i \in \Phi_1$, $\exists \mu_j \in \Phi_2$, $(\mu_i = \mu_j) \vee$ (causal-approximation(μ_i, μ_j)). From the simplest models found in this way, the final model may be selected using preference constraints or some other form of additional processing and selection.

In most cases, model fragment selection occurs under incomplete knowledge of the exact operating conditions. In order to determine the applicability of certain constraints, the values of certain variables are required. Yet the exact numeric values are usually not known or determined. Qualitative methods, such as those used by CM and QPC, deal only with the sign of the variables and therefore often do not use all the knowledge available. Order-of-magnitude reasoning (Raiman, 1991) combines principles of qualitative and numeric calculi as it focuses on the signs and relative magnitudes of the values. This compositional modeller represents and propagates values by means of the logarithm-based order of magnitude calculus NAPIER (Nayak, 1992). It allows improved precision in model fragment selection over qualitative methods without incurring the full computational expense of precise numeric methods.

4.5 Tailoring relevant influences for predictive and explanatory leverage (TRIPEL)

TRIPEL (Rickel & Porter, 1994, 1997) is a compositional modeller designed for predicting a system's behaviour with respect to certain variables of interest, under hypothetical operating conditions. The task specification of this compositional modeller consists of a set of variables of interest (target-participants) and a set of driving conditions. Driving conditions are a partial initial state description that may include a value or a rate of change for certain variables of interest.

TRIPEL's inference mechanisms then search for an adequate model. In the general framework of compositional modelling as given in Section 3, the inference mechanisms consist of searching adequate model fragments and the model is defined by the resulting set of model fragments. In TRIPEL, a model fragment $\mu = \langle P^s, P^t, C^s, C^o, c^t, A \rangle$ where c^t is a single QPT influence relating the target-participants P^t. Each element of $P^t(\mu)$ is a variable. The model fragment selection process considers the adequacy of each individual target-participant and influence in the emerging model rather than model fragments as a whole. The model fragments, however, contain part of the information on which adequacy of target-participants and influences are based, such as the operating conditions $C^o(\mu)$ and assumptions $A(\mu)$.

In TRIPEL, a model is considered to be *adequate* if no less complex model exists that contains influences and variables at the relevant level of detail. Influences and variables can be aggregated into less detailed alternatives and the permitted aggregation is defined in terms of the timescale of a problem. A *timescale* represents the relative speed at which phenomena occur, such as seconds, hours, months. Additionally, TRIPEL formalises what approximations, i.e. simplifications, are allowed using timescale information. This enables TRIPEL to resolve much ambiguity in model fragment selection by means of choosing the appropriate timescale for the problem. Timescale information is integrated in the model fragment library as follows. On the one hand, if $c^t(\mu)$ is a functional or indirect influence then it is considered instantaneous. On the other hand, if $c^t(\mu)$ is a differential or direct influence then it represents the effect of a process. In the latter case, there must be an assumption $a_i \in A(\mu)$ that represents the fastest timescale on which the effect of the differential influence is significant.

Other knowledge used in the determination of the adequacy of target-participants and influences with respect to the task description consists of aggregation hierarchies. An aggregation hierarchy of target-participants defines how sets of target-participant add up to another target-participant. Because target-participants in TRIPEL represent numeric properties of source-participants, this aggregation hierarchy forms a more detailed variant of the "part-of hierarchy" of the compositional modeller. In an aggregation hierarchy of influences, child-nodes represent the set of influences that explain the influence represented by their parent node. Each influence that is explained represents the same phenomenon as the set of influences that explain it. The explained influence, however, is more approximate or simpler.

For a given timescale of the problem, the aggregation hierarchies and time-scale information jointly determine the simplifications that are allowed. For example, the modeller may treat influences that occur on a slower timescale than that of the problem as insignificant. Influences that occur on a timescale that is faster than that of the problem may be replaced by the so-called quasi-static approximations (Iwasaki & Simon, 1994; Kuipers, 1987). Separate target-participants may be replaced by their aggregates when they equilibriate on a timescale at least as fast as the timescale of the problem itself.

The simplicity of a model is expressed by the total number of variables in the model. This notion of simplicity contrasts with that of most other compositional modellers. In Nayak (1994, 1995) and Nayak and Joskowicz (1996), for example, a model M_1 is simpler than a model M_2 if for each model fragment that constitutes M_1, M_2 contains a model fragment with equal or greater complexity or detail. Although this criterion may render model fragment selection more efficient, it is less generally applicable. The latter concept of simplicity cannot, for example, compare a model that represents a small subsystem in great detail with a model that represents a larger system containing that subsystem with little detail. Consequently, the latter criterion of simplicity limits the associated compositional modeller's ability to determine the appropriate scope of a system automatically.

The inference mechanism of TRIPEL performs a best-first search for the simplest model that is valid. Allowed simplifications are determined as described above. Validity is defined by a number of rules, called adequacy constraints. These constraints define which variables must be included in the model, which variables may be exogenous and which must be endogenous and so on. The simplest set of target-participants and influences that satisfies these adequacy constraints is considered the most adequate model. The presence of adequacy constraints may require considerable domain-context-dependent knowledge, of course.

4.6 Device modelling environment (DME)

DME (Iwasaki & Low, 1992) is a system that provides an environment for the design of physical, especially electro-mechanical, devices. DME is built on the compositional modeller described in (Iwasaki & Levy, 1994; Levy *et al.*, 1997}. The task specification in DME is an initial state description and a query that consists of a set of quantities whose values must be predicted, the set of target-participants that are considered exogenous (as opposed to TRIPEL, DME does not determine exogenous variables automatically), a set of assumptions that must hold in all simulated states and a set of constraints over target-participants that must be enforced. DME aims at constructing the simplest model that meets the requirements of the query and then simulating it from the initial state specification.

The model fragment library in DME is organised as follows. The structure of the model fragments closely follows the general framework of Section 3. However, model fragments are assumed to represent a type of physical system's component or physical phenomenon. Different model fragments representing the same component or phenomenon must have the same set of source-participants and structural conditions and are organised as follows. A Composite Model Fragment (CMF) Φ^{CMF} is a set of model fragments such that $\forall \mu_i, \mu_j \in \Phi^{CMF}$, $[P^s(\mu_i) = P^s(\mu_j) = P^s(\Phi^{CMF}] \wedge [C^s(\mu_i) = C^s(\mu_j) = C^s(\Phi^{CMF})] \wedge [A(\mu_i) = A(\mu_j) = A(\Phi^{CMF})]$. CMFs are DME's way of representing the same phenomenon under the same assumptions, but under different operating conditions. The operating conditions typically describe mutually exclusive ranges of variables. DME also assumes that operating conditions cannot be used to differentiate between operating conditions. As such, assumptions are used (similar to the $\mathcal{C}(\text{exists}(p_i^s))$ assumptions used in CM) to introduce new target-participants. Consequently, $\forall \mu_i, \mu_j \in \Phi^{CMF}, P^t(\mu_i) = P^t(\mu_j)$.

Different CMFs that represent the same physical component or phenomenon under different assumptions (including target-participants), are grouped in assumption classes (this is yet another definition of assumption class). More formally, an assumption class Φ^{AC} is a set of composite model fragments such that $\forall \Phi_i^{CMF}, \Phi_j^{CMF} \in \Phi^{AC}$, $[P^s(\Phi_i^{CMF} = P^s(\Phi_j^{CMF}] \wedge [C^s(\Phi_i^{CMF}) = C^s(\Phi_j^{CMF})]$. The CMFs are partially ordered with respect to the underlying assumptions. This ordering of CMFs is defined by a directed acyclic graph in which the links are labelled by sets of literals. These literals represent the differences in assumptions between CMFs. Positive (or negative) relevance literals represent the addition (or removal) of assumptions, with respect to source-participants or target-participants, when switching between model fragments in the direction of the associated arc.

Similar to the approach in Nayak (1994, 1995) and Nayak and Joskowicz (1996), DME's concept of model adequacy assumes a model fragment library in which replacing a model fragment by a more relevant model fragment involves replacing a relatively simpler model fragment by a more complex and detailed one. Within this framework, positive relevance literals denote addition of complexity and detail whilst negative relevance literals denote removal of complexity and detail. Other literals represent changes in assumptions, other than relevance claims, between model fragments of the same assumption class.

Model fragment selection is achieved by means of two procedures of relevance reasoning. The first determines what target-participants are relevant to the posed problem. For each of the initial variables in the query, the selection algorithm chains backwards through all assumption classes in which model fragments exist with equations (postconditions) that may causally explain the query variables. These causal explanations are produced by the postconditions, which are equations that define functional relations between variables. For each set of assumption classes that explains one of the query variables, the second procedure identifies the simplest CMF that is consistent with respect to the assumptions stated in the query and the assumption classes from which a CMF is already instantiated. This choice of assumption class and CMF must be revised each time a new CMF and associated assumption class is instantiated. The assumptions of each instantiated CMF are then added to the modelling assumptions of the current model. Its inputs, i.e. the variables of this CMF that have not yet been explained by the current model, are added to the backward-chaining queue of the first procedure. Finally, this CMF is added to the current model as a direct causal explanation for the variable that triggered its instantiation. As mentioned earlier, this procedure may have added modelling assumptions to the current model, and hence previously selected CMFs may no longer be valid. Therefore the first procedure discards any CMF whose assumptions are no longer valid from the model and replaces it with the next simplest CMF in the associated assumption class that is consistent with the rest of the model being constructed.

This backward-chaining mechanism performs the same function as the combination of the object completion algorithm in CM (Falkenhainer & Forbus, 1991), which selects the model components to be added to the scenario using a “part-of hierarchy” and uses domain-specific rules to complete the candidate modelling environments. However, DME presents a more uniform approach that relies directly on the domain theory and does not depend on any specific rules nor on the specific decomposition of the scenario. Nevertheless, the independence between assumptions and model fragments in CM allows for more flexibility in the modelling formalism.

