ELSEVIER

Knowledge-Based Systems 9 (1996) 105–118

Knowledge-Based SYSTEMS

Redesign and reuse in compositional knowledge-based systems

Frances M.T. Brazier, Pieter H.G. van Langen, Jan Treur, Niek J.E. Wijngaards

Artificial Intelligence Group, Department of Mathematics and Computer Science, Vrije Universiteit Amsterdam, De Boelelaan 1081a, 1081 HV Amsterdam, Netherlands

Abstract

The paper introduces a task model for the redesign of compositional knowledge-based systems based on a generic task model of design. A generic task model of design provides an abstract description of a design task and a generic structure which can be refined for design tasks in specific domains of application. A generic task model of design, shown to incorporate redesign, is presented and refined to a task model for the redesign of compositional knowledge-based systems. The applicability of this task model is illustrated for the redesign of a diagnostic knowledge-based system.

Keywords: Redesign; Reuse; Compositional architectures; Generic task models

1. Introduction

Knowledge of alternative (models of) systems and system components is often the basis for redesign of an existing system, for example a software system or hardware system. This holds in particular for the redesign of compositional knowledge-based systems. Existing task models, varying from generic to more specific and instantiated or noninstantiated, are candidate components for replacement, refinement, specialisation or instantiation of components of an existing knowledge-based system. Such components are also often used during initial design. Redesign is, in essence, an inherent part of most design processes; new requirements or new domain knowledge often influence design processes. Design is a complex task, in which extensive knowledge of the domain of application is essential. The domain knowledge for design is broad. It includes not only knowledge of characteristics of the design object domain and knowledge of existing (partial) *design object descriptions* and sets of *requirements*, but also knowledge of *design strategies* to guide the design process. As design necessarily entails (re)use of such design knowledge, a thorough analysis of a design process is of importance in understanding the extent to which existing design domain knowledge can be effectively employed and how.

In principle, design is a process in which, given existing design object descriptions and a set of requirements (and their qualifications), an object is designed on the basis of knowledge of the design object domain and knowledge of design strategies. *Qualifications* of requirements denote preferences between (sets of) requirements, indicating the importance of (sets of) requirements for the design process. Some requirements may be qualified as hard, and others as soft, for example. During a design process, individual requirements may be (temporarily) translated (by deductive or heuristic reasoning) to a set of more specific requirements. Fulfilment of the specific requirements implies the fulfilment of the more broadly specified requirements from which they were derived. Reasoning about requirements is also needed to manage conflicting requirements, and to determine which requirements should be imposed (and which should be retracted) at a given point in the design process [1].

A *generic task model of design*, in which reasoning about requirements and their qualifications and reasoning about design object descriptions are distinguished, has been proposed by Brazier, Langen, Ruttkay, and Treur [2]. This model is based on a logical analysis of design process [3] and on analyses of existing applications [4,5]. It provides not only an abstract description of a design process that is comparable with a *design model* [6] or a *design theory* [7], but also a generic structure which can be refined for specific design tasks in different domains of application. Refinement of the generic task model of design, by specialisation and

SSDI 0950-7051(95)01026-2

instantiation, involves the specification of knowledge about applicable requirements and their qualifications, about the design object domain, and about design strategies.

Reuse of task models is essential to a compositional approach to system design. It provides a basis for reuse at more specific levels: reuse of more specific task models and reuse of specific (instantiated and noninstantiated) components designed to perform specific subtasks. A description of this process for the design of an elevator configuration, based on the documentation provided by Yost [8], can be found in Brazier, Langen, Treur, Wijngaards and Willems [9]. Notions similar to our notion of a generic task model [10] can be found in the literature, such as generic tasks [11,12] and interpretation models [13].

In general, reuse involves both retrieval (e.g. from a library) and modification of applicable task models and components; this paper focuses primarily on modification. The generic task model of design proposed by Brazier, Langen, Ruttkay and Treur [2] is shown to incorporate redesign and it will be (re)used to obtain a task model for redesign of compositional knowledge-based systems. The applicability of this task model for redesign will be illustrated for the redesign of a (compositional) diagnostic knowledge-based system. In this example, new requirements are imposed on an existing knowledge-based system. The redesign process will be described, illustrating how existing components are reused (selected and modified) to meet the new set of requirements.

2. Design concepts and a generic task model of design

In this section, the characteristics of design processes are informally introduced, the concepts related to the design task are explained, and a generic task model of design is presented.

2.1. Characteristics of design processes

An initial design problem statement is expressed by a user as a set of initial requirements and requirement qualifications. *Requirements* impose conditions and restrictions on the structure, functionality and behaviour of the *design object* for which a structural description is to be generated during design. *Qualifications* of requirements are qualitative expressions of the extent to which (individual or groups of) requirements are considered hard or preferred, either in isolation or in relation to other (individual or groups of) requirements. At any one point in time during design, the design process focuses on a specific subset of the set of requirements. This subset of requirements plays a central role; the design process is (temporarily) committed to the current requirement qualification set. The aim of generating a design object description is to satisfy these requirements. Other qualifications of requirements may play a heuristic role.

During design the considered subsets of the set of requirements may change as may the requirements themselves. The same holds for design object descriptions and design object knowledge; they evolve during design. The strategy employed for the coordination of requirement qualification set manipulation and design object description manipulation may also change during the course of a single design process. Modifications to the requirement qualification set, the design object description and the design strategy may be the result of straightforward implications drawn from knowledge available to a design support system. Modifications may also be the result of specific knowledge about appropriate default assumptions (see also Smith and Boulanger [14]), or the result of interaction with an outside party (e.g. a client or a designer).

In order to manage the complexity of design, the *design history* plays an important role. It can help to find solutions to design problems that have proved to be proficient in the past, it can indicate why these solutions were chosen, and it can prevent unintended retracing of design steps. The rationale (the record of decisions and the reasons they are based on) is a part of the design history.

Fig. 1 illustrates an example of a two-dimensional *design space* spanned by requirement qualification set and design object descriptions. In general, for each given point in a design space, a large (and possibly infinite) number of other points could be generated by means of modification, but only few are of interest (because they are generated by modifications that 'make sense'). These are the possible alternative choices for the next step in the design process. To describe the dynamics of the design process, the circumstances must be specified under which a choice among these alternatives is made. For each of the two dimensions spanning

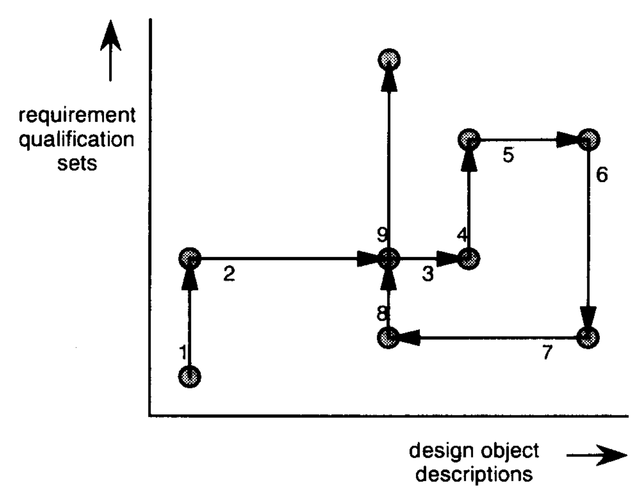

Fig. 1. Example of navigation through the design space.

the design space, this involves strategic knowledge of the various design decisions that have to be made. Furthermore, strategic knowledge is needed to determine whether and along which of the two dimensions the next step of the design process is to be made.

