Int. J. Human-Computer Studies (2000) **52,** 77–110
Article No. ijhc.1999.0322
Available online at http://www.idealibrary.com on IDEAL®

On the use of shared task models in knowledge acquistion, strategic user interaction and clarification agents†

FRANCES M. T. BRAZIER, CATHOLIJN M. JONKER, JAN TREUR AND NIEK J. E. WIJNGAARDS
Department of Artificial Intelligence, Vrije Universiteit Amsterdam, De Boelelaan 1081a, 1081 HV Amsterdam, The Netherlands email: *{frances, jonker, treur, niek}@cs.uv.nl*

(*Received 2 December 1998 and accepted 7 June 1999*)

In this paper, three different roles of a shared task model as an intermediate representation of a task are presented and illustrated by applications developed in cooperation with industry. First the role of a shared task model in knowledge acquisition is discussed. In one of the two applications, decision support in the domain of soil sanitation, one of the existing generic task models for diagnostic reasoning provided a means to structure knowledge acquisition. In the second application, diagnosis of chemical processes, the acquisition process resulted in a shared task model for diagnostic reasoning on Nylon-6 production. Secondly, the role of a shared task model in designing user interaction is addressed. Three levels of interaction are considered of importance: interaction at the object level, at the level of strategic preferences, and at the level of task modification. In an application in the domain of environmental decision making, this led to the design of a user interface based on the acquired shared task model, within which all three levels of interaction were available to users. Finally, the role of shared task models within a multi-agent system including a clarification agent is addressed. Two software agents were designed that each share a task model with the user: one for a diagnosis task, and one for a clarification task. The shared model of the clarification task reflects the shared task model of diagnosis; clarification includes clarification of the overall diagnostic reasoning process. The multi-agent architecture presented has been developed to support a user both at the level of the diagnostic task he or she is performing and at the level of clarification. The architecture has been applied to the diagnosis of chemical processes.

1. Introduction

Decision support systems are most often designed to provide expert users with the information they need to solve a problem. More extensive support, however, is provided by knowledge-based systems that not only are capable of performing complex computation but that also are equipped with explicit knowledge of the decision process. The acquisition of such knowledge is not as trivial as it may seem. Although experts differ in their approaches to problems, in almost all situations different alternatives are thought

† Preliminary work on different parts of this paper have been presented earlier in the following workshop and conferences: Section 3 in Brazier, Treur and Wijngaards (1996*a*), Section 4 in Brazier, Treur and Wijngaards, (1996*b*) and Section 5 in Jonker and Treur (1998).

1071-5819/00/010077 + 34 $35.00/0

through and compared. Decision support systems ideally support experts in this process. Not only the opportunity to influence the approach taken by systems (for example the sequence of tasks) is of importance, but also the opportunity to influence the more local levels of strategic reasoning involved in decision-making processes.

User centred task analysis is essential to the design of such systems (Barnard, 1993; Brazier & Treur, 1994). The tasks users perform in specific decision-making situations must be identified, in addition to relations between tasks. The designer of a system (in general a knowledge engineer) and one or more experts must reach a common understanding of the tasks involved in a specific decision-making process. The types of decision an expert user would perfer to make him/herself and the ways in which an expert user would like to be able to influence a system's reasoning, must be identified.

The common understanding between knowledge engineers and expert users can be supported by a *shared task model*. A shared task model can play a different role in different activities.

- During knowledge acquisition, a shared task model is a means to acquire a common understanding of a task in interaction with experts.
- During design of a system, a shared task model can be used (1) as a basis for user interaction, and also (2) as a basis for the design of a clarification support agent.

In this paper, all the three roles will be described in detail and illustrated by examples that have been tested. In Section 2, approaches to knowledge acquisition and models of knowledge-based and multi-agent systems are described. In Section 3, it is shown how shared task models have been developed for two domains: decision support for soil sanitation and diagnosis of Nylon-6 processes. During an acquisition process, different types of interaction between expert users and a system designed to support such users, can be identified. In Section 4, an example is shown of a user interface based on a shared task model in the domain of decision support for environmental decision making. In Section 5, shared task models are applied within the multi-agent system paradigm: a clarification agent is constructed. In Section 6, the results are discussed.

2. Knowledge acquisition

To structure the exchange of knowledge between a knowledge engineer and an expert user, often mediating representations are used (e.g. Ford, Bradshaw, Adams-Webber & Agnew, 1993). From our perspective, one of the results of knowledge acquistion (and task analysis) is a shared task model: a model which both the knowledge engineer and one or more expert user(s) agree to be an acceptable representation of the task structure for which support is to be provided.

2.1. A COMPOSITIONAL APPROACH

The compositional development method DESIRE (Design and Specification of Interacting Reasoning components; cf. Brazier, Dunin-Keplicz, Jennings & Treur, 1997; Brazier, Jonker & Treur, 1998) is a development method for single- or multi-agent systems. The development method is supported by graphical design software. Translation to an operational system is straight-forward; in addition to the graphical design tools, the software environment includes implementation generators with which specifications can be translated into executable code of a prototype system. Within this development method knowledge of the following three types is distinguished.

1. *Process composition*
   — Identification of processes at different process abstraction levels
      • input and output information types of each of the processes
      • process abstraction hierarchy
      • task delegation.
   — Process composition relation
      • information exchange
      • task control.
2 *Knowledge composition*
   — Identification of knowledge structures at different knowledge abstraction levels
      • information types
      • knowledge bases
      • task delegation.
   — Knowledge composition relation
      • composition of information types
      • composition of knowledge bases
      • relations between knowledge bases and information types.
3. *Relation between process composition and knowledge composition.*

Process composition identifies the relevant processes at different levels of (process) abstraction, and describes how a process can be defined in terms of (is composed of) lower-level processes.

Knowledge composition identifies the knowledge structures at different levels of (knowledge) abstraction, and describes how a knowledge structure can be defined in terms of lower-level knowledge structures. The knowledge abstraction levels may correspond to the process abstraction levels, but this is often not the case.

Each process in a process composition uses knowledge structures. Which knowledge structures are used for which processes is defined by the relation between process composition and knowledge composition.

2.2. TASK MODELS

A shared task model, as a mediating representation, is the result of negotiation between a knowledge engineer and one or more experts. The purpose of the negotiation is to acquire a common understanding of the task. An expert has extensive (often implicit) knowledge of a domain and of his/her task and strategies. A knowledge engineer has knowledge of existing models of related tasks which may or may not be applicable, and of ways to modify and combine such models for the domain at hand. Abstract task models are often used to structure the knowledge acquistion process. Task models are similar to models resulting from task analysis approaches such as KAT/KTS (Johnson & Johnson, 1991, 1993).