Similar to the compositional modeller described in Nayak (1994, 1995) and Nayak and Joskowicz (1996), DME follows causal influence paths through the domain theory. Whereas the approach in Nayak (1994, 1995) and Nayak and Joskowicz (1996) first constructs the most complex model and simplifies it once it is found, DME maintains the simplest CMF of each assumption class and only adjusts it when necessary. DME’s method may, therefore, result in considerable savings when complex assumption classes are involved, although its criterion of simplicity is also similar to that in Nayak (1994, 1995) and Nayak and Joskowicz (1996) and has the same limitations as explained in Section 4.5.

DME is also similar to TRIPEL in that the model fragment selection procedure searches through the causal relations between target-participants covered by the model fragments. However, in TRIPEL, the knowledge used to determine the appropriate trade-off between detail and simplicity is formalised in different forms that are specific to the type of simplification. In this way, TRIPEL deals with aggregation hierarchies of influences and target-participants, timescale information associated with influences, and so on. This permits TRIPEL to use generic (at least with respect to a domain) meta-level knowledge to determine the allowed simplifications. In DME, all of this information must be contained within CMFs and assumption classes. Consequently, DME’s model fragment library must be more carefully constructed such that each consistent combination of model fragments can produce a model.

4.7 Model fragment selection and numeric uncertainty calculi

A compositional modeller, designed to generate explanations of industrial systems, for users with different levels of experience, is presented in Biris and Shen (1999). The modeller takes a scenario, a model fragment library, a set of components and processes of interest to the user and a metric of the desired level of detail as its inputs. The latter two inputs are produced by the explanation generator, which interfaces with the user, and by a user-modelling tool respectively. Model fragment selection is guided by a Bayesian network (Pearl, 1988) that relates all applicable model fragments to one another. This Bayesian network enables the modeller to deal with uncertainty about the level of detail of the selected model and its constituent fragments that best represent the user's needs and preferences. Hence this work illustrates how the symbolic methods of finding consistent models can be enriched with a numeric method to deal with uncertainty with respect to the appropriateness of individual model fragments (Parsons & Hunter, 1998).

The first step of model fragment selection consists of the construction of the Bayesian network. A Bayesian network is a directed acyclic graph. Each node in the network represents the selection or rejection of an individual model fragment instance. The structure of the network is based on the topological structure of the system being modelled. This way, each model fragment in this modeller represents a component. Different model fragments modelling the same component are grouped in assumption classes, strictly ordered within each assumption class with respect to their level of detail. When the outputs of a component are connected to the inputs of another component, their respective model fragments are linked to one another. Domain heuristics are utilised to assign probabilities to the root nodes and the links required by the modelling network. In principle, this approach has similarities with the compositional modellers discussed in Sections 4.4 and 4.6, except for the explicit use of numeric uncertainty to reason about model selection. Instead of reasoning with adequacy constraints which are derived from empirical knowledge, the work here applies formal default reasoning techniques to check for the consistency of the resulting selected model fragments and hence the composed system model. Solving the Bayesian network results in the most probable model fragment for each component with respect to the desired level of detail for the user.

4.8 Summary

The general principles of compositional modelling and the essential knowledge representation formalisms for such modelling were discussed in Section 3. This section explicates this framework with respect to a number of important approaches to compositional modelling. From this survey, it can be concluded that whilst the methods employed for knowledge representation and inference are very similar, the model fragment selection, model composition and model evaluation techniques can be very different. The differences are largely due to the variety of the types of problem specification, which depends on the problems the respective compositional modellers are aimed at. Indeed, tasks such as query answering, behaviour description at the appropriate level of abstraction, causal ordering and so on, require very different types of model and involve distinct sets of modelling choices. However, the underlying ideas of these compositional modellers may be extended beyond the individual implementations.

5 General issues regarding compositional modelling

Figure 3 categorises the model construction techniques involved in compositional modelling as inference, model fragment selection, model composition and model evaluation. The overall inference mechanisms commonly shared by different compositional modellers have been discussed in Section 3.3, and the other techniques that are specific to individual approaches have been discussed in Section 4. This section further examines the existing work on model fragment selection, model composition and model evaluation techniques and summarises the features of compositional modellers. It then provides a discussion about the strengths and weaknesses of compositional modelling. Finally, a proposal for a set of guidelines is given for compositional modelling design.

5.1 Summary of component techniques for compositional modelling

5.1.1 Model fragment selection

The inference mechanism of a compositional modeller identifies the applicable model fragments and instantiates them to a given scenario. From the set of instantiated model fragments constructed, a subset must be selected. This selection is based on the requirements imposed by the task specification and may be revised after model composition and model evaluation phases. The following techniques may aid in model fragment selection.

- *Assumption grouping*. Mutually inconsistent assumptions and operating conditions can be grouped in sets (Falkenhainer & Forbus, 1991). These sets then form domains of assumptions from which only a single one may be used to select the model fragments participating in the construction of the scenario model. Additional requirements or inconsistencies with respect to the scenario model are effectively formulated as constraints over these domains (Keppens & Shen, 2000) and constraint satisfaction techniques (Miguel & Shen, 1999) will aid in model composition.
- *Model fragment grouping*. Mutually inconsistent model fragments can be grouped in a directed graph (Levy *et al.*, 1997; Nayak & Joskowicz, 1996). These graphs are somewhat similar to a Graph of Models (see Section 2.2.3), but the nodes contain fragments of models, rather than entire models. From each network that is deemed relevant to the task specification at hand, a model fragment must be selected.
- *Property categorisation*. In certain problem domains, the conditions underlying the necessity of phenomena, processes and components and the appropriateness of assumptions and approximations can be expressed in terms of a limited number of properties. Compositional modellers using this feature define an ordered set of states that the properties can take. This way, each model fragment can be assigned a value for each of the properties under consideration, based on the phenomena, processes an so on. it describes. The model fragment selection method involves choosing the appropriate state for the properties. All model fragments that contain these states are then selected. For example, in TRIPEL (Rickel & Porter, 1997), the suitability of model fragments are expressed in terms of timescale (with values in terms of seconds or days, for example). Here, model fragment selection is based on selecting the appropriate timescale at which a problem should be considered.
- *Exhaustive selection*. Some compositional modellers, such as Farquhar (1993) do not aim at selecting a single best model. Instead, these approaches select all model fragments that meet the known conditions and compose them into all possible models.

5.1.2 Model composition

The aim of model composition is to combine the selected model fragments to form a consistent model. Model consistency can be defined as follows:

Definition 4 (*Model consistency*) A set of model fragments Φ defines a consistent model if $\forall \mu_i, \mu_j \in \Phi$: $P^t(\mu_i), C^o(\mu_i), C^t(\mu_i), A(\mu_i), P^t(\mu_j), C^o(\mu_j), C^t(\mu_j), A(\mu_j) \nvDash \bot$.

As definition 4 specifies, a model is consistent if its underlying assumptions, operating conditions and postconditions are consistent. Inconsistencies can be determined by generating the behaviour envisionment (Falkenhainer & Forbus, 1991) or by symbolically checking the model fragments constituting the model (Levy *et al.*, 1997). In the former case, a model is inconsistent if the envisionment is empty. In the latter case, the model composition mechanism needs to identify different (and hence inconsistent) perspectives that must be separated from one another. The following techniques may be used to maintain consistency.

- *Direct mapping*. The different modelling perspectives may be grouped per individual part of the scenario. The model fragments representing these different perspectives then have the exact same source-participants and structural conditions. Typically, such a group of model fragments describes a single component, when component-connection models are used to specify the scenario, or a single process (Levy *et al.*, 1997; Nayak & Joskowicz, 1996).

- *Truth maintenance system* (TMS). A TMS (de Kleer, 1986) can be used to record the dependencies of postconditions from operating conditions, assumptions and structural conditions. Inconsistent relations can be reported explicitly to a TMS, which will trace back its causes to the underlying assumptions (Falkenhainer & Forbus, 1991). Alternatively, the contents of a TMS can be translated to constraints over sets of assumptions and operating conditions and constraint satisfaction techniques can be used to guarantee consistent scenario models.

5.1.3 Model evaluation

The consistent scenario model or, in some cases, emerging scenario models need to be evaluated. Model evaluation aims at determining whether a scenario model built is adequate for the problem at hand. A model is adequate if it is sufficiently simple and sufficiently complete for use in the problem-solver.

There exist different techniques to measure or compare model simplicity (or its inverse, model complexity) and completeness. These are summarised below. Intuitively, simplicity and complexity seem inversely proportionate. Some compositional modellers, such as Nayak and Joskowicz (1996) and Levy *et al.* (1997), make this assumption explicitly. Therefore optimising model simplicity and completeness involve trading off both properties. Many compositional modellers search for the simplest model that meets a set of completeness requirements (Falkenhainer & Forbus, 1991). Other approaches restrict the scenario model's complexity and search for the most complete model (Keppens & Shen, 2000). Alternatively, a specific point in the spectrum of models between the very complete but complex and the very simple but incomplete may be searched that best fits the user's preferences (Biris & Shen, 1999).

Generally speaking, one model is simpler than another if it is easier to understand by a human and contains less extraneous detail. This depends on the number and nature of the variables involved and their relations. That is, it depends on the approximation of the number and nature of the phenomena, processes and components (to be described in the model). Unfortunately, computational complexity prohibits a metric that incorporates all possible features regarding the simplicity of a model. Nevertheless, the following techniques are frequently used to measure model simplicity or complexity.