In Fig. 1, a nine-step sequence of requirement qualification set modifications and design object description modifications is shown. The first step in the sequence, depicted by the arrow labelled 1, represents a modification to the initial requirement qualification set only. The initial design object description is modified in Steps 2 and 3. After Step 3 (i.e. at the point at which the arrow labelled 3 ends), modification of the design object description halts for some reason. Maybe the design object description satisfies all requirements of the current requirement qualification set, or maybe there is reason to believe that no design object description can be created that satisfies all requirements. In each case, the current requirement qualification set is modified in Step 4, taking into account the reason why modification of the current design object description stopped. After the modifications in Steps 5–8, the design process reaches an interesting state. The sequence of modifications has led to a requirement qualification set and a design object description that in combination are equivalent to the result of Step 2. Therefore a direction is sought that is different from the one chosen in Step 3, resulting in Step 9 in a modification to the current requirement qualification set. In summary, there are five requirement qualification set modifications (Steps 1, 4, 6, 8 and 9) and four design object description modifications (Steps 2, 3, 5 and 7).

The concepts employed within our framework for design can be divided into those related to design object descriptions, those related to requirement qualification sets, and those related to coordination of the design process. In this paper, informal descriptions of these concepts are presented. The formal semantics of design processes (based on partial logic for the states and temporal partial logic [15] for the reasoning traces (sequences of states)) are discussed by Brazier, Langen, and Treur [3].

2.2. Concepts related to design object descriptions

The concepts distinguished with respect to the design object include the following:

- *design object description*: a (partial) description of properties of a design object,
- *design object description space*: the set of all possible design object descriptions,
- *domain knowledge about design objects*: knowledge about properties of an object and relations between properties of objects in the domain,
- *design object refinement*: a relation which holds between two design object descriptions if the second design object description is a conservative modification of the first,
- *design object description modification steps*: a relation which holds between two design object descriptions if the second design object description is a modification of the first (allowing revision, for example).

A design object may have a hierarchical structure (e.g. a compositional architecture) and this structure is reflected in the design object description.

2.3. Concepts related to requirement qualification sets

The concepts distinguished with respect to the requirement qualification sets include the following:

- *requirement*: a specification of a property of an object that is desired or expected,
- *requirement qualification*: a statement qualifying sets of requirements,
- *requirement qualification set space*: the space of all possible requirement qualification sets,
- *requirement qualification specialisation*: a relation between a requirement qualification set and a more specific version of the requirement qualification set,
- *commitment to requirements*: a translation of requirements and their qualifications into requirements, expressing which requirements must be satisfied,
- *requirement qualifications set modification steps*: a relation which holds between two requirement qualification sets if the second requirement qualification set is a modification of the first (allowing refinement and revision, for example).

Requirements may have a hierarchical structure, reflecting dependencies between requirements; one requirement may entail other, more specific requirements. This structure makes it possible to retract related parts of the requirement qualification set, or to expand the requirement qualification set by more specific requirements (decomposition).

2.4. Concepts related to design process coordination

A design process is not a random process. Explicit strategic knowledge is used to coordinate interaction between the spaces of requirement qualification sets and design object descriptions. This knowledge is required to *evaluate* the current state of the *design process*, and to draw conclusions about continuation of the design process, i.e. whether the design process should continue and how.

Concepts distinguished in the context of design process coordination include the following:

- *current requirements*: the requirements that are currently in the focus of the requirement qualification set manipulation process and that are to be satisfied in the design object description manipulation process,

- *design problem description*: a definition of an initial (and possibly empty) design object description, domain knowledge and an initial requirement qualification set,
- *design solution*: a set of requirements together with a design object description such that, according to given domain knowledge, the design object description satisfies the set of requirements,
- *evaluations of current requirements*: information on which requirements are already satisfied by the current design object description and which are not (yet),
- *state of the design process*: information on the progress of requirement qualification set manipulation and design object description manipulation.

2.5. Generic task model of design

A generic task model models domain independent characteristics of a class of complex tasks. The knowledge in a generic task model includes knowledge of

- a hierarchical task decomposition,
- information exchange between (sub)tasks,
- sequencing of (sub)tasks,
- generic knowledge structures,
- task delegation between participants.

In the conceptualisation of a generic task model of design [2] shown in Fig. 2, (sub)tasks are represented by (sub)components, information exchange by information links and sequencing of tasks by task control knowledge. Neither knowledge structures nor the role(s) of different participating agents in design are explicitly depicted.

The role of a user (e.g. a designer or a client) in interaction with a design support system, based on the above generic task model of design, is often to participate in modifying requirement qualification sets and/or design object descriptions, and the evaluation of the design process.

The above generic task model of design can be subdivided into three parts: manipulation of requirements and requirement qualifications, manipulation of design object descriptions, and design process coordination. The subcomponents of each part are described below.

The four subcomponents related to the *manipulation of requirement qualification sets* are as follows:

- *Modification*: The current requirement qualification set is analysed, proposals for modification are generated and compared, and the most promising (according to some measure) is selected.
- *Deductive refinement*: The current requirement qualification set is deductively refined by means of the theory of requirement qualification sets.
- *Update of current description*: The current requirement qualification set is stored and maintained.
- *Update of modification history*: The history of requirement qualification sets modification is stored and maintained.

The four subcomponents related to the *manipulation of design object descriptions* are as follows:

- *Modification*: The current design object description is

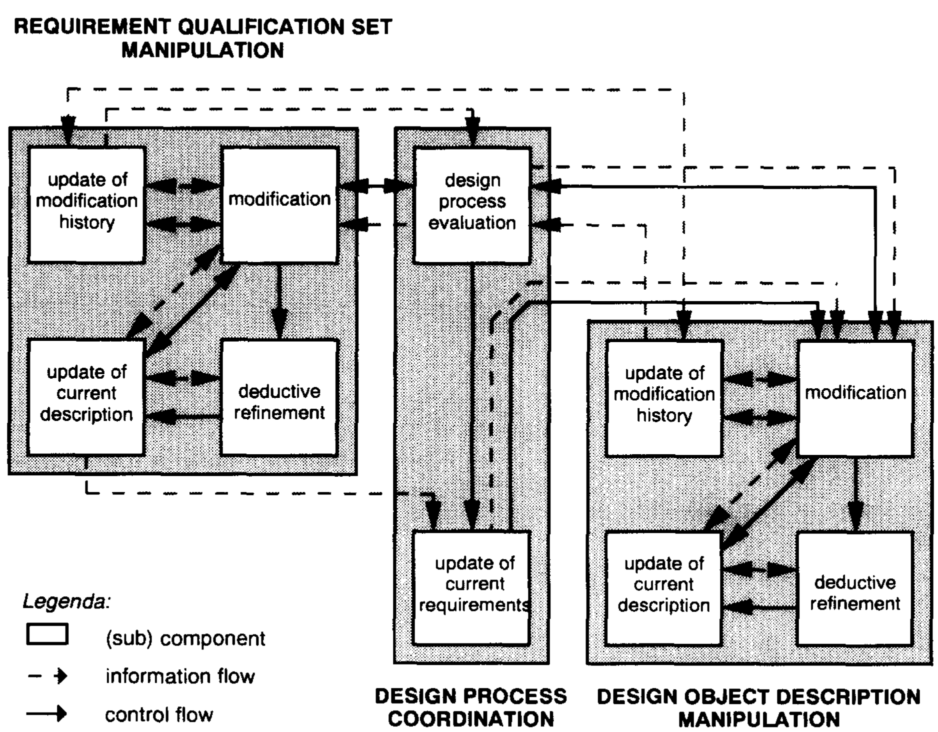

Fig. 2. Generic task model of design.