Within DESIRE (Brazier, Dunin-Keplicz, Jennings & Treur, 1997; Brazier, Jonker & Treur, 1998), a number of such abstract task models, generic task models, exist which are used for this purpose. These models have been defined on the basis of experience and logical analysis. The concept of a generic task, introduced by Chandrasekaran (1986, 1990) and Brown and Chandrasekaran (1989), is comparable to the notion of *generic task model* in that both of these are generic with respect to domains. Generic task models within the DESIRE framework, however, are generic with respect to both tasks and domain: generic task models can be refined with respect to the task by *specialization* (e.g. further decomposition of a sub-task) and refined with respect to the domain by *instantiation* (e.g. addition of domain-specific knowledge). Moreover, the way a generic task model is specified in DESIRE is more declarative (with semantics based on temporal logic, cf. Brazier, Treur, Wijngaards & Willems, 1999) than the way generic tasks are described in Chandrasekaran (1986, 1990) and Brown and Chandrasekaran (1989). The integral approach to levels of abstraction within the DESIRE framework supports the use of generic task models during knowledge acquistion. Different levels of abstraction and composition play a role during the negotiation phase.

3. Shared task models in knowledge acquisition

The use of shared task models in knowledge acquisition will be illustrated by examples in the area of diagnosis. In most situations in which diagnosis is required not all relevant facts are known in advance. In practice, in fact, diagnosis is not often based on complete information. The acquisition of additional (observation) information is an essential part of most diagnostic processes. In general, diagnosis includes a number of sub-processes such as the determination of hypotheses, the choice of applicable observations, the performance of observations, and the interpretation of observation results. Strategic considerations such as the suitability of an observation, the likelihood of a hypothesis being true, and the cost and effect of an observation, play an important role in these processes. A number of existing (generic) task models for diagnostic reasoning in which strategic knowledge is explicitly modelled are described in this section. These models are used in interaction with experts to structure acquisition of shared task models of diagnostic reasoning for specific domains of application.

In Section 3.1, a generic model for diagnostic reasoning is described. Two specializations of this model are presented in Sections 3.2 and 3.3. The process of knowledge acquisition is illustrated for two domains in Sections 3.4 and 3.5.

3.1. THE GENERIC TASK MODEL SIX FOR DIAGNOSTIC REASONING

As described above, shared task models are acquired in interaction with experts, using existing (generic) task models to structure the process of knowledge acquisition. A generic task model of diagnostic reasoning designed for this purpose (Treur, 1993; Brazier & Treur, 1994) is shown in Figure 1.

In this model five tasks are distinguished: `hypothesis determination`, `hypothesis validation`, `observation determination`, `hypothesis evaluation` and `diagnostic process coordination`. The task `hypothesis determination` reasons the appropriateness of possible hypotheses within a given state of the diagnostic process and determines which hypotheses are to be further investigated. The task `hypothesis validation` validates the hypotheses by determining appropriate observations and evaluating hypotheses on the basis of results

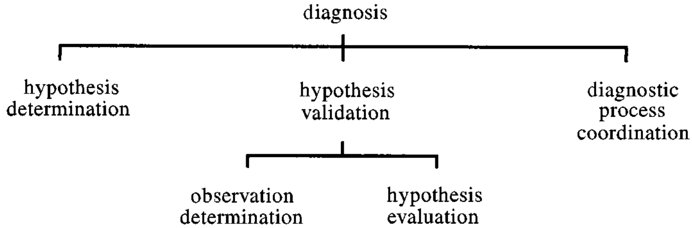

FIGURE 1. Tasks within SIX: a generic task model of diagnostic reasoning.

of performing these observations. The task `observation determination` analyses the current state of the diagnostic process with respect to observation performance and determines which observations are the most appropriate. The task `hypothesis evaluation` performs the observations, and determines the relation between the observation results and the current hypotheses. The task `diagnostic process coordination` analyses the implications of the observation results for the hypotheses and determines which hypotheses are rejected and which are confirmed. On the basis of an analysis of the current overall state of the diagnostic process, the decision to conclude the diagnostic process may be made. If, however, the diagnostic process is continued, the required subsequent processes (for example, determination of hypotheses or observations) are identified.

Diagnostic reasoning processes can be based on *causal* or on *anti-causal* domain knowledge. In the first case, derivations about the domain follow the direction of causality: the predicted observable consequences are derived from hypotheses (possible causes), after which (some of) the predicted observations are verified. For this type of reasoning a causal knowledge is required that specifies how the causal consequences of hypotheses can be derived (e.g. represented by a causal network).

In the second case, the domain knowledge is used to derive hypotheses from information on observable (symptoms). Here the direction of derivation is against the direction of causality: it proceeds from observable findings (in particular, those that actually were observed) to the causes. For this type of reasoning, knowledge is required that specifies how hypotheses can be derived from observable findings: this type of knowledge is called anti-causal knowledge.

In both cases strategic reasoning is required to determine the appropriate hypotheses on which to focus and the appropriate observations to be performed, as modelled by the generic task model for diagnostic reasoning SIX described above. This generic task model can be refined by specialization to two slightly different models for diagnostic reasoning based on anti-causal domain knowledge and causal domain knowledge, respectively. These specializations are described in Sections 3.2 and 3.3

3.2. A SPECIALIZATION OF THE GENERIC TASK MODEL FOR ANTI-CAUSAL DIAGNOSTIC REASONING

The specialization for diagnostic reasoning based on anti-causal domain knowledge is obtained by decomposing the task `hypothesis evaluation` into two sub-tasks: `observation performance` and `observation result interpretation`, as shown in Figure. 2.

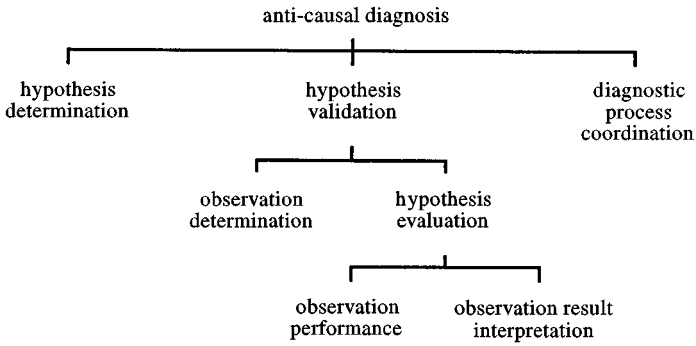

FIGURE 2. Tasks within a task model for diagnostic reasoning based on anti-causal knowledge.