- *Assumption rank*. When model fragment selection uses assumption grouping, model simplicity can be expressed by means of the rank of the employed assumption in their respective ordered sets. In Falkenhainer and Forbus (1991), for example, model complexity is defined as the sum of the ranks of the assumptions.
- *Model fragment order.* When model fragment selection is based on model fragment grouping, model simplicity can be expressed by means of the order of the model fragments in their respective ordered set.
- *Scenario model-based heuristic*. This approach applies a heuristic to the scenario model itself (e.g. by counting the total number of variables (Keppens & Shen, 2000; Rickel & Porter, 1997)) rather than evaluating the knowledge used in the model construction process. As explained in Section 4.5, metrics measuring the global properties of the emerging models are more general because they compare complete scenario models instead of performing a number of local comparisons that are difficult to combine.

A model is typically regarded to be more complete than others if it incorporates more phenomena, processes and components, with a description at a lower granularity, and uses variables and relations that better approximate actual behaviour. The following techniques are used to measure or compare model completeness in compositional modellers.

- *Task specification analysis*. Most compositional modellers involve some form of task specification analysis. A task specification, such as a query or an initial state specification, may contain a number of model variables or relations that can be matched with the target-participants and postconditions of prescribed model fragments. Otherwise, inference mechanisms or an interface may be provided to translate the task specification into a minimal set of target-participants and postconditions. This analysis provides the basis upon which larger scenario models are derived (Falkenhainer & Forbus, 1991).

- *Model fragment order.* This approach combines naturally with the model fragment order-based approach to calculate model simplicity. The combination is based upon the assumption that a more complete model is necessarily more complex, and that a simpler model is necessarily less complete.
- *Approximate reasoning*. Approximate-reasoning techniques can be used to compute an appropriate level of completeness. This approach requires techniques to assign completeness levels (e.g. probabilities or possibilities, depending on the employed approximate reasoning techniques) for the model fragments (e.g. based on model fragment order, as in Biris & Shen (1999)) and operators to combine these completeness levels. By means of such techniques, the completeness levels of the emerging models are compared with the most appropriate level computed and the closest model is retained.
- *Topological structure*. In compositional modelling of engineering systems, analysis of the topological structure of a device usually helps reveal what components should be considered with respect to the problem at hand. As explained in Section 4.1, CM utilises a hierarchical representation of an artefact topology, representing the way in which each subsystem is composed of smaller subsystems and components. Given a number of relevant components, heuristics (such as the minimal covering system) are applied to find the other relevant subsystems in the system.
- *Causal ordering*. Many problem-solvers require a model in order to derive values for unknown variables using known parameter values. If the task specification of a compositional modeller contains restrictions on what model variables may be utilised as independent variables and what else should be regarded as dependent ones, the causal ordering between model variables must be considered (Levy *et al.*, 1997; Nayak, 1994). For each candidate or emerging scenario model, a causal ordering between the model variables must be established for model evaluation. This can be achieved explicitly by the use of specific relations that impose or suggest possible causal orderings, or implicitly by means of certain inference mechanisms (Shen *et al.*, 1999). As long as an emerging scenario model does not meet all causal requirements, model fragments may have to be replaced if they are inconsistent with the causal requirements. On the other hand, model fragments may have to be added in an effort to complete the causal path from appropriate exogenous variables to prescribed endogenous variables.
- *Observation-based validation*. Currently, no compositional modellers use observations to validate models. In Nayak and Joskowicz (1996), model validation consists of checking whether the right model is constructed by comparing an expected behaviour specification with the behaviour that can be extrapolated from the composed model. However, there has been considerable work developed in the literature of model-based reasoning in general and model-based diagnosis in particular which addresses the issue of how observations obtained from a physical device can be used to validate and verify the model used (see Hamscher et al. (1992) and Shen & Leitch (1995) for further information).
- *Activity constraints*. Activity constraints specify conditions under which certain target-participants, assumptions and operating conditions must or must not be considered. Such constraints are formulated using domain-specific knowledge and may be represented explicitly in the model fragments (Mittal & Falkenhainer, 1990) or be derived from the structure of the knowledge base (Keppens & Shen, 2000). Dynamic constraint-satisfaction techniques provide an efficient tool for resolving systems of such constraints (Miguel & Shen, 1999).
- *Adequacy constraints*. The validity of a model may be determined by means of so-called adequacy constraints. These adequacy constraints are general rules that determine the conditions under which a model is valid using the properties of the scenario model (Rickel and Porter, 1997). For example, the rules may state when variables must be exogenous/endogenous, when certain phenomena/ processes must be included in the model, and so on. The properties may be the same as those used for property categorisation based model fragment selection. Adequacy constraints are not

necessarily enforced by means of constraint-satisfaction techniques. They represent general modelling principles that may be implicitly followed by any inference mechanism.

5.2 Strengths of compositional modelling approaches

Generality. The representations used in most compositional modellers do not necessarily limit the scope of potential domains to which the methodology can be applied. Although compositional modelling stems from research efforts in the domain of qualitative reasoning on physical systems, its application area need not be limited to physical systems or mathematical models. The basic concept behind compositional modelling consists of substituting a set of model fragments that comply with the format of definition 1. Although such generic inference mechanisms are not yet available, this premise applies to a wide variety of model construction problems. Virtually all research in compositional modelling to date applies to constructing mathematical models.

Some of the resulting compositional modellers are suitable for many different domains (Farquhar, 1993). They use model fragments to describe the generic processes and utilise no domain-specific representation formalisms. These processes are identifiable parts of a system that can be modelled by a number of variables and constraints over these variables. This notion of process is highly generic (more so than the notion of component) and most domain systems could be seen as being composed of such processes. In physical systems, the function of a component can be modelled as a process. The processes of macro-economics can be seen as activities that occur either by certain entities or because of certain entities. For example, if the government sector is considered in a macro-economic model, variables and constraints should be added to a model representing taxing of consumers and producers, subsidies, and so on (Dornbusch & Fischer, 1994). The same property can be observed in models of ecosystems, but here the entities are populations, stocks of resources, aggregate phenomena imposed on or caused by other entities (e.g. rainfall or air pollution) and so on (Heller & Struss, 1996).

Specificity. It is generally recognised that weak methods tend to be less efficient than their more specific counterparts. The compositional modelling approach is, however, sufficiently flexible to incorporate domain-specific assumptions and representations that may render it more efficient. The introduction of causal approximation in Nayak (1994) is a most notable illustration of this. As discussed in Section 4.4, a causal approximation is a specific kind of approximation relation between model fragments which imposes restrictions on the contents of a set of causally approximate model fragments. This approach makes three assumptions about the domain system and the problem at hand: all inconsistent model fragments can be partially ordered in a single network, the restrictions imposed by causal approximations are realistic and the resulting limitations on potential model simplifications do not compromise problem-solving. Yet, as is argued in Rickel and Porter (1997) these assumptions do not necessarily hold beyond the intended application domain. In cases where these assumptions hold, causal approximations allow considerably more efficient algorithms than more generic compositional modellers.

Rich representational framework. Recent developments in compositional modelling enable the designer of a model fragment library to organise domain knowledge in representational languages with features such as

- inheritance hierarchies (Nayak, 1995b),
- level-of-detail orderings on component models (Levy *et al.*, 1997; Nayak & Joskowicz, 1996; Rickel & Porter, 1997),
- postconditions that can be both discrete event rules and continuous functional relations (Iwasaki *et al.*, 1995),
- varying degrees of accuracy versus generality by taking advantage of quantitative and semi-quantitative information (Farquhar & Brajnik, 1994; Forbus & Falkenhainer, 1990; Nayak & Joskowicz, 1996) and
- incorporated timescale-based abstractions (Rickel & Porter, 1997).

Of course, tools must exist that can apply the resulting models to solving given problems. This is enabled by simultaneous developments in qualitative and semi-quantitative simulators – the predominant problem-solvers created and used in conjunction with compositional modellers. These research efforts improve the richness of the representational framework of compositional modelling and can express a great variety of aspects and problems of a carefully selected domain.

Composable domain theory. The knowledge base of a compositional modeller consists of composable fragments of theoretical knowledge regarding a domain. Specific applications of such knowledge need not be known at the time of implementation of the model fragment. Only the elements (e.g. system variables) the problem cases will consist of what must be known. In general, the range of different systems in most domains are constructed from a much smaller number of classes of system components or processes. Therefore, in theory, a reasonably small number of model fragments can model an exponentially larger range of systems. The component structure of the domain theory itself eases the maintenance and extension of this knowledge base, which is generally a difficult task in knowledge-based applications. In the domain of knowledge engineering, decomposition has been proposed as one means of simplifying this task. Central to compositional modelling is an already decomposed knowledge base. Decomposition allows component knowledge to be verified and validated independently from unrelated knowledge. In other words, model fragments can be adapted or added without much intervention in the remainder of the knowledge base.

Different perspectives with a single domain theory. A compositional modeller can compose a range of target models for a given scenario. It does not define a functional relation between the domain of scenarios and the domain of target models. Instead, a compositional modeller can use information on user preferences and abilities (Biris & Shen, 1999), explicit timescale specifications (Rickel & Porter, 1997) or explicit assumptions to guide its search for a suitable model. As such, compositional modelling provides an efficient and economical means of storing modelling knowledge of different types of perspective. When different types of perspective affect separate aspects of a system, they can be modelled by separate sets of model fragments. Model fragments of the different sets may be combined to produce a consistent and complete system model. As a result, the argument of the previous paragraph applies to perspective-taking. A relatively small model fragment library can be used to represent a large number of models representing different combinations of perspectives on the same system.

5.3 Major limitations and possible improvements

The knowledge-acquisition bottleneck. As mentioned earlier, the compositionality of compositional modelling allows for a reasonably small knowledge base to cover a large domain of systems. However, this feature does not solve, although it helps to reduce, *the knowledge acquisition problem*, commonly associated with knowledge based systems (Luger & Stubblefield, 1993). The use of compositional modelling avoids the need for the knowledge engineer to foresee all possible systems or subsystems that will be modelled. Yet such modellers rely on a complete and coherent description of model fragment classes. Each distinctive way of modelling each relevant model component must be described in the knowledge base.