analysed in relation to the current requirement set, proposals for modification are generated and compared, and the most promising (according to some measure) is selected.
- *Deductive refinement*: The current design object description is deductively refined by means of the theory of design object descriptions.
- *Update of current description*: The current design object description is stored and maintained.
- *Update of modification history*: The history of design object descriptions modification is stored and maintained.

The two subcomponents related to *design process coordination* are as follows:

- *Design process evaluation*: The status of the design process is evaluated and control coordinated; the design process may continue by activation of requirement qualification set manipulation and/or design object description manipulation or by termination of process.
- *Update of current requirements*: In this component, the current requirement set (subordinate to the current requirement qualification set) is stored and maintained.

The overall coordination of the design process together with the local coordination within the manipulation components determines the course of the design process. This corresponds to the notion of design navigation as described by Petrie, Cutkosky, and Park [16].

3. Redesign of compositional architectures

Modelling redesign of compositional knowledge-based systems requires commitments to

- structure for design objects,
- a generic task model for redesign,
- knowledge to be used to refine the generic task model.

In this section, the notion of compositional architecture is used as a structure for design objects. Furthermore, it is argued that redesign is an integral part of design and can therefore be modelled by a generic task model of design, and the types of knowledge involved in the redesign of compositional architectures are described.

3.1. Compositional architectures

One of the crucial elements which play a role in the (re)design of any design object is the structure of the object. Structure (for example in terms of components and links) is often specified in terms of a specific (standardised) framework. For the redesign of knowledge-based systems the compositional framework for knowledge-based systems [17,18] provides such structure. Compositional architectures specified within this compositional framework include structures for the following:

- hierarchies of components with input and output interfaces,
- information links between components,
- both task control knowledge (to control activation of a component's subcomponents) and kernel knowledge (a composed component's subcomponents or the domain knowledge required to perform a primitive component's task) within each component.

As the design objects considered in this paper are compositional knowledge-based systems, the modelling framework *DESIRE* [17–19] can be used for the (partial) description of design objects.

3.2. Modelling redesign as a design task

Modification of both requirement qualification sets and design object descriptions is an essential part of the design process described above. Requirements may change as a result of new insights (e.g. by a designer or a client) during a design process. In general, requirements *will* change, because in practice requirements are often imprecise, incomplete, and ambiguous. Likewise, new design object knowledge may be discovered during design. Design inherently involves trial and error, both with respect to requirement qualification sets and design object descriptions.

As in initial design (i.e. design starting without a design object description), requirements, their qualifications, and design object descriptions may evolve during redesign (design on the basis of an existing design object description). Redesign is in principle part of most design processes. During design, a (partial) design object description exists that is modified in the course of the design process. A generic task model of design should therefore be applicable to redesign.

3.3. Knowledge related to redesign of compositional knowledge-based systems

Redesign of compositional knowledge-based systems requires knowledge about the following:

- *design objects* (and their manipulation), which are themselves compositional knowledge-based systems,
- *requirements* of compositional knowledge-based systems (and their manipulation),
- the *redesign process coordination*.

Knowledge about *design objects* includes knowledge about the structure, function and behaviour of compositional knowledge-based systems, and knowledge about how to modify them, such as knowledge to

- determine the syntactical correctness of the formal specification of a compositional architecture (e.g. correct use of names and references within (a component of) a compositional architecture, or correct use of formulae of the logic used within knowledge bases and information links),

- assess the coherence of components, kernel links and task control links,
- assure the consistent distinction between components of a compositional architecture into object level components, metalevel components, metametalevel components, etc.,
- analyse interactions with the user(s) and when and at which level (object level, metalevel, etc.) (e.g. the order in which output information is generated or requests for information are deferred to the user),
- locate problematic parts in the current compositional architecture description (i.e. determine the focus on the current design object description),
- determine when to introduce new components, or when to decompose a component,
- determine when additional information links between components are essential,
- analyse the hierarchical compositional structure,
- generate candidate modifications to the current design object description, compare candidate modifications and put them in a specific order, and select candidates accordingly.

Requirements of a compositional knowledge-based system address the desired or needed structure, function and/or behaviour of the system. This includes knowledge to

- generate candidate modifications to the current requirement qualification set, compare candidate modifications and put them in a specific order, and select candidates accordingly,
- locate problematic requirements and their qualifications (i.e. determine the focus on the current requirement qualification set),
- analyse requirements for contradicting requirements, either directly or indirectly (via domain knowledge),
- determine source of requirements (e.g. a particular client).

Redesign process coordination knowledge is needed to

- evaluate the current state of the design process,
- determine when to modify either the current (partial) description of the compositional knowledge-based system or the current requirement qualification set,
- determine the interaction between agents (e.g. clients, designers).

4. Task model of redesign

To use the generic task model of (re)design in a particular domain of application, *specialisation* of the generic task model is required to tune the generic task model to the specific redesign task, followed by *instantiation* of the specialised task model to the domain of application. To redesign compositional knowledge-based systems, the modification subcomponents within the RQS manipulation and DOD manipulation components of the generic task model of (re)design have each been specialised into

- *analysis of current description*, which investigates what problems are present in the current RQS or DOD,
- *modification focus determination*, which determines which parts of the current RQS or DOD must be modified to be able to resolve the identification problems,
- *modification method determination*, which determines the method for modifying the parts of the current RQS or DOD that are in focus,
- *modification according to method*, which modifies the parts of the current RQS or DOD that are in focus, according to the method determined.

The decomposition of modification into these four subcomponents is shown in Fig. 3. For the redesign of compositional knowledge-based systems, these subcomponents have to be instantiated with domain knowledge about compositional knowledge-based systems.

5. Example of compositional system redesign

In this section, the applicability of the generic task model or (re)design is illustrated for the redesign of a compositional knowledge-based system for diagnostic reasoning. Redesign is assumed to be performed by a redesign support system in close interaction with a knowledge engineer. The redesign process will be explained with reference to a trace, showing the sequence in which components of the task model for redesign of compositional knowledge-based systems are activated and the results of these components.