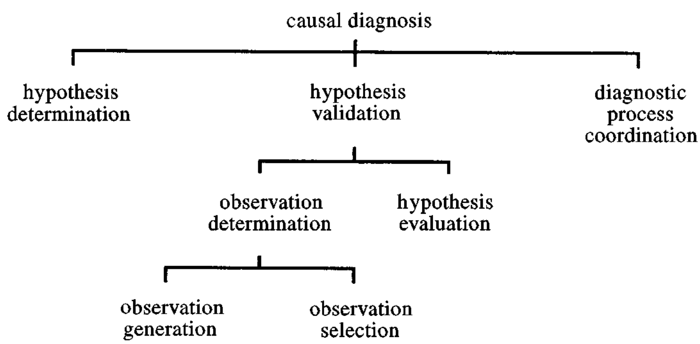

FIGURE 3. Tasks within a task model for diagnostic reasoning based on causal knowledge.

The task `observation performance` is responsible for the "execution" of the observations selected by the task `observation determination`. The results of the observation may be acquired directly by object-level interaction with an expert user, or may be acquired automatically from other systems. No further reasoning about the domain is performed in this task. The acquired observation information is used by the task `observation result interpretation` to draw conclusions about the hypotheses, by means of the available anti-causal domain knowledge.

3.3. A SPECIALIZATION OF THE GENERIC TASK MODEL FOR CAUSAL DIAGNOSTIC REASONING

The specialization for diagnostic reasoning based on causal domain knowledge is obtained by decomposing the `observation determination` task into two subtasks: `observation generation` and `observation selection`, as shown in Figure 3.

The task `observation generation` takes the hypothesis on which it is focused as its input and using causal domain knowledge observable causal consequences are derived. These observable causal consequences are predictions of the findings that should be observed if the hypothesis holds. The predicted findings, influenced by the assumed hypothesis, are suitable candidates for observations to be selected. The task `observation selection` analyses the candidates and selects one or more observations on the basis of this analysis.

3.4. A SHARED TASK MODEL FOR SOIL SANITATION

One domain in which a shared task model was developed in interaction with experts is the domain of soil sanitation (Boelens, 1991). During the acquisition process, the generic task model of diagnostic reasoning (SIX) presented above played an important role. This model was used for structure interaction with the experts. In this section, the domain of soil sanitation is introduced, an indication of the required functionality of a support system is given and finally the acquisition of the shared task model is described.

3.4.1. Soil sanitation

Soil sanitation is a relatively young but fast-growing area of expertise. Polluted soil is found in many locations (in the Netherlands at least several thousand) and depending on the severity of the pollution the soil may need to be sanitized. At the level of provincial and local authorities the problem of soil sanitation usually is encountered during urban renewal. Pragmatic solutions are often chosen. Such solutions are based on two major decisions: how the site is to be sanitized and how the soil can be disposed.

Several procedures have been formulated concerning soil sanitation. Inventory research provides an indication of the different types of contaminations. Initial investigations aim to provide a global insight into the nature and concentrations of the contaminants. Further investigation concentrates on the nature, extent and concentrations of the contaminations as well as the spreading-probabilities. The goal of these investigations is to provide enough information for the sanitation procedure. The sanitation procedure consists of a comparison of the possible sanitation alternatives on environmental, technical, and financial aspects. Sanitation is planned and executed.

The domain of sanitation consists of types of contamination found (heavy metals, cyanide, aliphatic or hydrocarbons, aromatic compounds and volatile helogenic hydrocarbons) and types of soil (sandy, loamy, loamy and clayey, peaty and mixed). These types are only top-levels of taxonomies. Possible (general) sanitation techniques are: removing the contamination, prevent spreading of the contamination (isolation), or changing the function of the site. When removing the contamination either the soil is not removed (*in situ* techniques) or the soil is dug up. A soil sanitation alternative is a plan: one or more pollution remedial techniques are applied to the polluted site.

3.4.2. Acquisition of a shared task model for soil sanitation

Experts working in the domain of soil sanitation were aware of the need for more support in choosing the best soil sanitation alternative. Although large bodies of knowledge are available, the experts lacked support for flexible use of that knowledge. Ideally, the experts should be able to influence the use of the knowledge: when should the knowledge be used and what sanitation alternatives or observations may be investigated.

The following knowledge was readily available in pre-defined procedures and/or algorithms.

- How to choose remedial techniques based on their technological features and the situation at the polluted site.
- How to combine pollution remedial techniques with sanitation alternatives.
- How to predict the results of sanitation alternatives, based on the situation at the polluted site.
- How to compare sanitation alternatives to environmental standards or constraints.
- How to weigh the various (groups of) evaluation criteria for sanitation alternatives.
- How to perform sensitivity analysis to determine which type of additional investigations is most effective with regard to selecting the best alternative.

Initial analysis of the experts' reasoning process to find the best alternative for a specific situation given the option to collect additional information about the situation, is, in fact, a form of diagnosis. Experts agreed that this generic task model (the generic task model for diagnostic reasoning, described in Section 3.1) provided a basis for subsequent discussion. The mapping of the terminology in the domain to the terminology in which the generic task model is presented, was relatively straight-forward: sanitation alternatives in this domain are hypotheses, performance of "additional investigation" is the performance of observations, and acquired information corresponds to observation results. The resulting task decomposition is shown in Figure 4.