Recent work on the Compositional Modelling Language (CML) (Bobrow *et al.*, 1996) proposes a unified syntax for defining ontologies and knowledge bases for compositional modellers. It is hoped that this work will help to relieve the knowledge-acquisition bottleneck by simplifying the exchange of ontologies and model fragment libraries between organisations and research groups that use different compositional modellers.

Incremental model construction. Most compositional modellers compose a new model from scratch for each problem specification. Most applications of compositional modelling-based systems, however, require different versions of the same model (or models with minor variations). For example, monitoring and diagnosis applications often require model-based reasoners to adapt their view of the system under consideration when changes in the system's normal description occur. Nevertheless, such adaptation is usually needed only for a small part of the overall system. The flexibility to “repair” an existing model in response to new requirements or slight alterations in the configuration of a system would be beneficial to many applications of compositional modelling. For this, recent developments in dynamic constraint-satisfaction techniques (Miguel & Shen, 1999a) may help to devise a principled means of incremental model construction.

Task specifications. A model fragment essentially states how a certain part of a model can be constructed under well-defined conditions. In theory, the models constructed by means of such a model fragment library are a declarative description of what is known about the system under consideration. In practice, however, the problem-solver tends to be inseparably linked to the compositional modeller. The compositional modellers discussed in Section 4 all perform a limited class of tasks. Only a few research efforts aim at expanding the types of applications that a single compositional modeller. One approach involves standardising the language that is used to represent model fragments, such as CML (Bobrow *et al.*, 1996). This work may form the foundation of a framework that simplifies the exchange of model fragment libraries between compositional modellers. Another consists of proposing a generalised framework within which different inference mechanisms fit. The work presented in Choueiry *et al.*, (1998) proposes using a planning system to select and apply different algorithms used in compositional modelling and problem-solving with respect to a given task. This work may promote a more generic use of compositional modelling with a single model fragment library for different problem-solving tasks.

Coherence and completeness assumptions. The knowledge acquisition problem for most compositional modellers is further complicated by imposing the coherence and completeness requirements on the model fragment library. Some compositional modellers alleviate this issue by verifying or validating each proposed model before passing it on to the problem-solver (Nayak & Joskowicz, 1996). This prevents inconsistent or invalid models from being used. A more principled approach to this problem consists of verifying and validating the model fragment library before it is used for model construction. However, such tools do not exist yet and form a research issue in their own right.

Model composition optimisation. Model fragment selection and composition is inherently intractable. However, in time-critical applications of model construction, such as monitoring, diagnosis and control, the model composition process should be optimised under time constraints. Only a few algorithms that solve specific problems have been proven to finish in polynomial time with respect to the number of model fragments. These algorithms achieve this by making strong assumptions about the way the model fragment library can be derived from the domain theory (e.g. causal approximations). In cases where the assumptions apply, such solutions are valid. Yet there is a need to find similar solution for applications in other domains where time imposes a constraint on problem-solving.

There is also a need to find inference mechanisms that improve the efficiency of more generic compositional modellers. Methods for caching the models, or the spaces of the models, of frequently used subsystems are needed in order to increase considerably the efficiency of model fragment selection. In fact, domain experts are known to reuse models of frequently used subsystems without reconstructing them from the basic components. For instance, an electrical engineer designing a circuit board will normally recognise a standard amplifier and know how this subsystem behaves without studying how the constituent components of an amplifier interact with one another. Any subsystem that is pre-modelled in terms of model fragments of the constituent components would impose an additional overhead to the model fragment selection algorithm as the interactions between these components must then be checked. Hence this extension should include additional inference mechanisms that can make informed assumptions with respect to which frequently occurring subsystems may be part of a given scenario.

Cost estimation and utility of model-design choices. Model fragment selection aims at constructing adequate models with respect to a problem-solving task. Model adequacy is usually defined in terms of a trade-off between simplicity and validity. Models must be sufficiently simple to avoid unnecessary detail that may distract the user or the problem-solver from their respective tasks. Models must also be sufficiently valid and include all elements and relations relevant to the task at hand. However, different compositional modellers have formalised this notion in different ways. A more general solution to this problem may involve explicitly incorporating a calculus of cost and utility of model complexity. Utility of model complexity is herein an instance of the micro-economic notion of utility as it represents the subjective needs and desires of the user. Utility is likely to reach a maximum at a certain level of complexity, whilst cost is likely to increase with complexity as complexity tends to increase the effort required for analysing and using the model. When both cost and utility are measured using the same dimension (e.g. in terms of number of variables) then more formal methods can be used to compare different models with respect to adequacy. In Rickel and Porter (1994, 1997) more principled methods of comparing different models are presented for a specific domain theory. Further research in this area may produce less domain-dependent methods for this purpose.

5.4 Preliminary guidelines for selecting the right compositional modeller

Since there are significant differences between compositional modellers, it is important to contemplate which compositional modelling techniques are most appropriate, when considering to apply them. This section summarises a number of features of the presented compositional modellers, that should be considered in this respect.

Table 1 compares the general setting of the compositional modellers discussed in Section 4. The typical domain denotes the problem area to which the respective compositional modellers were first applied. Although the presented compositional modellers are applicable beyond these domains, their underlying approaches tend to be inspired by the abstractions that underly modelling in the original problem domain. For example, TRIPEL differentiates between modelling alternatives based on timescale considerations since these constitute the basis of most abstractions in the domain of plant physiology. The exception to the rule is QPC, since this produces all modelling alternatives (differentiated only by operating regions) which the system under consideration can attain from an initial state. The next column in Table 1 denotes the nature of the scenario model that constitutes the output of the compositional modeller and the input to the problem-solver (see Figure 3). This is either a model designed for simulation or a causal ordering of a set of variables (typically used in explanation applications). Both have their typical uses in different types of problem-solvers. The third and fourth columns in Table 1 compare the task specification expected in addition to a scenario and the qualitative versus quantitative nature of the problem-solver with respect to the compositional modellers.

A number of important properties of the knowledge base of the different compositional modellers are compared in Tables 2 and 3. Table 2 summarises the typical content of a model fragment. Some compositional modellers (Nayak & Joskowicz, 1996) require that model fragments represent physical components because of the way the model fragment library is organised and used, as discussed earlier. This may improve the efficiency and effectiveness of applications to physical systems. Other approaches may have a more generic content, which cannot be used for domain-specific optimisations.

Table 1 Main features of some compositional modellers

Approach	Typical domain	Scenario model	Task spec.	Problem-solver
CM (Falkenhainer & Forbus, 1991)	thermodynamics	simulation	query	qualitative/numeric
QPC (Farquhar, 1993)	generic	simulation	state	(semi-)qualitative
Causal approx. (Nayak & Joskowicz, 1996)	electro-mechanics	causal order	behaviour	orders of magnitude
TRIPEL (Rickel & Porter, 1997)	plant physiology	causal order	state	qualitative
DME (Levy *et al*., 1997)	electro-mechanics	simulation	state + query	qualitative
Probabilistic (Biris & Shen, 1999)	factory processes	causal order	user specs.	qualitative

Table 2 Properties model fragments

Approach	Subject	Postconditions
CM	processes within components	set of QPT influences
QPC	processes	set of QPT influences
Causal approx.	components	equation + causal order
TRIPEL	process	single QPT influence
DME	components	composable relations + causal order
Probabilistic	components	influences + nature of variable

Table 3 summarises the typical features of the knowledge bases of the compositional modellers discussed in Section 4. Finally, Table 4 compares the model construction methods used by the different compositional modeller, using the terminology introduced in Section 5.

Given the highly diverse designs of compositional modellers, the presented methods could be adapted to meet the needs of various applications. At present, however, many applications of compositional modelling tend to use a single approach.

5.5 Summary

Sections 3 and 4 presented the generic CM representational formalisms and their associated inference mechanisms. However, significant differences exist between the techniques used for model fragment selection, model composition and model evaluation. An overview of these techniques has been given and, based on this discussion, the main properties of the compositional modellers reviewed have been summarised in this section. A number of guidelines for the use of compositional modelling techniques have also been proposed, based on an overview of advantages and disadvantages of compositional modelling.

6 Applications

The strengths of compositional modelling have appealed to researchers in a wide variety of problem domains. As the predominant use of compositional modelling is for typical tasks in the domain of

Table 3 Organisation of the knowledge base

Approach	Knowledge base properties
CM	Assumptions organised in sets of mutually exclusive assumptions Part-of hierarchy of components
QPC	No assumptions Additional taxonomy and relations between objects
Causal approx.	Model fragments are grouped per component Model fragments are grouped in class hierarchy
TRIPEL	Variables and influences are organised in aggregation hierarchy Timescale abstraction
DME	Model fragments are grouped per component with respect to assumptions Model fragments are further grouped with respect to operating conditions
Probabilistic	Approximate reasoning about user requirements

Table 4 Scenario model construction

Approach	Fragment selection	Simplicity	Completeness
CM	assumption grouping	assumption rank	task specification analysis, topological structure, activity constraints
QPC	exhaustive selection	N/A	task specification analysis
Causal approx.	model fragment grouping	model fragment order	task specification analysis, causal ordering, model validation, topological structure, model fragment order
TRIPEL	property categorisation	scenario model based heuristic	task specification analysis, adequacy constraints
DME	model fragment grouping	model fragment order	task specification analysis, causal ordering, topological structure, model fragment order
Probabilistic	model fragment grouping	as completeness	task specification analysis, approximate reasoning, adequacy constraints, causal ordering

modelling and reasoning about physical systems, this overview will focus on such applications. This section presents a brief overview of existing applications.

6.1 *Articulate virtual laboratories*

Many real-world problems involve the design of physical artefacts. For instance, educational training of engineering courses of required extensive design tasks. To evaluate a design, the behaviour of the artefact and economic properties must be analysed either by producing a simulation model or by producing a prototypical working device. Both options are very time-consuming and the latter may involve safety and cost issues. Virtual laboratories address these concerns to some extent. Such software products combine a design interface with a simulation engine and allow students to experiment with different designs. Virtual laboratories still impose significant limitations, though. In particular, they cannot assist in improving models or provide support for evaluating the quality of models simulated.