The design object to be redesigned is a simple system for diagnostic reasoning, as shown in Fig. 4, entailing the formulation of complaints, determination of hypotheses on the basis of the complaints and symptoms observed, and evaluation of the determined hypotheses, possibly requiring additional observations.

The requirement qualification set on which the design of the original diagnostic reasoning system was based is shown in Table 1. These are requirements and their qualifications.

Two new, additional requirements imposed by the knowledge engineer on the diagnostic reasoning system are shown in Table 2.

These new requirements may or may not affect the design of the diagnostic reasoning system. Therefore,

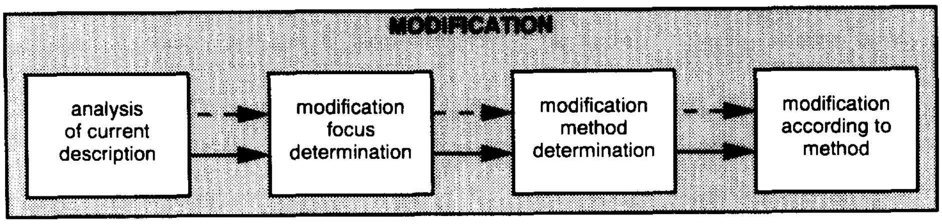

Fig. 3. Structure of the modification subcomponents.

first the new requirement qualification set is analysed and it is noticed that the new and rather abstract requirement qualifications *RQ1* and *RQ2* are to be refined to make their satisfaction possible. As a result, in total, five new requirements plus qualifications emerge. After that, it is decided that for the time being only the hard requirements need to be considered in the manipulation of the design object description. Then the current design object description is analysed to see if it satisfies these requirements. It is noticed that this is not the case; the requirements that resulted from the refinement of *RQ1* are not satisfied. In order to resolve this problem, a number of modifications to the design object description are made, resulting in a new specification of the component Hypothesis Determination.

Having succeeded in satisfying the hard requirements, it is then decided that requirements with other qualifications now also need to be considered. As a consequence, the soft requirement *RQ2* and its refinement are now also imposed on the current design object description. To satisfy these requirements, another change to the design object description is made: a subcomponent Strategy Determination plus the appropriate information links and task control is added to the specification of the component Hypothesis Determination.

The knowledge engineer is happy that all requirements could be satisfied (because s/he was not sure beforehand) and appreciates the changes made to the design object description. Seeing further opportunities, s/he adds a hard requirement *RQ3* which details the functionality of the subcomponent Strategy Determination: the strategy for determination of hypotheses is to be established by the diagnostic reasoning system in interaction with the user. This makes a third round of changing the design object description necessary, resulting in a decomposition of the subcomponent Strategy Determination. After this, the knowledge engineer feels comfortable with the new diagnostic reasoning system and imposes no further requirements.

This redesign process is presented in the trace below, showing the activation of (sub)components chronologically, together with the results of activation. Activated subcomponents are preceded by numbers. The abbreviations used are listed in Table 3.

Note that the result of each modification to the current description of requirement qualifications or the design object is taken to be the union of (a) the part of the current description that is *not* in focus and (b) the result of applying the method to the focus.

The knowledge engineer has started the design process evaluation and indicated that s/he wants to manipulate requirements and their qualifications.

\>> **RQS update of current description**
`By adding RQ1 and RQ2 (by the KE) the current description is updated to {RQa, RQb, RQc, RQd, RQ1, RQ2}.`

\>> **RQS update of modification history**
`The history is updated to`

RQS_0 = {RQa, RQb, RQc, RQd}
RQS_1 = {RQa, RQb, RQc, RQd, RQ1, RQ2}
rationale (⟨RQS_0, DOD_0⟩, ⟨RQS_1, DOD_0⟩, method (KE)).

RQS_1 is analysed and it is noticed that both *RQ1* and *RQ2* are abstract. To make a satisfactory design object description more specific requirement qualifications are needed. This problem is resolved by adding more specific requirement qualifications *RQ1′* ad *RQ2′* on the basis of defaulting reasoning.

\> > **RQS modification**

1. **analysis of current description**
RQ1 and RQ2 are abstract.
2. **modification focus determination**
The local focus of modification is set to {RQ1, RQ2}.
3. **modification method determination**
The method chosen is *modification by default reasoning.*
4. **modification according to method**
The default requirements and their qualifications are:

RQ1′: "The system is able to determine

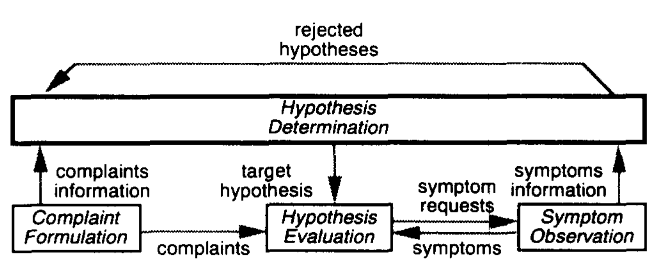

Fig. 4. Original diagnostic reasoning system.

Table 1
Requirement qualifications for the original diagnostic reasoning system

Identifier	Requirement	Qualification
*RQ*a	The system is able to formulate complaints	Hard
*RQ*b	The system is able to determine hypotheses	Hard
*RQ*c	The system is able to evaluate hypotheses	Hard
*RQ*d	The system is able to observe symptoms	Hard

which hypothesis is to be considered in a structured manner." (Hard.)

RQ2′: "If the system is able to determine which hypothesis is to be considered, then it is able to determine a strategy for determining hypotheses." (Soft.)

>> **RQS update of current description**

The current description is updated to {RQa, RQb, RQc, RQd, RQ1, RQ1′, RQ2, RQ2′} by adding RQ1′ and RQ2′.

>> **RQS update of modification history**

The history is updated by adding

RQS_1 = {RQa, RQb, RQc, RQd, RQ1, RQ2}
RQS_2 = {RQa, RQb, RQc, RQd, RQ1, RQ1′, RQ2, RQ2′}
rationale (⟨RQS_1, DOD_0⟩, ⟨RQS_2, DOD_0⟩, method(default_reasoning))
rationale (⟨RQS_1, DOD_0⟩, ⟨RQS_2, DOD_0⟩, has_interpretation(RQ1, {RQ1′}))
rationale (⟨RQS_1, DOD_0⟩, ⟨RQS_2, DOD_0⟩, has_interpretation (RQ2, {RQ2′})).

RQS_2 is analysed and it is noticed that it can be further refined in a unique manner. There is domain knowledge available that can be used to infer (in a deductive manner) from *RQ1′* three other more specific requirement qualifications.

>> **RQS modification**

1. **analysis of current description**

RQ1′ can be refined.

2. **modification focus determination**

The focus of modification is set to {RQ1′}.

3. **modification method determination**

Table 2
Additional requirement qualifications for the diagnostic reasoning system

Identifier	Requirement	Qualification
RQ1	The system proposes fewer faulty hypotheses, in comparison with random proposal	Hard
RQ2	If the system proposes fewer faulty hypotheses, in comparison with random proposal, then it should also be able to determine a strategy for proposing hypotheses	Soft

Table 3
Abbreviations used in the trace

Abbreviation	Explanation
KE	Knowledge engineer
R	Requirement
RQ(S)	Requirement qualification (set)
DOD	Design object description
CF	Complaint formulation
HD	Hypothesis determination
HE	Hypothesis evaluation
SO	Symptom observation

The method chosen is *modification by deductive refinement*.