3.4.2.1. Hypothesis determination. During knowledge acquisition it became clear that the determination of the most appropriate sanitation alternatives, should be seen as two separate tasks. The first is the determination of the most appropriate technique for the reduction of one or more pollutants at a polluted site (`remedial techniques`

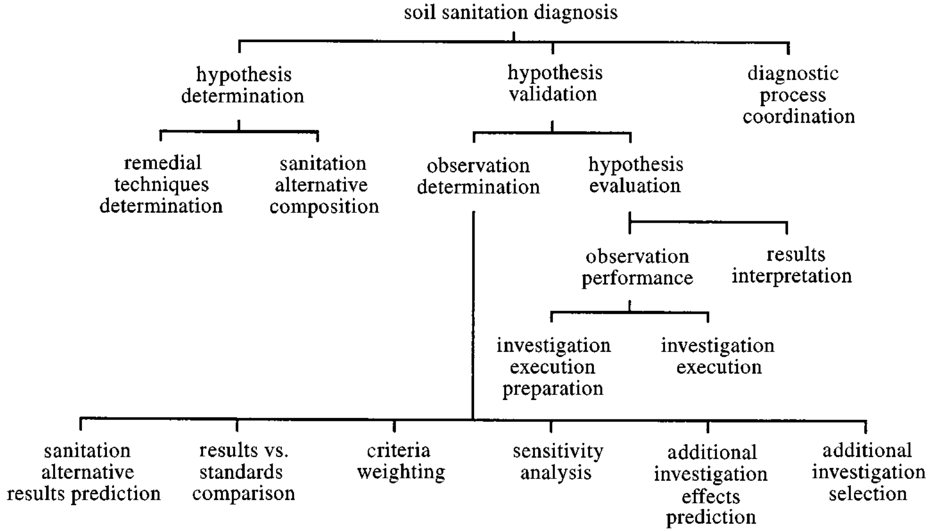

FIGURE 4. Task decomposition of soil sanitation diagnosis.

`determination` task). The second is the formulation of alternatives (i.e. hypotheses) on the basis of the available remedial techniques (`sanitation alternative composition` task).

3.4.2.2. Observation determination. The most extensive refinement of the generic task model was made with respect to `observation determination`. Different sub-tasks were identified using different types of knowledge (including the knowledge mentioned above) to determine the most appropriate observation.

Predictions about the (expected) reductions of pollutants at a particular site, are made using the available knowledge mentioned above (`sanitation alternative results prediction` task). Available knowledge is also used to determine the goodness of reductions, measured against directives on soil sanitation and construction materials (`results vs. standards comparison` task). Separate knowledge is available to decide how important different criteria are in the evaluation of sanitation alternatives (`criteria weighting` task).

Experts employ sensitivity analysis to determine which observations are most interesting in view of existing uncertainties (`sensitivity analysis` task). The knowledge experts have of models to predict the effect of observations on the criteria was also identified as a separate task (`additional investigation effects prediction` task). On the basis of the knowledge obtained by the performance of the above-mentioned sub-tasks, a decision is made as to which additional investigation should be performed taking into consideration cost and duration (`additional investigation selection` task).

3.4.2.3. Observation performance. Before actually performing observations experts reason about the information they expect to acquire and the way in which additional investigations should be performed (`investigation execution preparation` task). This task is distinguished from the actual performance of the additional investigations (`investigation execution` task).

3.4.2.4. Result interpretation. Further decomposition of `results interpretation` was not necessary: the results of the observations are interpreted.

3.4.2.5. Diagnostic process coordination. Experts recognized the appropriateness of a task for evaluation of the status of the process. On the basis of this analysis, experts decide whether to pursue further analysis of a situation, or not.

3.5. SHARED TASK MODELS IN DIAGNOSIS OF CHEMICAL PROCESSES

In a completely different domain, namely the domain of nylon production, the same SIX model was used during knowledge acquisition to structure discussions with an expert in this field. The expert involved identified the need for a system to support him in the diagnostic process, hopefully reducing the need for frequent on-site diagnosis. The nylon production process was described in detail and a few examples of types of problems with which the expert is confronted were discussed. As it was unclear how, in general, the expert structured his process of diagnosis, the two specializations of the SIX task model described in Sections 3.2 and 3.3: the causal and the anti-causal diagnosis task models, were introduced.

Initially, the process of nylon production, in principle, is based on causal knowledge in the domains of physics and chemistry, the knowledge engineers involved expected the diagnostic process to be based on causal reasoning. The task model for causal diagnostic reasoning was introduced. Further discussion and analysis of cases of diagnostic processes, however, showed that during the diagnostic process in this domain hypotheses themselves could be confirmed or rejected on the basis of direct observation, i.e. no causal or anti-causal knowledge at all was required. In addition, cases were identified in which hypotheses which could not be confirmed or rejected on the basis of direct observation, played an important role. In these cases, the expert used anti-causal knowledge to derive hypotheses from observed findings. At this point the task model for anti-causal diagnostic reasoning was introduced. The two models were compared, and the expert concluded that, in general, the anti-causal model was most applicable even though he realized that in some, exceptional (and complicated) situations, the causal model would be more applicable (in which observable findings are derived from hypotheses). The task `hypothesis determination` was further refined: a (limited) number of possible hypotheses are first identified, one of which is chosen for further examination. The first task is delegated to the system, the second to the expert user. By modelling the task in this way, the expert user explicitly and directly influences the reasoning process. The need for such strategic interaction was identified during the knowledge acquisition process.

The shared task model designed for diagnostic Nylon-6 production process is a specialization of the generic task model for diagnostic reasoning based on anti-causal knowledge, presented in Section 3.3. The first version of a system for diagnosis of Nylon-6 production processes based on this model, has been implemented and is currently being evaluated. In other domains in the same chemical plant, the causal model has shown to be more applicable.

4. Designing user interaction on the basis of a shared task model

A shared task model can be used to design the interaction with the user. In the case of expert users, different levels of interaction can be distinguished. A system designed, developed and implemented for the RIVM (the Dutch Research Institute for Environmental Studies) within SKBS (Foundation for Knowledge-Based Systems) for environmental decision support is used not only to illustrate the knowledge included in a shared task model, but also to demonstrate how the model is used to structure interaction with the user.

In Section 4.1, levels of interaction are described which can be modelled via a shared task model. In Section 4.2, the generic task model of design is described, which is used to develop a shared task model for environmental decision making, described in Section 4.3. In Section 4.4, the levels of interaction in the shared task model for environmental decision support are discussed.

4.1. LEVELS OF STRATEGIC INTERACTION

Within the context of a given task often specific sub-tasks may be assigned to either the expert user or the system. For example, the system may discover that certain information, required to be able to provide an answer to a specific user's request on a related issue, is unknown to the system. This is the case when, for example, the user has not yet made specific facts about a current problem known to the system. This type of interaction, *object-level interaction*, in which one of the parties (often the user) is requested to provide facts of this type, is not uncommon to knowledge-based systems.

Interaction between the expert user and the system is, however, often of a slightly different nature. In design and decision-making processes expert users frequently wish to influence the factors on which designs/decisions are based: the goals, the heuristics employed, preferences, assumptions, using the system to explore the results of different strategies. Interaction at this level, *the level of strategic preferences*, is not uncommon within the tasks examined (Brazier, Langen & Treur, 1998), but is not often included in knowledge-based system design.