To build advanced virtual laboratories that may avoid these limitations, CyclePad, an articulate virtual laboratory (AVL), has been proposed recently (Forbus & Whalley, 1994; Forbus *et al.*, 1999). It is an intelligent tutoring program for the support of teaching design of thermodynamic cycles. The architecture of this program is shown in Figure 8. Briefly, the *interface manager* takes care of any communication with the user, providing a CAD-style interface. The essential functionality of CyclePad can be divided into three modules: the *analysis module*, the *explanation module* and the *coaching module*. All of them are dependent on CM, which provides services that enable this AVL.

A design should be created with specific behavioural features in mind, for example minimal output of work, minimal efficiency, maximum operating cost and so on. However, substantial analysis is required to compute such information. The analysis module automates this aspect of the design effort, thereby allowing students to focus on the design task itself and facilitating consideration of alternative systems. In particular, CyclePad's design process proceeds through two modes. In the build mode, a scenario is constructed, and in the analyse mode, modelling assumptions are made and parameter values chosen. When a complete and consistent set of assumptions and parameter values are assigned, the compositional modeller is applied to construct a model of the system under design of the set of mathematical equations. Choosing this complete and consistent set of assumptions and parameter values is a complex problem. Certain assumptions and parameter values may be inconsistent with one another. Some assumptions may cause other assumptions or parameters to be considered or removed. These inconsistencies and dependencies amongst assumptions and operating conditions are stored automatically by CM algorithms in the TMS (see 4.1).

CyclePad can validate any modelling assumptions or parameter values chosen by the user using the information contained in the TMS. In fact, the coaching module provides contradiction reference

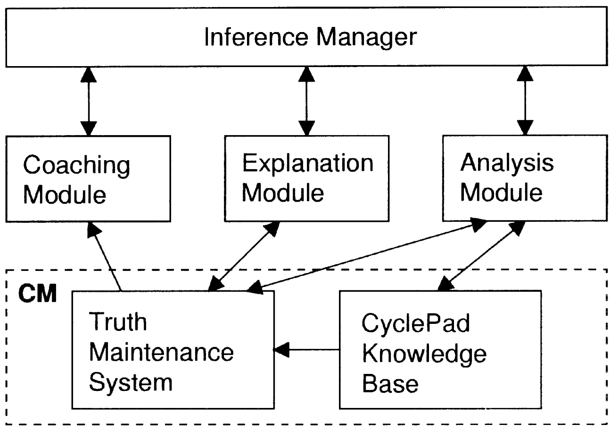


Figure 8 CyclePad's architecture (adapted from Forbus *et al.* (1999))

facilities to help students understand them in their proper context. Different types of inconsistency are distinguished according to the syntax of the involved assumptions. For each different type of assumption, a specific interface is presented to show the problem in an appropriate fashion. For example, if a pair of two parameter values is over a certain pre-set threshold (e.g. temperature and pressure), then a two-dimensional graph is shown containing all allowed combinations under the current modelling environment and a point that represents the parameter values under consideration.

Once a model of mathematical equations is available, evaluating the extent to which a design meets its requirements is possible. To this end, CyclePad extends CM with a constraint propagator (Abelson & Sussman, 1985; Davis, 1987; Weld & de Kleer, 1990) and a numerical sensitivity analysis algorithm (both contained in the analysis module of Figure 8). The former is used to derive the values of certain dependent variables given the equations and the parameter values. As such, it calculates specific properties of the system, such as work output, efficiency or operating costs. The latter is used to explore how sensitive a design is to changes in parameter values by means of conventional "what-if" analyses.

Graphs describing how significantly pairs of properties, for example pressure and volume, evolve throughout a cycle are an important and powerful means of understanding how a thermodynamic device works. The constraint propagator allows for the calculation of the necessary values of the properties of interest at the significant points within a cycle. However, domain-specific knowledge is required to compute the transitions between those points. For example, as heat is applied to a body of water the resulting temperature increase follows an isobar below the saturation point of water. Then, at the boiling point, the temperature remains steady until the water is completely vapourised and behaves as a gas. The distinct phases in a transition are modelled as operating conditions in model fragments and are derived from the TMS.

The *explanation module* provides the means to ask and answer common questions on a design and interfaces with the query-answering facility of CM. The topics on which questions can be asked – all things considered by the compositional modeller – are called *e-propositions*. E-propositions are of predefined types, e.g. a turbine in CM is a "part-proposition" and the temperature of the output of that turbine in CM is an "nvalue-proposition".

The actual explanations are called *e-arguments*, which are of predefined types as well, depending on the way they are derived by CM. For example, if the aforementioned temperature followed from an equation, the associated e-argument would be of the type "numeric-value-via-equation". The e-arguments contain information on references to other e-propositions (e.g. those associated with the parameters in the equation) that were used in deriving them. Such information follows directly from the TMS employed by the CM and from the way equations are used in the constraint propagator. Explanations generated are presented in the form of hypertext, so that the user can easily click on the different elements of an explanation (e.g. a parameter, an equation or an assumption) and ask some question about it.

CyclePad is currently being used as an educational tool in courses on thermodynamics engineering at universities and institutes of higher education around the world. Some schools have developed new curricula around this AVL (Baher, 1998a, 1998b) and the US Naval Academy reported that it allowed students to tackle more complex projects, some of which resulted in publishable technical papers (Wu & Burke, 1998; Wu & Dieguez, 1998).

6.2 Self-explanatory simulation

Simulation is the reproduction of a system's behaviour by recreating the underlying mechanisms (e.g. via a mathematical model). This way of analysing systems is often less expensive than building an artefact. Also, it allows the conduct of experiments in carefully designed and controlled environments (which may not be feasible in the real world). In conventional simulation environments, however, considerable effort is still required for the construction of simulation models and the translation of numeric simulation results into possible actual system behaviour. The reasons that a simulation model may behave in a certain way typically require further explanation or second-guessing by the user.

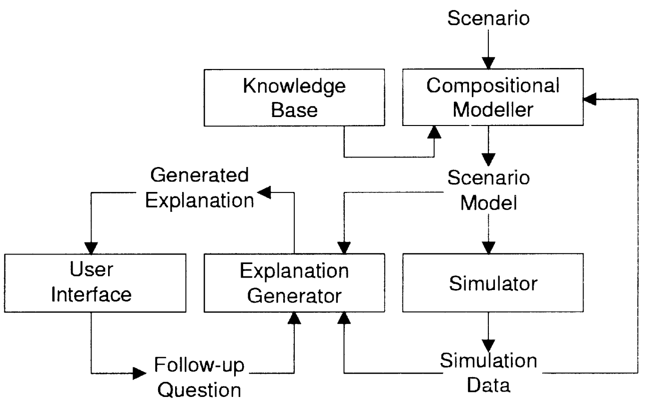


Figure 9 Architecture of a self-explanatory simulator

Self-explanatory simulators address these issues. They attempt to automate the construction and reconstruction (under changing conditions or operating modes) of a simulation model and provide explanations about the model and simulation results. A reference architecture for the so-called collaborative device modelling environment (Gruber & Gautier, 1993; Gruber *et al.*, 1996) that allows for self-explanation simulation is given in Figure 9.

CDME is based on the compositional modeller constructed for the device modelling environment, which was discussed in Section 4.6. The user must provide a scenario by composing a component-connection model of the device using standardised components. In the model fragment library, model fragments are grouped per matching generic component type and also per set of modelling assumptions. The compositional modeller provides techniques for finding the simplest consistent model of a given system and by selecting the appropriate assumptions and operating conditions (if these are unknown). The scenario model is passed onto the simulator. The resulting simulated behaviour is fed back to the compositional modeller which monitors whether all operating conditions continue to hold and updates the model if necessary. In this way, the complexity of simulation model construction is significantly reduced, as the user can reason in terms of components and the task of device topology, and model revision due to changing system states can be fully automated.

The explanation generator developed in this work can answer the following types of question.

- *What is this component? What is the definition of this parameter, relation, component or operating mode?* CDME can define a component, list its parameters (used for modelling it) and operating modes, and define these. The definitions are extracted from a library of definitions. The related parameters and operating modes are derived from the instantiated model fragment library. The parameters are the target-participants of the model fragment chosen to describe the component. Each set of assumptions that could be used in modelling the component specifies an operating mode.
- *What are the initial conditions of this simulation scenario?* These are provided as part of the task specification and are described in a coherent text via partial sentences that describe the parameters and operating modes involved.
- *What happened in this state? What just happened?* States are identified using conventional qualitative reasoning techniques (Kuipers, 1994). Basically, whenever a model fragment is replaced by another due to changes in assumptions or operating conditions, a new state is generated. As such, the explanation system can return all changes underlying a state change.
- *What caused this event? What caused this change in behaviour? What influences this parameter?* The causes of discrete state changes can be explained by listing the changes in assumptions or operating conditions that led to the change. The causes of continuous changes, e.g. changes of certain variables, are explained by means of the causal influences that are constructed from the model fragment postconditions during model construction (see 4.6). The resulting causal ordering forms the chain of causal links from which the explanation is constructed.

In CDME, CM is also used to develop a technique called *compositional text generation* that enables the creation of explanatory text. The model fragments in CDME are annotated with phases that define and identify the parameters, components and operating modes. For example, the operating mode “working normally” of a component named “Thruster *x*” can be composed into a sentence “In state *y*, thruster *x* is working normally”. Furthermore, the names of subcomponents represent how the parts can be composed in a larger subsystem. For instance, the parameter “pressure of” that describes the component “intake terminal” which is a subsystem of “thruster *x*” can be described as “the pressure of the intake terminal of thruster *x*”. As such, partial phrases representing the behaviour of model fragments can be composed into meaningful descriptions of the behaviour of larger subsystems.