>> **RQS deductive refinement**

The newly proposed requirements and their qualifications are:

RQ1′.1: "Structured determination of hypotheses involves being able to generate hypotheses." (Hard.)

RQ1′.2: "Structured determination of hypotheses involves being able to compare hypotheses." (Hard.)

RQ1′.3: "Structured determination of hypotheses involves being able to select hypotheses." (Hard.)

>> **RQS update of current description**

The current description is updated to {RQa, RQb, RQc, RQd, RQ1, RQ1′, RQ1′.1, RQ1′.2, RQ1′.3, RQ2, RQ2′}.

>> **RQS update of modification history**

The history is updated by adding

RQS_3 = {RQa, RQb, RQc, RQd, RQ1, RQ1′, RQ1′.1, RQ1′.2, RQ1′.3, RQ2, RQ2′}
rationale(⟨RQS_2, DOD_0⟩, ⟨RQS_3, DOD_0⟩, method(deductive_refinement))
rationale(⟨RQS_2, DOD_0⟩, ⟨RQS_3, DOD_0⟩, has_refinement (RQ1′, {RQ1′.1, RQ1′.2, RQ1′.3})).

RQS_3 is analysed and it is noticed that there seems to be no more problems. However, satisfying the new requirements has not yet been tried. No modification has been made to the design object description since the introduction of the new requirement qualifications, which can be concluded from inspecting the history (the DOD_0 is the only DOD that is known of). Therefore, in order to be cautious, all requirements with lower qualifications are discarded for the time being. Given the current RQS, this means that the soft requirements need not be satisfied by the current DOD.

>> **RQS modification**

1. **analysis of current description**

There seem to be no more problems with the current RQS, but, since no attempt has been made to create a new DOD since the introduction of new requirement qualifications, all requirements with lower qualifications are to be discarded for the moment.

2. **modification focus determination**
The focus of modification is set to {RQ2, RQ2′}.

3. **modification method determination**
The method chosen is *modification by deletion*.

4. **modification according to method**
All requirement qualifications in focus are deleted.

>> **RQS update of current description**
The current description is updated to {RQa, RQb, RQc, RQd, RQ1, RQ1′, RQ1′.1, EQ1′.2, RQ1′.3}.

>> **RQS update of modification history**
The history is updated by adding

RQS_4 = {RQa, RQb, RQc, RQd, RQ1, RQ1′, RQ1′.1, RQ1′.2, RQ1′.3}
rationale(⟨RQS_3, DOD_0⟩, ⟨RQS_4, DOD_4⟩, method(deletion))
rationale(⟨RQS_3, DOD_0⟩, ⟨RQS_4, DOD_0⟩, has_lower_qualification ({RQ2, RQ2′})).

After analysis shows that no **RQS** manipulation is needed any more, design process coordination comes into action and decides, on the basis of information about the histories of the current **RQS** and the current **DOD**, that the current **DOD** has to be (analysed and) manipulated.

>> **design process evaluation**
The current DOD has to be manipulated, on the basis of all requirements in the current RQS.

>> **update of current requirements**
The current requirements are:

Ra: "The system is able to formulate complaints."
Rb: "The system is able to determine which hypothesis is to be considered."
Rc: "The system is able to evaluate hypotheses."
Rd: "The system is able to observe symptoms."
R1: "The system proposes fewer faulty hypotheses, in comparison with random proposal."
R1′: "The system is able to determine which hypotheses is to be considered in a structured manner."
R1′.1: "Structured determination of hypotheses involves being able to generate hypotheses."
R1′.2: "Structured determination of hypotheses involves being able to compare hypotheses."
R1′.3: "Structured determination of hypotheses involves being able to select hypotheses.".

DOD_0, the specification of the original diagnostic reasoning system with components *CF*, *HD*, *HE*, and *SO*, is analysed and it is noticed that it does not satisfy all current requirements. In particular, the specification of the *HD* component (meant originally to fulfil *R*b) does not fulfil the requirements *R1*, *R1′*, *R1′.1*, *R1′.2* and *R1′.3* (notice that *R1′.1*, *R1′.2* and *R1′.3* are meant to refine *R1′*, which is a default interpretation of *R1*, which implies *R*b, according to the history). A possible solution to resolving this problem is to replace *HD* by a component taken from the library.

>> **DOD modification**

1. **analysis of current description**
The current DOD does not fulfil all current requirements.

2. **modification focus determination**
The focus of modification is set to {HD}.

3. **modification method determination**
The method chosen is *modification based on library consultation*.

4. **modification according to method**
HD is replaced by a composed component capable of structured determination of hypotheses (*libStructD*), with generic subcomponents for generation (*libG*), comparison (*libC*) and selection (*libS*), which are all renamed to tune them to the context of the hypothesis determination task.

>> **DOD update of current description**
The current description is updated to {CF, HE, SO, HD*, HD*:HG, HD*:HC, HD*:HS}.

>> **DOD update of modification history**
The history is updated to

DOD_0 = {CF, HD, HE, SO}
DOD_1 = {CF, HE, SO, HD*, HD*: HG, HD*: HC, HD*: HS}
rationale(⟨RQS_4, DOD_0⟩, ⟨RQS_4, DOD_1⟩, replaced_by (HD, HD*))
rationale (⟨RQS_4, DOD_0⟩, ⟨RQS_4, DOD_1⟩, meant_to_satisfy (HD*, {R1′}))
rationale(⟨RQS_4, DOD_0⟩, ⟨RQS_4, DOD_1⟩, meant_to_satisfy (HD*:HG, {R1′.1}))
rationale(⟨RQS_4, DOD_0⟩, ⟨RQS_4, DOD_1⟩, meant_to_satisfy (HD*:HC, {R1′.2}))
rationale(⟨RQS_4, DOD_0⟩, ⟨RQS_4, DOD_1⟩, meant_to_satisfy(HD*:HS, {R1′.3}))

rationale(⟨RQS4, DOD0⟩, ⟨RQS4, DOD1⟩, method(library_consultation))
rationale(⟨RQS4, DOD0⟩, ⟨RQS4, DOD1⟩, is_based_on (HD*, *libStructD*))
rationale(⟨RQS4, DOD0⟩, ⟨RQS4, DOD1⟩, is_based_on(HD*:HG, *libStructD:libG*))
rationale(⟨RQS4, DOD0⟩, ⟨RQS4, DOD1⟩, is_based_on(HD*:HC, *libStructD:libC*))
rationale(⟨RQS4, DOD0⟩, ⟨RQS4, DOD1⟩, is_based_on(HD*:HS, *libStructD:libS*)).

DOD_1 is analysed and it is noticed that it contains components that are still generic and need domain knowledge to perform their task in the domain of application. It is the knowledge engineer's job to provide this domain knowledge.