Although a shared task model is the result of interaction with an expert user, it is not necessarily "the" correct model of a task for all problems in all domains. The expert user may want to be able to influence, for example, the sequencing or choice of sub-tasks in a particular situation. The design/decision support system with which the user interacts should make this possible. This is not only of importance for the individual expert for whom a system may have been designed, but also for other experts users (often the expert involved in the design of a system represents a class of experts for which the system is designed) for whom the model can be seen as a model of consensus. This model may need to be adapted for individual experts. This level of interaction has been termed *the level of task model modification.*

4.2. A GENERIC TASK MODEL OF DESIGN

This section briefly explains the generic task model of design (referred to as GTMD) introduced by Brazier, Langen, Ruttkay and Treur (1994), and described in Brazier, Langen, Treur, Wijngaards and Willems (1996) for (parametric) design of elevators. GTMD is used to structure the acquisition process, clearly distinguishing three sub-tasks, each focusing on reasoning about one of the following three aspects.

- Requirements and preferences.
- The design object.
- (Coordination of) the overall design process.

This model is generic not only in the sense that it describes the design independent of the domain of application for the design task [in the same sense as Brown and Chandrasekaran (1989), and Chandrasekaran (1986, 1990)], but also in the sense that it describes design at a high level of abstraction. GTMD clearly specifies a role for design history and design rationale within each of the three sub-tasks of design. Figure 5 shows a simplified picture of GTMD. The figure shows the component `design`, which is a model of the design task as a whole. The design task is composed of three sub-tasks: design process coordination, requirement qualification set manipulation and design object description manipulation, which are modelled by the components `design process coordination`, `RQS manipulation` and `DOD manipulation`, respectively. In the sequel, *requirement qualification set* will often be abbreviated as *RQS* and *design object description* as *DOD*.

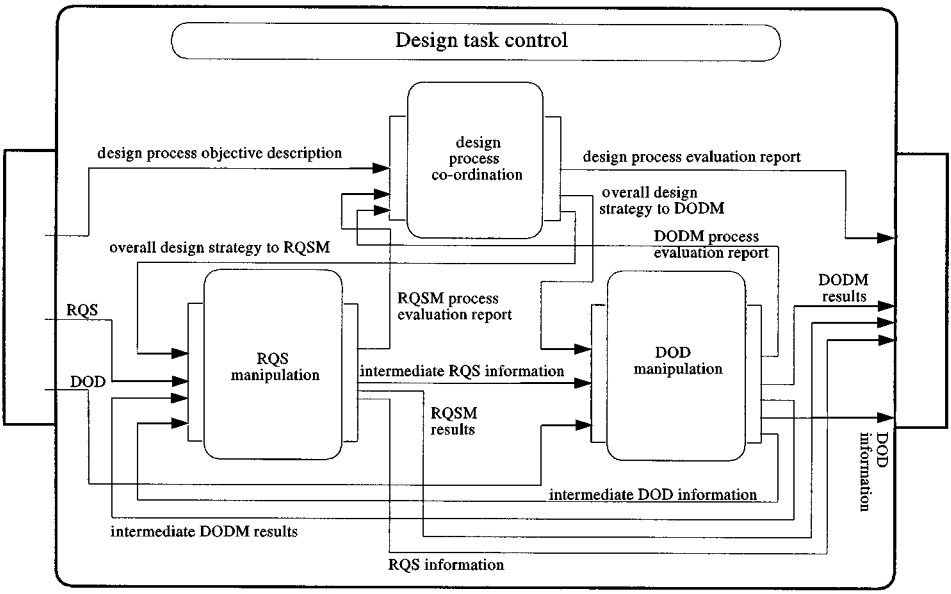

FIGURE 5. Top level of generic task model of design.

The task of *design process coordination* is to determine the future course of the design process: should the design process continue or not, and if so, according to which strategy. For example, after `RQS manipulation` has delivered a new RQS, design process coordination may propose to proceed with `DOD manipulation` in order to make a DOD that satisfies the current requirements, taking a specific earlier design as a starting point.

The task of *RQS manipulation* is to generate an RQS based on the needs and desires of, for example, a client. Part of the task is to analyse whether the current RQS includes (apparently) conflicting requirements and whether all requirements it includes are sufficiently refined to be used as input for `DOD manipulation`. Another part is to resolve any conflicts that are detected and to refine any requirements that need to be further detailed.

The task of *DOD manipulation* is to generate a DOD. A DOD is most frequently a partial design description, that often contains inconsistent information and often does not satisfy (or may even violate) the requirements on which the design process is focussed. However, the ultimate goal is to acquire a DOD that is complete, consistent and satisfactory. A DOD is *complete* if it contains all the information that is relevant to the process of creating the design object (through assembly, construction, fabrication, manufacturing or any other form of implementation). A DOD is *consistent* if it does not contain contradictory information. A DOD is *satisfactory* with respect to an RQS if it satisfies each of the requirements in the RQS (e.g. the requirements on which the RQS focusses).

4.3. A SHARED TASK MODEL FOR ENVIRONMENTAL DECISION MAKING

The task of constructing combinations of possible political environmental measures which can be taken to reduce specific types of pollution, is a task with which most governments are faced. In general, this task is done "by hand" using models with which the result of a particular measure on emission can be calculated. These systems most often do not support the process of choosing and combining (sets of) measures which together should reach a given goal.

Within the SKBS project (supported by RIVM and the Dutch Ministry of Economic Affairs) the decision making/design process entailed in this particular task has been analysed and modelled. In interaction with policy makers, a shared task model has been developed for the task of designing, for a given set of goals (with respect to the reduction of future emissions of polluting matters; e.g. NO_2), a set of environmental measures for processes in, e.g. metal industry, oil refineries, traffic, agriculture; these processes form a taxonomy. This shared task model and its role in the design of interaction with the expert user will be discussed below. Based on the knowledge contained in this shared task model and further analysis the interactive decision support system SENSE has been developed.

4.3.1. The shared task model: task decomposition and delegation

Based on the generic task model for design tasks, a shared task model was developed and used in communication with experts in this field. The main tasks for environmental policy making as distinguished in the generic task model of design are the following

- *Acquire problem statement*. To acquire the user's goals, in terms of overall emissions for specific polluting matters.
- *Manipulate requirements qualification sets*. To determine requirements (and their qualifications) at different levels of abstraction on future emission for the various processes.
- *Manipulate design object descriptions*. To determine sets of measures for different processes.
- *Design process coordination*. To coordinate the overall design strategy.