Self-explanatory simulators, are of important potential for the following.

- *Requirements documentation*. The intended behaviour of a system can be specified by showing it in action and explaining the underlying mechanisms.
- *Specification of operating procedures*. These describe the actions that must be undertaken under well-defined conditions and configurations of a system. With CDME, for example, the actual conditions and configurations can be specified formally with reference to a model and the expected consequences of its operating procedures.
- *Knowledge-sharing and reuse in collaborative design*. Instead of creating models individually, design decisions can be evaluated by means of a single formal knowledge base and the underlying model can be generated automatically. Additionally, the underlying assumptions, the presumed operating procedure and the resulting behaviour can be evaluated formally (Iwasaki *et al.*, 1997).

6.3 Diagnosis

In its most general form, a diagnostic system determines the actual configuration of a system and establishes how this configuration differs from an appropriate one. Model-based diagnosis (Hamscher *et al.*, 1992) is a research area in which techniques are developed to predict a system’s behaviour from a technical-level description, often called the structure of the system, and to trace inconsistencies with the observed behaviour back to the technical-level model. GDE (General Diagnostic Engine) (de Kleer & Williams, 1987) and GDE + (de Kleer & Williams, 1989), for example, predict system behaviour from a component-connection model and known parameters and traces inconsistencies back to faulty components or predefined fault modes respectively.

Being an approach aimed at deriving mathematical models from a technical-level scenario, compositional modelling is a suitable approach to aid in model-based diagnosis. Hence, several compositional modelling-based approaches to diagnosis have been developed, such as those presented in Biswas and Yu (1993), Xia *et al.* (1993) and Struss and Heller (1998). The next subsections discuss conventional diagnostic applications of physical systems and diagnosis of ecological systems respectively.

6.3.1 Diagnosis of physical systems

A number of diagnostic engines based on the use of composition modelling involve the traditional problem domain of engineering. For instance, component-oriented diagnosis (Biswas & Yu, 1993) aims at diagnosing complex industrial systems that may be more easily described by component-connection models than by mathematical models. Most such diagnostic engines require qualitative or analytic models of the system being diagnosed, in which the variables are explicitly linked to the actual system components (Biswas *et al.*, 1997; Mosterman *et al.*, 1995). A compositional modeller within such a diagnostic engine constructs a mathematical model of the system, using the given component-connection model, the available information about assumptions on the system’s mode of operation and the obtained observations. During the model construction process, the relation between variables and components can be explicitly maintained by means of the information contained in the model fragments. A simulator can then predict system behaviour under the presumed conditions. The diagnostic engine compares the model-based behavioural prediction with the available observations and then either generates fault hypotheses or request additional observations.

The Automated Intelligent Modeller for Diagnosis (AIMD) (Xia *et al.*, 1993) is also a diagnoser that uses compositional modelling to construct a model and thereafter to predict system behaviour. AIMD is based on the compositional modeller presented in Ahriz and Xia (1997) and Ahriz and Tomasena (1999), in which postconditions are formulated in terms of bond graphs. The partial bond graphs in AIMD are annotated with predefined energy-domain specifications. Additionally, AIMD allows particular assumptions to be linked to their corresponding bond graph node type and added to a model independently of one another, depending on what assumptions are appropriate. For example, friction on the shaft of a motor may be associated with a resistance node. This use of partial bond graphs is similar to the use of causal influences in other compositional modellers. However, in the bond graph formalism, the number of concepts and the compositionality of these concepts are more restricted. This provides additional focus earlier in model construction that can potentially improve efficiency. On the downside, bond graphs do not possess the versatility of mathematical equations. It can therefore be difficult for systems like AIMD to be extended to cope with problems that require the explicit exhibition of the behaviour of the system under consideration.

6.3.2 Diagnosis of ecological systems

Not all systems of interest are typically described by means of a topology of subsystems that should perform a well-understood function. Appropriate descriptions of ecological systems typically involve process that occur due to the combined presence of a number of prerequisite elements under certain conditions or constraints. For example, the process of ozone formation in the lower atmosphere may occur as a result of high solar radiation and the presence of certain substances. Hence conventional approaches of diagnosis are not suitable in this domain.

Process-oriented diagnosis (Struss & Heller, 1998, 1999), which has been applied to ecological systems, requires a system, containing a scenario, i.e. a description of the structure of the ecological system, quantity specifications for the known parameters and a domain theory (which mainly consists of model fragments). As shown in Figure 10, the initial scenario is translated by a compositional modeller into a conceptual representation (more specifically, qualitative influence diagrams (QID)) as discussed in Section 2.1.2. As with any compositional modeller, this involves the making of modelling assumptions. The resulting QIDs are then translated into constraints by applying closed-world assumptions to the sets of influences that causally affect the same variable. These constraints, combined with the quantity specifications for certain parameters, are used to extrapolate the system behaviour.

If this system behaviour is inconsistent with given observations, diagnostic candidates (i.e. minimal sets of assumptions that may contradict the observations) are generated to explain the discrepancy between actual observations and model predictions. These candidates consist of modelling assumptions and closed-world assumptions (de Kleer & Brown, 1984). Candidates are tested by

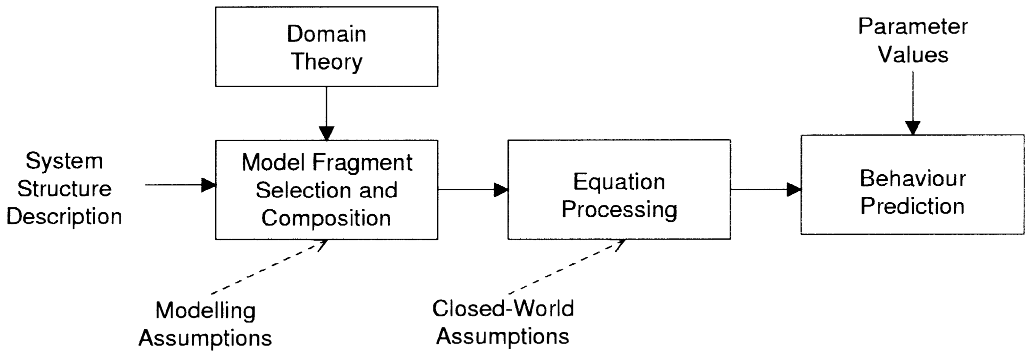


Figure 10 General architecture of process-oriented diagnosis (Heller & Struss, 1998)

creating a new model without them, by retracting all the assumptions they contain. Modelling assumptions are retracted by searching for a consistent scenario model that is not based on a superset of the contradictory assumptions. A closed-world assumption is false only if an unanticipated influence affects the respective variable. Therefore a model fragment must be found that introduces such an additional influence and a new closed-world assumption must be formulated to construct the new constraint. This search can be focused with respect to the model fragments that causally affect the variable in question. An ATMS is used to trace which new assumptions must be added to and which assumptions must be removed from the model.

This approach to diagnosis is more general than many others because it can identify new elements, external to those expected to be part of the system, that cause the unwanted behaviour, provided that model fragments are defined in the domain theory that describes the effect of these external elements. For this reason, the approach is potentially suitable for diagnosing ecological systems (Struss, 1998). An ecological system does not consist of components that malfunction or subsystems that operate according to a specific failure mode. Instead, there usually are disturbances, i.e. unanticipated external influences that interrupt the natural equilibrium, that account for the unwanted behaviour. If present in the domain theory, this diagnostic engine can help identify candidate sets of such disturbances.

6.3.3 Monitoring and control

An architecture is proposed in (McIlraith *et al.*, 1998) for adaptive monitoring and control of a dynamic configuration of multiple autonomous component-systems that need to cooperate to perform a set of joint functions. The specific requirements of this task are not met by traditional controllers because the system under control is dynamic in nature. Over time, upgrades and repairs may cause certain component-systems to be replaced by different systems. In the long term, the configuration of the system may change and new functions may be added to and old functions removed from the requirements list.

Within this architecture, the compositional modeller maintains a valid model of the system under control. The input to the compositional modeller consists of a formal high-level scenario of the system. The model fragment library is designed to construct a mathematical model for the controller. When new component-systems are added, when old ones are removed, or whenever the relations between the component-systems change (i.e. the configuration of the system changes) the compositional modeller computes a new mathematical model based on the altered scenario. Because the knowledge base of a compositional modeller consists of small reusable model fragments, new types of component-systems can be represented by new combinations of model fragments. Changes in technology can be propagated to the knowledge base by defining new model fragments. In short, the compositional modeller enables the adaptivity of monitoring and control by automating the process of model construction whenever the system under control changes.

6.4 Socio-economic systems

Applications of compositional modelling beyond the physical domains are very scarce. However, compositional modellers are sufficiently flexible to be adapted to construct a domain theory of various socio-economic or ecological systems. This is supported by the fact that qualitative and semi-quantitative simulators offer an excellent means to deal with incomplete knowledge on the magnitudes of socio-economic parameters and the relations that exist between them (Scott *et al.*, 1995; Wyatt *et al.*, 1995).

A proposal for using compositional modelling techniques to represent and reason about enterprises is given in Wang (1997). This approach views organisations as networks of interdependent business processes. Model fragments define such processes with respect to underlying structural and operational conditions and assumptions. A given scenario defines the structure by which business processes are interconnected. The specific processes that apply to a situation depend on the specific assumptions and operating conditions that apply, whilst the latter may change dynamically as the organisation changes.

Hence some of the compositional modelling techniques discussed in Falkenhainer and Forbus (1991) fit this profile.

SQPC has also been applied to modelling macro-level socio-economic problems (Brajnik & Lines, 1998). A taxonomy consisting of objects in terms of societies, environments, populations, activities and so on, and the relations between these, is defined in this work. Built upon this taxonomy, model fragments are created to represent influences that occur under certain conditions between attributes of such objects. For example, certain harmful emissions may occur in an environment as societies perform industrial activities. By applying the domain theory to slightly differing scenarios and under different initial states, this work allows performing what-if analyses without a complex redesign of the system model.