>> DOD modification

1. **analysis of current description**
The components HD*, HD*:HG, HD*:HC, HD*:HS are not instantiated; thus, the current DOD is not complete.
2. **modification focus determination**
The focus of modification is set to {HD*, HD*:HG, HD*:HC, HD*:HS}.
3. **modification method determination**
The method chosen is *modification by the KE*.
4. **modification according to method**
The refinements added to the description by the KE are:

HD*:HG$_{inst}$, which is the instantiation of HD*:HG with domain-specific knowledge,
HD*:HC$_{inst}$, which is the instantiation of HD*:HC with domain-specific knowledge,
HD*:HS$_{inst}$, which is the instantiation of HD*:HS with domain-specific knowledge.

>> DOD update of current description

The current description is updated to {CF, HE, SO, HD*, HD*:HG$_{inst}$, HD*:HC$_{inst}$, HD*:HS$_{inst}$}.

>> DOD update of modification history

The history is updated by adding

DOD_2 = {CF, HE, SO, HD*, HD*:HG$_{inst}$, HD*:HC$_{inst}$, HD*:HS$_{inst}$}
rationale(⟨RQS4, DOD1⟩, ⟨RQS4, DOD2⟩, method(KE))
rationale(⟨RQS4, DOD1⟩, ⟨RQS4, DOD2⟩, is_instantiation_of (HD*:HG$_{inst}$, HD*:HG))
rationale(⟨RQS4, DOD1⟩, ⟨RQS, DOD2⟩, is_instantiation_of (HD*:HC$_{inst}$, HD*:HC))
rationale(⟨RQS4, DOD1⟩, ⟨RQS4, DOD2⟩, is_instantiation_of (HD*:HS$_{inst}$, HD*:HS)).

DOD_2 is analysed and it is noticed that manipulation of the current DOD has been successfully accomplished. Therefore, design process coordination becomes active and decides, on the basis of information about the histories of the current RQS and the current DOD, that the current RQS has to be manipulated.

>> DOD modification

1. **analysis of current description**
The current description fulfils all requirements and is complete.

>> design process evaluation

The current RQS is to be manipulated next.

The first results of redesigning the diagnostic reasoning system are shown in Fig. 5.

From this point on, the trace will be continued in an abbreviated form. The global results of only the components for modification, the deductive refinement, the update of modification history, the update of current requirements, and the design process evaluation will be presented.

RQS_4 is analysed and it is noticed that there seem to be no problems. However, satisfying the requirements with less hard qualifications has not yet been tried; this can be concluded from inspecting the history. As a result, it is decided that the soft requirements now also need to be satisfied by the current DOD.

>> RQS modification

1. **analysis of current description**
There seem to be no problems with the current RQS, but requirements with less hard qualifications have yet not been considered.
2. **modification focus determination**
The focus of modification is set to { }.
3. **modification method determination**
The method chosen is *modification by retrieval from history*.
4. **modification according to method**
All requirements with lower qualifications which have been deleted from the

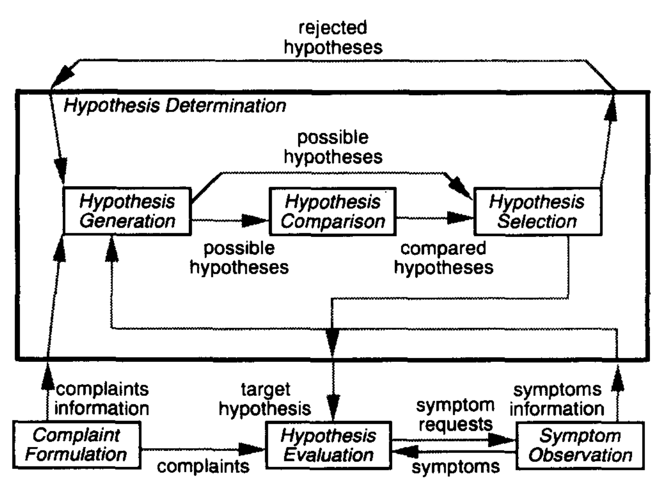

Fig. 5. Results of the first change of the diagnostic reasoning system.

requirement qualification set in the past according to history are collected. In this case, RQ2 and RQ2′.

>> **RQS update of current description**

The current description is updated to {RQa, RQb, RQc, RQd, RQ1, RQ1′, RQ1′.1, RQ1′.2, RQ1′.3, RQ2, RQ2′}.

>> **RQS update of modification history**

The history is updated by adding

RQS_5 = {RQa, RQb, RQc, RQd, RQ1, RQ1′, RQ1′.1, RQ1′.2, RQ1′.3, RQ2, RQ2′}
rationale(⟨RQS_4, DOD_2⟩, ⟨RQS_5, DOD_2⟩, method(retrieval_from_history))
rationale(⟨RQS_4, DOD_2⟩, ⟨RQS_5, DOD_2⟩, has_lower_qualification({RQ2, RQ2′})).

After RQS manipulation has been completed, design process coordination comes into action and decides that the current DOD has to be manipulated.

>> **design process evaluation**

The current DOD has to be manipulated, on the basis of all requirements in the current RQS.

>> **update of current requirements**

The set of current requirements is extended with:

R2: "If the system proposes fewer faulty hypotheses, in comparison with random proposal, then it should also be able to determine a strategy for proposing hypotheses."
R2′: "If the system is able to determine which hypothesis is to be considered, then it is able to determine a strategy for determining hypotheses."

>> **DOD modification**

DOD_2 is analysed and it is noticed that it does not satisfy all requirements. In particular, the component HD* does not fulfil requirement R2′ (which is a default interpretation of R2). Therefore, HD* is *modified by means of library consultation*, so as to include a new generic subcomponent for strategy determination, StratD, which is based on the library component *libStratD*.

>> **DOD update of modification history**

The history is updated by adding

DOD_3 = {CF, HE, SO, HD*, HD*:HG_{inst}, HD*:HC_{inst}, HD*:HS_{inst}, HD*:StratD}.

>> **DOD modification**

DOD_3 is analysed and it is noticed that the component HD*:StratD needs domain knowledge to perform its task in the domain of application. Therefore, this generic component is instantiated by means of *modification by the KE*.

>> **DOD update of modification history**

The history is updated by adding

DOD_4 = {CF, HE, SO, HD*, HD*:HG_{inst}, HD*:HC_{inst}, HD*:HS_{inst}, HD*:$StratD_{inst}$}.

>> **DOD modification**

DOD_4 is analysed: it fulfils all current requirements and is complete.

>> **design process evaluation**

Manipulation of the current DOD has been successfully accomplished. The current DOD satisfies all current requirements. Therefore, the current RQS is to be manipulated next.

The results of the second change to the diagnostic reasoning system are shown in Fig. 6.