The shared task model includes more detailed knowledge of each of these main tasks and the delegation of sub-tasks. A pictorial representation of decomposition and information exchange, as shown in Figure 6, provides the basis for the user interface of the SENSE system.

4.3.2. Acquire problem statement

To acquire the problem statement a user specifies a list of desired reductions for matters distributed over different regions (see Figure 7). The system completes this list (by adding desired reductions on processes that are logically implied by desired reductions specified by the user). The following two tasks are relevant to this process.

- Determine problem statement specification (user).
- Refine problem statement (system).

4.3.3. Manipulate requirement qualification sets

The task of defining priorities and relations between requirements (requirement qualifications) involves the user specifying the extent to which each of the processes involved

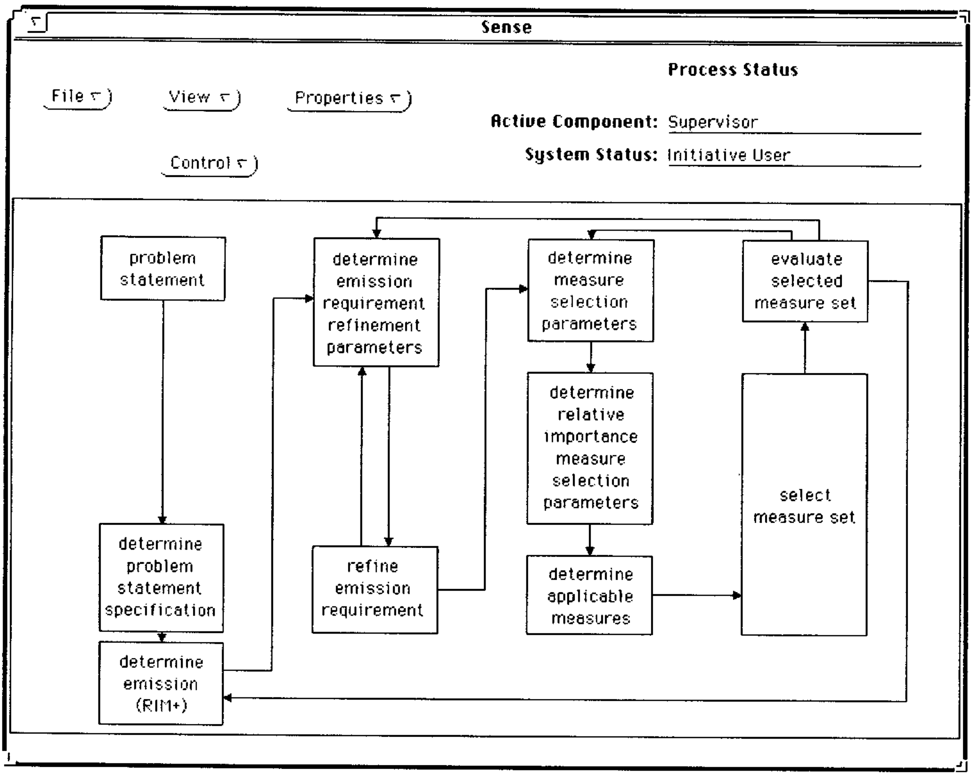

FIGURE 6. User interface including the shared task model (translated from Dutch).

should contribute to the required reduction. The system uses these parameters to determine a refinement (distribution) of the emission requirements to the lower levels of processes in the taxonomy. Furthermore, the user specifies parameters to be used for the measure set selection: costs, future expandability, social acceptability. The system calculates relative measure selection parameters from these user-specified parameter (by normalization). The tasks distinguished in this context are the following

- Determine emission requirement refinement parameters (user).
- Refine emission requirements (system).
- Determine measure selection parameters (user).
- Determine relative measure selection parameters (system).

4.3.4. Manipulate design object descriptions

Given a set of requirements a set of measures needs to be found. The system determines which measures are applicable. Based on the applicability of the measures, and the measure selection parameters determined in the previous tasks, the system selects a set of measures. This set is presented to the user who can then indicate whether he/she accepts or rejects the proposed set of measures. If the user accepts the proposed set of measures

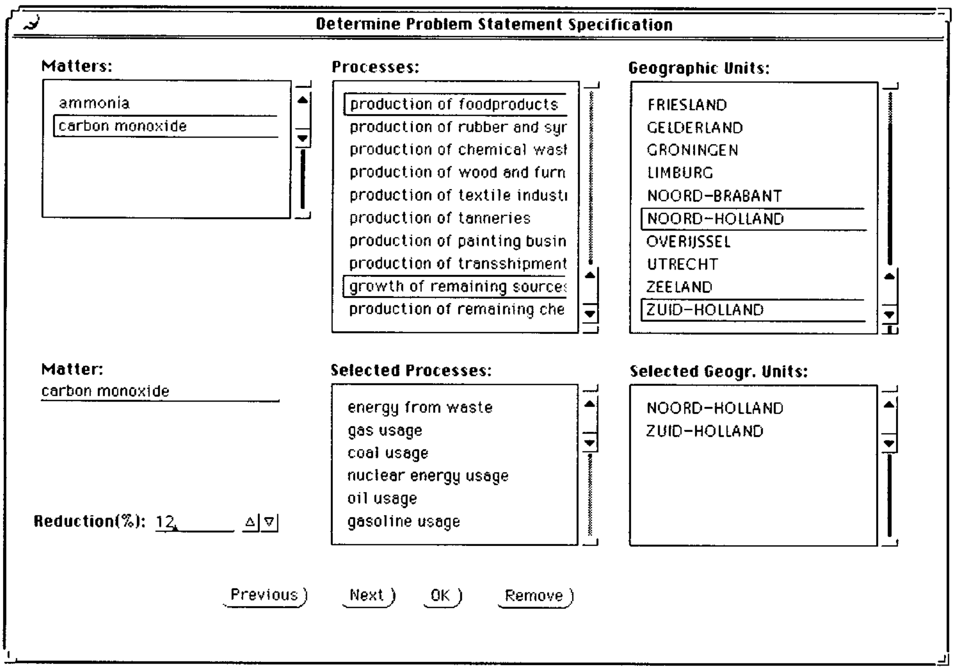

FIGURE 7. Determine problem statement: specification of user goals.

the system determines the emissions implied by the selected set. This is performed by a calculation-intensive system which predicts the effects of measures on future emissions on the basis of models designed for this purpose (the RIM + system). If the set of measures has been rejected the emissions are not determined but the next task, (decide about continuation) described below, is performed. The following tasks are distinguished.