6.5 *Summary*

This section has discussed a number of influential applications of compositional modelling. In particular, in intelligent tutoring environments that involve model construction, the application of CM allows building systems capable of showing and explaining to students what decisions must be taken and how particular, seemingly independent, choices are in fact interrelated. In explanation systems, compositional modelling enables the trace of particular events and states of a system back to underlying operating modes and modelling assumptions. In model-based diagnostic applications, compositional modelling allows one to search for fault hypotheses other than failure modes; more specifically, unanticipated disturbances can be identified as external causes to unexpected behaviour. There have been many more other applications of CM reported in the literature. This section has only provided an overview of the most influential as representatives of such applications in an effort to demonstrate the utility of the compositional modelling techniques.

7 Conclusion

This paper has presented an overview of compositional modelling, a knowledge-based methodology for system modelling that has established itself as a leading approach. The knowledge base of a compositional modeller consists of composable pieces of knowledge, termed model fragments, representing phenomena that occur as results of domain processes or components. By selecting and composing sets of model fragments, based on a given scenario and a task specification, compositional modellers can model any system composed of elements contained in its model fragment library.

Different types of algorithm that automate model construction exist. This paper has identified the most influential automated modellers and classified them along two dimensions. The first dimension indicates the different representation formalisms, which are used for computations with abstract levels of knowledge, e.g. component-connection models, qualitative differential equations and so on. The second dimension categorises automated modellers as deductive, inductive or hybrid (i.e. combining elements of both inductive and deductive reasoning). This classification forms a basis upon which to analyse the strengths and weaknesses of individual approaches to automated model construction, including compositional modelling, of course. In particular, compositional modellers typically produce mathematical models from some symbolic representation of the different entities in the system and of the relations between these entities. They tend to perform inferences in a deductive or hybrid manner.

A formal framework for compositional modelling has been presented. This framework introduces, and generalises, the common features of most compositional modellers. The focus of this discussion lies in the knowledge representation issues related to the compositional modeller, since the inference mechanisms of individual compositional modellers to date are highly specific and depend on their own specialised knowledge base. Within this framework, a number of important compositional modellers were described, analysed and compared. The relation between individual representations and the more generic formalism is explained, in particular with respect to the specific content of model fragments and how they are organised in the model fragment library. An overview of the specific inference mechanisms of each compositional modeller is also presented.

Further, important strengths, limitations and possible improvements of existing compositional modellers have been identified. Compositional modelling provides a generic framework for assisting problem-solvers by constructing an adequate model for the problem-solver to work with. Yet it is capable of taking advantage of knowledge in more problem-specific types of representation formalism, thus enabling the use of a rich representational language for the knowledge base. Compositional modellers are typically able to generate a wide variety of models that meet the requirements of a given scenario and then to choose one of these models with respect to other requirements provided by the task specification.

Knowledge acquisition proves to be the main limitation of compositional modelling. However, because the knowledge base consists of generic fragments, the size of the knowledge base need not be linear with the number of types of problem specification that can be presented to it. For a suitable problem domain, this number may favourably increase in an exponential way with respect to the size of the knowledge base, owing to the composability of model fragments. Nevertheless, possible improvements to compositional modelling exist, including inference mechanisms for local adaption of models in response to changing requirements, verification and validation tools for model fragment libraries, new inference mechanisms to optimise the selection between alternative model fragments that apply to the same scenario subsystem and hence the model composition.

Compositional modelling is a suitable methodology for a wide range of problems, such as explanation and tutoring, design, prediction, diagnosis and monitoring, and control. Most applications are relevant to one or more subdomains of physics and engineering. However, compositional modelling can be sufficiently adaptive to model socio-economic systems successfully, using a process-oriented approach. Future research in compositional modelling may improve the field both in breadth and in depth.

References

Abelson, H and Sussman, GJ, 1985, *Structure and Interpretation of Computer Programs* MIT Press/McGraw-Hill.

Addanki, S, Cremonini, R and Penberthy, JS, 1991, “Graph of models” *Artificial Intelligence* **51** 145–177.

Ahriz, H and Tomasena, M, 1999, “Modeling as a fragment assembling process” *Proceedings of the 13th International Workshop on Qualitative Reasoning about Physical Systems* 1–10.

Ahriz, H and Xia, S, 1997 “Automatic modelling for diagnosis” *Proceedings of the 11th International Workshop on Qualitative Reasoning about Physical Systems* 3–12.

Amarel, S, 1968, “On representations of problems of reasoning about actions” *Machine Intelligence* **3** 131–171.

Amsterdam, J, 1992, “Automated qualitative modeling of dynamic physical systems” Ph.D. Dissertation, MIT.

Baher, J, 1998a, “How articulate virtual labs can help in thermodynamics education: a multiple case study” *Proceedings of the Frontiers in Education 1998 Conference*.

Baher, J, 1998b, “Integrating ‘articulate educational software’ into engineering courses: a formative evaluation” *Procceedings of the Annual Meeting of the American Educational Research Association*.

Biris, E and Shen, Q, 1999, “Automatic modelling using Bayesian networks for explanation generation” *Proceedings of the 13th International Workshop on Qualitative Reasoning about Physical Systems* 19–26.

Biswas, G and Yu, X, 1993, “A formal modeling scheme for continuous systems: focus on diagnosis”, Proceedings of the 13th International Joint Conference on Artificial Intelligence 1474–1479.

Biswas, G, Kapadia, R and Yu, XW, 1997, “Combined qualitative-quantitative steady state diagnosis of continuous-valued systems” *IEEE Transactions on Systems, Man and Cybernetics* **27**(2) 167–185.

Biswas, G, Manganaris, S and Yu, X, 1993, “Extending component connection modeling for analyzing complex physical systems *IEEE Expert* **8**(1) 48–57.

Bobrow, D, Falkenhainer, B, Farquhar, A, Fikes, R, Forbus, K, Gruber, T, Iwasaki, Y and Kuipers, BJ, 1996, “A compositional modeling language” Proceedings of the 10th International Workshop on Qualitative Reasoning about Physical Systems 12–21.

Bradley, E and Stolle, R, 1996, “Automatic construction of accurate models of physical systems” *Annals of Mathematics and Artificial Intelligence* **17** 1–28.

Brajnik, G and Lines, M, 1998, “Qualitative modeling and simulation of socio-economic phenomena” *Journal of Artificial Societies and Social Simulation* **1** citeseer.nj.nec.com/brajnik97qualitative.html

Breedveld, PC, 1984, “Physical systems theory in terms of bond graphs”, Ph.D. Dissertation, Twente University.

Breunese, A, Top, JL, Broenink, JF and Akkermans, H, 1998, “Libraries of reusable models: theory and application” *Simulation* **71**(1) 7–22.

Choueiry, B, McIlraith, S, Iwasaki, Y, Loeser, T, Neller, T, Engelmore, RS and Fikes, R, 1998, "Preliminary thoughts towards a practical theory of reformulation for reasoning about physical systems" *Proceedings of the 12th International Workshop on Qualitative Reasoning about Physical Systems* 21–31.

Coghill, GM and Chantler, MJ, 1999, "Constructive and non-constructive asynchronous qualitative simulation" *Proceedings of the 13th International Workshop on Qualitative Reasoning about Physical Systems* 51–61.

Coghill, GM, Shen, Q, Chantler, MJ and Leitch, RR, 1999, "Towards the use of multiple models for diagnosis of dynamic systems *Proceedings of the 9th International Workshop on Principles of Diagnosis* 51–58.

Crawford, J, Farquhar, A and Kuipers, BJ, 1990, "A compiler from physical models into qualitative differential equations *Proceedings of the 12th National Conference on Artificial Intelligence* 365–372.

Davis, R, 1984, "Diagnostic reasoning based on structure and behavior" *Artificial Intelligence* **24** 347–410.

De Kleer, J and Brown, JS, 1984, "A qualitative physics based on confluences" *Artificial Intelligence* **24** 7–83.

De Kleer, J and Williams, BC, 1987, "Diagnosing multiple faults" *Artificial Intelligence* **32** 97–130.

De Kleer, J and Williams, BC, 1989, "Diagnosis with behavioral modes" *Proceedings of the 11th International Joint Conference on Artificial Intelligence* 1324–1330.

De Kleer, J, 1986, "An assumption-based TMS" *Artificial Intelligence* **28** 127–162.

Dornbusch, R and Fischer, S, 1994, *Macroeconomics* McGraw-Hill.

Falkenhainer, B and Forbus, KD, 1991, "Compositional modeling: finding the right model for the job" *Artificial Intelligence* **51** 95–143.

Farquhar, A and Brajnik, G, 1994, "A semi-quantitative physics compiler" *Proceedings of the 8th International Workshop on Qualitative Reasoning about Physical Systems* 81–89.

Farquhar, A, 1993, "Automated modeling of physical systems in the presence of incomplete knowledge" Ph.D. Dissertation, University of Texas at Austin.

Farquhar, A, 1994, "A qualitative physics compiler" *Proceedings of the 12th National Conference on Artificial Intelligence* 1168–1174.

Fedorov, VV, 1972, *Theory of Optimal Experiments* Academic Press.

Forbus, KD and Falkenhainer, B, 1990, "Self-explanatory simulations: an integration of qualitative and quantitative knowledge" *Proceedings of the 8th National Conference on Artificial Intelligence* 380–387.

Forbus, KD and Whalley, PB, 1994, "Using qualitative physics to build articulate software for thermodynamics education" *Proceedings of the 8th International Workshop Qualitative Reasoning about Physical Systems* 106–113.

Forbus, KD, 1984, "Qualitative process theory" *Artificial Intelligence* **24** 85–168.

Forbus, KD, 1990, "The qualitative process engine" in D Weld and J de Kleer (eds) *Readings in Qualitative Reasoning about Physical Systems* Morgan-Kaufmann.