>> **RQS modification**

Now all requirements, hard and soft, have been satisfied, the KE is asked whether any further modifications to the current RQS are needed. The result of *modification by the KE* is the addition of the following single requirement plus qualification:

RQ3: "If the system is able to determine a strategy for determining hypotheses, then it should also be able to determine strategies on the basis of

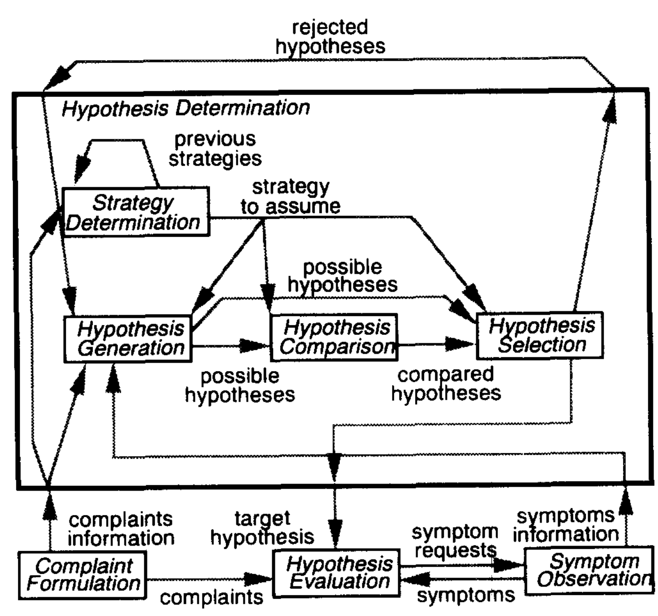

Fig. 6. Results of the second change to the diagnostic reasoning system.

interaction with two agents, viz. system and user." (Hard.)

>> **RQS update of modification history**
The history is updated by adding

RQS_6 = {RQa, RQb, RQc, RQd, RQ1, RQ1', RQ1'.1, RQ1'.2, RQ1'.3, RQ2, RQ2', RQ3}.

>> **RQS modification**
There seem to be no more problems in RQS_6.

>> **design process evaluation**
Manipulation of the current RQS has been successfully accomplished, so therefore the current DOD is to be manipulated, on the basis of all requirements in the current RQS.

>> **update of current requirements**
The set of current requirements is extended with:

R3: "If the system is able to determine a strategy for determining hypotheses, then it should also be able to determine strategies on the basis of interaction with two agents, viz. system and user."

>> **DOD modification**
The component HD*:$StratD_{inst}$ specified in DOD_4 does not fulfil requirement R3. Therefore, HD*:$StratD_{inst}$ is replaced, based on *modification by library consultation*, by a composed component capable of multi-agent strategy determination StratD* (which is based on the library component *libMAStratD*). This composed component contains two subcomponents, one for each of the agents mentioned in requirement R3, each capable of strategic determination: StratD_System and StratD_User (which are both based on the library component *libStratD*).

>> **DOD update of modification history**
The history is updated by adding

DOD_5 ={CF, HE, SO, HD*, HD*:HG_{inst}, HD*:HC_{inst}, HD*:HS_{inst}, HD*:StratD*, HD*:StratD*:StratD_System, HD*:StratD*:StratD_User}.

>> **DOD modification**
DOD_5 is analysed and it is noticed that the subcomponents of HD*:StratD* need domain knowledge to perform their task in the domain of application. Therefore, these generic components are instantiated by means of *modification by the KE*.

>> **DOD update of modification history**
The history is updated by adding

DOD_6 = {CF, HE, SO, HD*, HD*:HG_{inst}, HD*:HC_{inst}, HD*:HS_{inst}, HD*:StratD*, HD*:StratD*:$StratD\_System_{inst}$, HD*:StratD*:$StratD\_User_{inst}$}.

>> **DOD modification**
DOD_6 is analysed. It fulfils all current requirements and is complete.

>> **design process evaluation**
Manipulation of the current DOD has been successfully accomplished: the current DOD satisfies all requirements in the current RQS. Therefore, the current RQS is to be manipulated next.

The third change to the diagnostic reasoning system is shown in Fig. 7

Even though all requirements stated by the KE are satisfied, s/he may again change his/her mind about, or have new ideas regarding, the structure and the behaviour of the compositional architecture for diagnostic reasoning.

>> **RQS modification**
The KE makes no changes to RQS_6.

>> **design process evaluation**
Manipulation of the current RQS has been successfully accomplished, no alterations were made to the current RQS, and

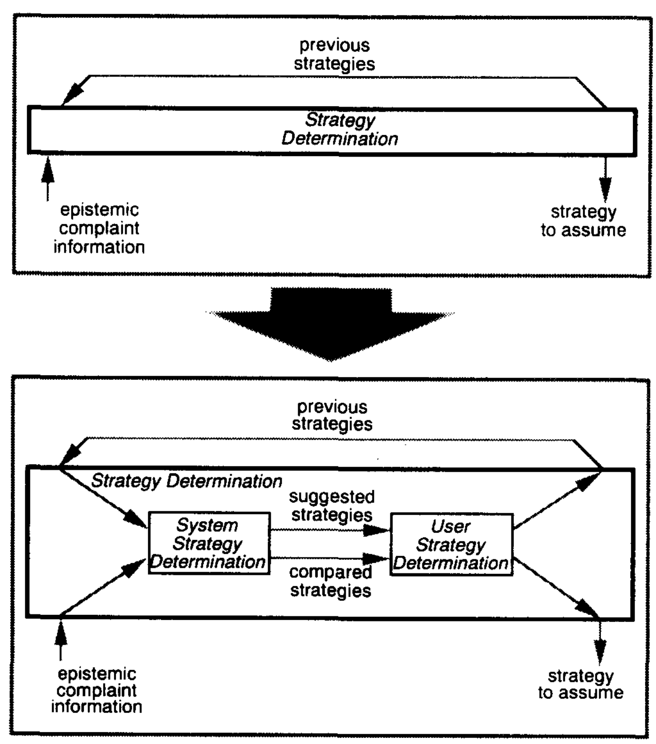

Fig. 7. Third change to the diagnostic reasoning system.

`the current DOD satisfies all requirements of the current RQS. Therefore, the design process can come to an end.`

6. Discussion and conclusions

In this paper it has been shown that our generic task model of design and our conceptual framework for design can be used for redesign, and this has been illustrated for the redesign of compositional knowledge-based systems. To tune the generic task model of design to this particular domain of application, it has been refined (by specialisation and instantiation), resulting in a task model for the redesign of compositional knowledge-based systems.

Redesign, which is in our view an integral part of design, is a dynamic process in which requirements and their qualifications most frequently change in the course of design. The rationale behind redesign is often based on inconsistencies between 'new' requirements and/or their qualifications and one or more existing designs, but it can also be related to new knowledge about the design object domain or design strategies. All three aspects are clearly distinguished in the generic task model of design.

The types of knowledge required to redesign a compositional knowledge-based system, about compositional systems, the requirements of such systems, and the redesign process itself, were introduced. Further research on the way in which redesign systems can take the behaviour of compositional architectures into account is required, in particular with respect to the types of requirements which can be posed.