- Determine applicable measures (system).
- Select measure set (system).
- Evaluate selected measure set (user).
- Determine emissions (system).

4.3.5. Design process coordination

The coordination of the overall design strategy is a task in itself. In principle, the system has sufficient knowledge to be able to coordinate the design process according to a global design strategy. However, at some points in the process, the user may wish to influence the strategy. For example, after the emissions for a selected set of measures (in fact for a number of them) have been determined, the results are presented to the user who can indicate whether this is a satisfactory solution or not. The two sub-tasks are as follows.

- Decide about design process continuation (user).
- Determine design process continuation (system).

4.4. LEVELS OF INTERACTION IN THE EXAMPLE SHARED TASK MODEL

In this section, the levels of interaction in the shared task model for environmental decision support are discussed.

4.4.1. Object-level interaction

Within the shared task model, the only exchange of factual information is initiated by the system when the system presents the selected sets of measures. This is one-sided interaction: the user is not given the opportunity to change object-level information on the measures themselves. The user's influence on the combination of measures is limited to the specification of strategic preferences.

4.4.2. Interaction at the level of strategic preferences

Most interaction within the shared task model is directly related to the user's preferences and requirements with respect to the choice of measures. This level of interaction, the level of strategic preferences, is modelled based on the following

- Establishing the initial goals of the design process (see Figure 7).
- Providing strategic preferences on the manner in which requirements should be refined or distributed over the various processes

Specifying strategic preferences in measure set selection (see Figure 8). They are in fact soft requirements; they guide the reasoning strategy for a (preferred, optimal) set of measures.

4.4.3. Interaction at the level of task model modification

This most global level of interaction is related to the global sequencing of tasks: under which conditions which sub-task should be performed. Interaction at this level is for example: after having seen the selected set of measures, the user decides which sub-task is

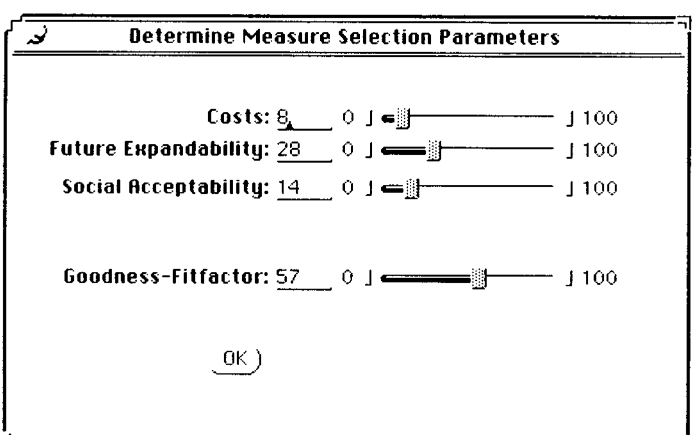

FIGURE 8. Strategic preferences: determine measure selection parameters

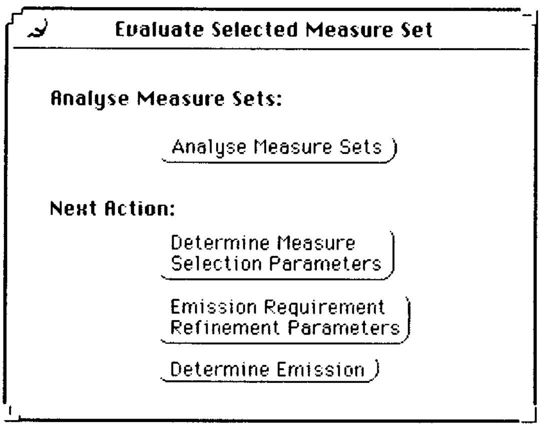

FIGURE 9. Task model modification

to be performed: determine measure selection parameters, determine emission requirement refinement parameters or determine emissions (see Figure 9).

5. Shared task models as a basis for a clarification agent

The agent paradigm provides a useful perspective to model cooperation between a user and a system on a complex task, such as diagnosis (e.g. Brazier & Treur, 1994). By considering both the user and the diagnostic support system as agents, a number of notions from the area of multi-agent systems can be exploited, such as autonomy, pro-activeness and reactiveness of both agents, and communication and cooperation between the agents (Castelfranchi, 1995; Jennings, 1995; Wooldridge & Jennings, 1995*a*, *b*). When human agents work together with automated agents in a cooperative task, clarification is often needed. For example, clarification may be needed not only about the meaning of terms used in the task, but also on the process; e.g. on the overall problem-solving method that is followed in the task, or on a specific sub-task that is performed. Clarification can be considered as a complex task, which is at a meta-level with respect to the (object) task on which clarification is generated. Usually, human agents can clarify many aspects by themselves, but sometimes they may need support from another agent which cooperates with the human agent to perform the clarification task.

This perspective applied to a diagnostic task results in a multi-agent architecture consisting of three agents: the *user*, the *diagnostic support agent* and the *clarification support agent*. Each pair of these agents has interaction: the user and the diagnostic support agent cooperate in the object task, the user and the clarification support agent cooperate in the clarificataion task and the clarification support agent monitors and inspects the diagnostic support agent, the user and their interaction. In the approach introduced here, the user and the diagnostic support agent have a shared model of the diagnostic task, whereas the user and the clarification support agent have a shared model of the clarification task. The shared task model for clarification reflects the shared task model of disgnosis; clarification includes clarification of the overall diagnostic reasoning process. The architecture introduced in this paper has been applied to chemical process diagnosis, in cooperation with Dutch chemical industry, as a continuation of the research on diagnosis of Nylon 6 production processes described in Section 4.3.

In Section 5.1, the overall multi-agent architecture is introduced. In Section 5.2, the (shared) task models for both the diagnostic object task and the clarification task are briefly introduced and related to the three agents. In Section 5.3, the agent models of the diagnostic support agent and the clarification support agent are discussed in greater detail and the interaction pattern between the agents is discussed. Section 5.4 addresses example clarification processes and the knowledge used.