Forbus, KD, Whalley, PB, Everett, JO, Ureel, L, Brokowski, M, Baher, J and Kuehne, SE, 1999, "CyclePad: an articulate virtual laboratory for engineering thermodynamics *Artificial Intelligence* **114** 297–347.

Forrester, JW, 1961, *Industrial Dynamics* MIT Press.

Gawthrop, P, 1996, *Metamodelling: Bond Graphs and Dynamic Systems* Prentice Hall.

Gruber, TR and Gautier, PO, 1993, "Machine-generated explanations of engineering models: a compositional modeling approach" *Proceedings of the 13th International Joint Conference on Artificial Intelligence* 1502–1508.

Gruber, TR, Vemuri, S and Rice, J, 1996, "Model-based virtual document generation" Technical report KSL-96-16, Knowledge Systems Laboratory, Stanford University.

Hamscher, W, Console, L and de Kleer, J, 1992, *Readings in Model-Based Diagnosis* Morgan-Kaufmann.

Heller, U and Struss, P, 1996, "Transformation of qualitative dynamic models – application in hydro-ecology" *Proceedings of the 10th International Workshop on Qualitative Reasoning about Physical Systems* 83–92.

Heller, U and Struss, P, 1998, "Diagnosis and therapy recognition for ecosystems: usage of model-based diagnosis techniques" *Proceedings of the 12th International Symposium "Computer Science for Environment Protection"*.

Iwasaki, Y, Farquhar, A, Fikes, R and Rice, J, 1997, "A web-based compositional modeling system for sharing of physical knowledge" *Proceedings of the 15th International Joint Conference on Artificial Intelligence* 494–500.

Iwasaki, Y and Levy, AY, 1994, "Automated model selection for simulation" *Proceedings of the 12th National Conference on Artificial Intelligence* 1183–1190.

Iwasaki, Y and Low, CM, 1992, "Device modeling environment: an integrated model-formulation and simulation environment for continuous and discrete phenomena" *Proceedings of the Conference on Intelligent Systems Engineering* 141–146.

Iwasaki, Y and Simon, HA, 1994, "Causality and model abstraction" *Artificial Intelligence* **67**(1) 143–194.

Iwasaki, Y, Farquhar, A, Saraswat, V, Bobrow, D and Gupta, V, 1995, "Modeling time in hybrid systems: how fast is 'instantaneous'?" *Proceedings of the 14th International Joint Conference on Artificial Intelligence* 1773–1780.

Kay, H, Rinner, B and Kuipers, B, 2000, “Semi-quantitative system identification” *Artificial Intelligence* **119** 103–140.

Keppens, J and Shen, Q, 2000, “Towards compositional modelling of ecological systems via dynamic flexible constraint satisfaction” *Proceedings of the 14th International Workshop on Qualitative Reasoning about Physical Systems* 74–82.

Kuipers, BJ, 1986, “Qualitative simulation” *Artificial Intelligence* **29** 289–338.

Kuipers, BJ, 1987, “Abstraction by time scale in qualitative simulation” *Proceedings of the 6th National Conference on Artificial Intelligence* 621–625.

Kuipers, BJ, 1994, *Qualitative Reasoning: Modeling and Simulation with Incomplete Knowledge* MIT Press.

Leitch, RR, Shen, Q, Coghill, G and Chantler, M, 1999, “Choosing the right model” *IEE Proceedings on Control Theory and Applications* 435–449.

Levy, AY, Iwasaki, Y and Fikes, R, 1997, “Automated model selection for simulation based on relevance reasoning” *Artificial Intelligence* **96** 351–394.

Ljung, L, 1999, *System Identification: Theory for the User* MIT Press.

Luger, GF and Stubblefield, WA, 1993, *Artificial Intelligence: Structures and Strategies for Complex Problem Solving* The Benjamin/Cummings Publishing Company, Inc.

McIlraith, S, Biswas, G, Fromherz, M, Howe, J, Fikes, R, Bobrow, D, Cutkosky, M, Engelmore, RS and Neller, T, 1998, “Model-enabled control of hybrid systems” Technical report KSL-98-22, Stanford University.

Miguel, I and Shen, Q, 1999a, “Extending FCSP to support dynamicaly changing problems” *Proceedings of the 8th International Conference on Fuzzy Systems* 1615–1620.

Miguel, I and Shen, Q, 1999b, “Hard, flexible and dynamic constraint satisfaction” *Knowledge Engineering Review* **14**(3), 199–220.

Mitchell, TM, 1997, *Machine Learning* McGraw-Hill.

Mittal, S and Falkenhainer, B, 1990, “Dynamic constraint satisfaction problems” *Proceedings of the 8th National Conference on Artificial Intelligence* 25–32.

Mosterman, PJ, Kapadia, R and Biswas, G, 1995, “Using bond graphs for diagnosis of dynamic physical systems” *Proceedings of the 5th International Workshop on Principles of Diagnosis* 81–85.

Nayak, PP and Joskowicz, L, 1996, “Efficient compositional modeling for generating causal explanations” *Artificial Intelligence* **83** 193–227.

Nayak, PP, 1992, “Order of magnitude reasoning using logarithms” *Proceedings of the 3rd International Conference on Principles of Knowledge Representation and Reasoning* 201–210.

Nayak, PP, 1994, “Causal approximations” *Artificial Intelligence* **70** 277–334.

Nayak, PP, 1995, *Automated Modeling of Physical Systems* Lecture Notes in Artificial Intelligence, Springer.

Neter, J, Kutner, MH, Nachtsheim, CJ and Wasserman, W, 1996, *Applied Linear Statistical Models* Irwin.

Parsons, S and Hunter, A, 1998, “A review of uncertainty handling formalisms” in idem, *Applications of Uncertainty Formalisms* Springer-Verlag.

Pearl, J, 1988, *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference* Morgan-Kaufmann.

Pugh, AL, 1976, *DYNAMO User’s Manual* MIT Press.

Raiman, O, 1991, “Order of magnitude reasoning” *Artificial Intelligence* **51** 11–38.

Ramachandran, S, Mooney, RJ and Kuipers, BJ, 1994, “Learning qualitative models for systems with multiple operating regions” *Proceedings of the 8th International Workshop on Qualitative Reasoning about Physical Systems* 212–223.

Richards, BL, Kraan, I and Kuipers, BJ, 1992, “Automated abduction of qualitative models” *Proceedings of the 10th National Conference on Artificial Intelligence* 723–728.

Rickel, J and Porter, B, 1994, “Automated modeling for answering prediction questions: selecting the time scale and system boundary” *Proceedings of the 12th National Conference on Artificial Intelligence* 1191–1198.

Rickel, J and Porter, B, 1997, “Automated modeling of complex systems to answer prediction questions” *Artificial Intelligence* **93** 201–260.

Say, AC and Kuru, S, 1996, “Qualitative system identification: deriving structure from behavior” *Artificial Intelligence* **83** 75–141.

Scott, JA and Coghill, GM, 1998, “Qualitative Euler integration with continuity” *Proceedings of the 12th International Workshop on Qualitative Reasoning about Physical Systems* 114–122.

Scott, JA, Leitch, RR and Wyatt, GJ, 1995, “Reducing precision to achieve accurate economic models” *Economic and Financial Computing* Summer/Autumn 79–97.

Shen, Q, Peng, T and Milne, R, 1999, “Dimensional analysis based causal ordering” *Proceedings of the 13th International Workshop on Qualitative Reasoning about Physical Systems* 193–202.

Struss, P and Heller, U, 1998, “Process-oriented modeling and diagnosis: revising and extending the theory of diagnosis from first principles” *Proceedings of the 9th International Workshop on Principles of Diagnosis*.

Struss, P and Heller, U, 1999, “Model-based support for water treatment” *Proceedings of the IJCAI’99 Workshop on Qualitative and Model Based Reasoning for Complex Systems and their Control* 84–90.

Struss, P, 1998, “Artificial intelligence for nature: why knowledge representation and problem solving should play a key role in environmental decision support” in HD Haasis and KC Ranze (eds) *Computer Science for the Environmental Protection ’98* Metropolis Verlag.

Ulrich, K, 1988, “Computation and pre-parametric design”, Technical Report 1043, MIT.

Walter, G, 1999, *Compartmental Modeling with Networks* Birkhauser Boston.

Wang, J, 1997, “Enterprise modeling: a compositional modeling approach” *Proceedings of the 4th Conference of the International Society for Decision Support Systems* 53–70.

Washio, T and Motoda, H, 1998, “Discovery of first-principle equations based on scale-type-based and data-driven-reasoning” *Knowledge-Based Systems* **10** 403–411.

Weld, DS and Addanki, S, 1991, “Query directed approximations” in B Faltings and P Struss (eds) *Recent Advances in Qualitative Physics* MIT Press.

Weld, DS and de Kleer, J, 1990, *Readings in Qualitative Reasoning About Physical Systems* Morgan Kaufmann.

Weld, DS, 1990, “Approximation reformulations” *Proceedings of the 8th National Conference on Artificial Intelligence* 407–412.

Wu, C and Burke, TJ, 1998, “Intelligent computer aided optimization on specific power of an OTEC Rankine power plant” *Applied Thermal Engineering* **18**(5) 295–300.

Wu, C and Dieguez, M, 1998, “Intelligent computer aided design on optimization of specific power of finite-time Rankine cycle using CyclePad” *Computer Application in Engineering Education* **16**(1) 9–13.

Wyatt, GJ, Leitch, RR and Steele, AD, 1995, “Qualitative and quantitative simulation of interacting markets” *Decision Support Systems* **15** 105–113.

Xia, S, Linkens, DA and Bennett, S, 1993, “Automated modelling and analysis of dynamic physical systems using qualitative reasoning and bondgraphs” *Intelligent Systems Engineering Journal* Autumn 201–212.