Hierarchical (de)composition is of central importance within our approach, the formal basis of which has been discussed by Brazier, Treur, Wijngaards and Willems [17,18]. Specialisation and instantiation are of importance both for the design of the redesign system (with respect to the generic task model of design), but also with respect to the compositional architecture to be redesigned. At the level of the compositional architecture, the system to be redesigned, the hierarchical decomposition contains valuable information about the way in which components are related. A number of requirements for refinement and replacement of components in relation to other components are described. The level of abstraction of each component with respect to other components is defined. Knowledge of alternatives and of the design rationale behind previous designs is to a certain extent included in the hierarchical decomposition. Not only does this require further analysis of design rationale, but it also requires further study on the way in which such information can be described and employed (in a way similar to that of, for example, Vanwelkenhuysen and Mizoguchi [20]). The specification of criteria for storage and search in libraries of reusable components, to be consulted by redesign systems, is clearly related.

Current foundations research focuses on the formal semantics of redesign systems, in particular with respect to dynamic aspects of design systems, i.e. transitions in the requirement qualification space and the domain object description space, and also transitions between the spaces [3]. Formal semantics of the dynamics can be used for the development of verification techniques [21].

Acknowledgements

The research described in this paper has been supported in part by the Dutch Foundation for Knowledge Based Systems (SKBS) within the A3 project 'An Environment for Modular Knowledge-Based Systems (Based on meta-Knowledge) for Design tasks', and by NWO-SION within the REVISE project 612-322-316 (Evolutionary Design in Knowledge-Based Systems).

References

[1] F.M.T. Brazier, P.H.G. van Langen and J. Treur, Modelling conflict management in design: an explicit approach. *Technical Report IR-376*, Department of Mathematics and Computer Science, Vrije Universiteit Amsterdam, Netherlands, 1994 (also in *J. Artificial Intelligence, Engineering Design and Manufacturing* (1995) (special issue on conflict management)).

[2] F.M.T. Brazier, P.H.G. van Langen, Z. Ruttkay and J. Treur, On formal specification of design tasks, in J.S. Gero and F. Sudweeks (eds.), *Proc. Artificial Intelligence in Design '94* Kluwer, Netherlands, 1994, pp. 535–552.

[3] F.M.T. Brazier, P.H.G. van Langen and J. Treur, A logical theory of design, *Technical Report IR-374*, Department of Mathematics and Computer Science, Vrije Universiteit Amsterdam, Netherlands, 1994 (also in *Proc. IFIP WG5.2 Second Workshop on Formal Design Methods for CAD, Mexico City*, 1995).

[4] H.A. Brumsen, J.H.M. Pannekeet and J. Treur, A compositional knowledge-based architecture modelling process: aspects of design tasks, in *Proc. Avignon-92: Twelfth Int. Conf. Artificial Intelligence, Expert Systems and Natural Language*, Vol. 1, 1992, pp. 283–294.

[5] P.A. Geelen and W. Kowalczyk, A knowledge-based system for the routing of international bank payment orders, in *Proc. Avignon-92 Twelfth Int. Conf. on Artificial Intelligence, Expert Systems and Natural Language*, Vol. 2, 1992, pp. 669–677.

[6] R.D. Coyne, M.A. Rosenman, A.D. Radford, M. Balachandran and J.S. Gero, *Knowledge-Based Design Systems*, Addison-Wesley, USA, 1990.

[7] T. Smithers, 'On the nature of theory and design, in T. Smithers (ed.), *Workshop Notes of AID '94 Workshop on the Role and Nature of Theory in AI in Design Research*, 1994.

[8] G.R. Yost, Configuring elevator systems, *Technical Report Stanford Version 1.3* Medical Computer Science Group, Knowledge Systems Laboratory, Stanford University, USA, 1994.

[9] F.M.T. Brazier, P.H.G. van Langen, J. Treur, N.J.E. Wijngaards and M. Willems, Modelling a design task in DESIRE: the VT example, *Technical Report IR-377*, Department of Mathematics and Computer Science, Vrije Universiteit Amsterdam, Nether-

lands, 1994 (also in *Int. J. Human–Computer Studies* (1995) (special issue on Sisyphus)).

[10] W. Kowalczyk and J. Treur, On the use of a formalized generic task model in knowledge acquisition, in B.J. Wielinga, J.H. Boose, B.R. Gaines, A.T. Schreiber and M.W. van Someren (eds.), *Current Trends in Knowledge Acquisition: Proc. EKAW '90: European Knowledge Acquisition Workshop*, IOS Press, Netherlands, 1990, pp. 198–221.

[11] D.C. Brown and B. Chandrasekaran, *Design Problem Solving: Knowledge Structures and Control Strategies*, Pitman, UK, 1989.

[12] B. Chandrasekaran, Design problem solving: a task analysis, *AI Magazine*, *11*, (1990) pp. 59–71.

[13] J. Breuker (ed.), Model driven knowledge acquisition: interpretation models, *Deliverable A1*, ESPRIT Project 1098, 1987.

[14] I.F.C. Smith and S. Boulanger, Knowledge representation for preliminary stages of engineering tasks, *Knowledge-Based Systems*, *7* (1994) pp. 161–168.

[15] J. Engelfriet and J. Treur, Temporal theories of reasoning, in C. MacNish, D. Pearce and L.M. Pereira (eds.), *Logics in Artificial Intelligence: Proc. JELIA '94: Fourth European Workshop on Logics in Artificial Intelligence* Springer-Verlag, 1994, pp. 279–299 (also to appear in *J. Applied Non-Classical Logic* (special issue of selected papers from JELIA '94).

[16] C.J. Petrie, M.R. Cutkosky and H. Park, Design space navigation as a collaborative aid, in J.S. Gero and F. Sudweeks (eds.), *Proc. Artificial Intelligence in Design '94* Kluwer, Netherlands, 1994, pp. 611–623.

[17] F.M.T. Brazier, J. Treur, N.J.E. Wijngaards and M. Willems, A formalization of hierarchical task decomposition, in M. Aben, D. Fensel, F. van Harmelen and M. Willems (eds.), *Proc. ECAI'94 Workshop on Formal Specification Methods for Knowledge Based Systems*, 1994, pp. 97–112.

[18] F.M.T. Brazier, J. Treur, N.J.E. Wijngaards and M. Willems, Temporal semantics and specification of complex tasks, *Technical Report IR-375*, Department of Mathematics and Computer Science, Vrije Universiteit Amsterdam, Netherlands, 1994 (also in J.C. Bioch (ed.), *Proc. NAIC '95: Dutch Artificial Intelligence Conf.*, Erasmus University, Netherlands, 1995.

[19] I.A. van Langevelde, A.W. Philipsen and J. Treur, 'Formal specification of compositional architectures, in B. Neumann (ed.), *Proc. ECAI '92: 10th European Conf. Artificial Intelligence*, John Wiley, UK, 1992, pp. 272–276.

[20] J. Vanwelkenhuysen and R. Mizoguchi, Workplace-adapted behaviours: lessons learned for knowledge reuse, *Proc. Second Int. Conf. Building and Sharing Very Large-Scale Knowledge Bases, Enschede, Netherlands, 1995.*

[21] J. Treur and M. Willems, On verification in compositional knowledge-based systems, in A. Preece (ed.), *Proc ECAI '94 Workshop on Validation of Knowledge-Based Systems* 1994, pp. 4–20 (updated version in M.-C. Rousset and M. Ayel (eds.), *Proc. EUROVAV '95: European Symposium on Validation and Verification of KBSs, Chambéry, France, 1995.*