5.1. THE MULTI-AGENT ARCHITECTURE

In this section, a multi-agent architecture is proposed in which three agents are distinguished: the user, the diagnostic support agent and the clarification support agent. As both the user and the diagnostic support agent may needs information from the external world (for example on results of inquiries or observations performed to acquire additional information), interaction with the external world needs to be explicitly modelled. In Figure 10, the top-level composition (i.e. abstracting from the internal structure of the agents) is depicted.

The user, the diagnostic support agent and the clarification support agent are autonomous, and in principle run in parallel. They all reason and react not only on the basis of their own knowledge, but also on the basis of information provided by communication with other agents and observation in the external world. The clarification support agent almost continually interacts with the diagnostic support agent: the diagnostic support agent keeps the clarification support agent informed of the status of the diagnostic process. The clarification support agent can explicitly request additional information

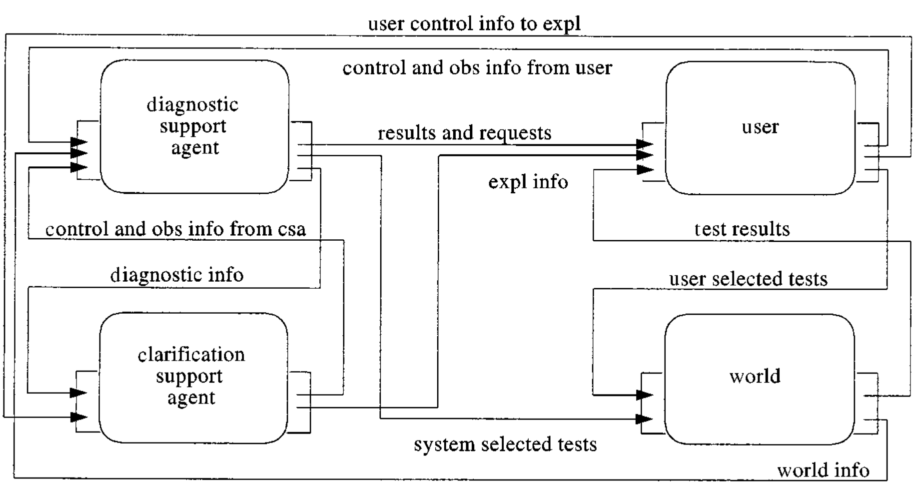

FIGURE 10. Top level composition of the multi-agent system.

from the diagnostic support agent, for example, to analyse a specific line of reasoning. The diagnostic support agent and the clarification support agent communicate information to the user, who in turn may communicate additional information on the basis of which a diagnostic and/or clarification process may continue. The *weak agent notion* (cf. Wooldridge & Jennings, 1995*a*) provides a useful concept to model the interactive concurrent processes involved. On the basis of this agent notion the required knowledge and behaviour is specified at a conceptual level. The weak agency characteristics autonomy, reactiveness, pro-activeness and social abilities all apply in the context of application.

5.2. THE SHARED TASKS MODELS FOR DIAGNOSTIC TASK AND CLARIFICATION TASK

The shared task models for diagnosis and clarification will be discussed in relation to the agents in which they occur: the diagnostic support agent in Section 5.2.1, the clarification support agent in Section 5.2.2 and the user in Section 5.2.3. Each of these agents is modelled according to the notion of weak agency: the weak-agency characteristics are realized by the tasks distinguished within an agent.

5.2.1. The processes within the diagnostic support agent

The diagnostic support agent has diagnosis as its main task. Its task composition is depicted in Figure 11. Similar to Section 3, the diagnostic task has three main tasks: (1) to coordinate and control the diagnostic problem solving process, (2) to determine hypotheses to be considered and (3) to validate these hypotheses. To determine which hypotheses are to be considered, a number of hypotheses are generated and one or more are selected. To validate hypotheses, a number of observations are determined and performed, and the hypotheses evaluated on the basis of the observation results.

5.2.2. The processes within the clarification support agent

The following global categories or levels of clarification can be distinguished: *terminological* clarification, clarification of *syntax* (of expected input) and clarification of the diagnostic *process*. Each of the different levels of clarification needed is modelled as a separate task both within the user and within the clarification support agent.

The actual clarification process is performed by the two agents in cooperation. In addition the clarification support agent has an own process control task. In Figure 12, the clarification support agent's composition is depicted. Note that the task model for process clarification has an object-meta relation with the task model of diagnosis. For each of the sub-tasks of diagnosis a meta-task is modelled that monitors and clarifies this

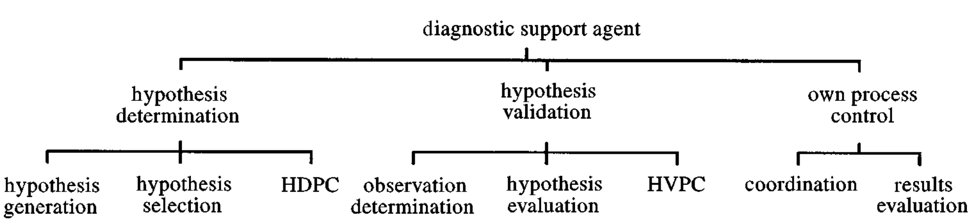

FIGURE 11. Processes at different abstraction levels within the diagnostic support agent.

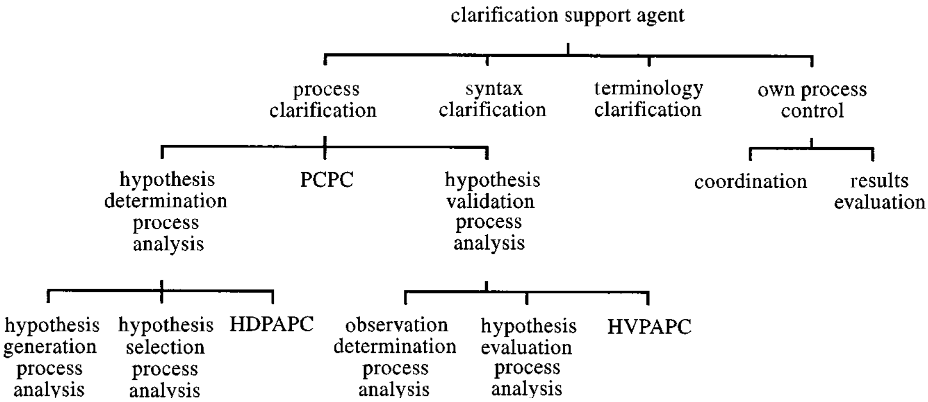

FIGURE 12. Processes at different abstraction levels within the `clarification support agent`.

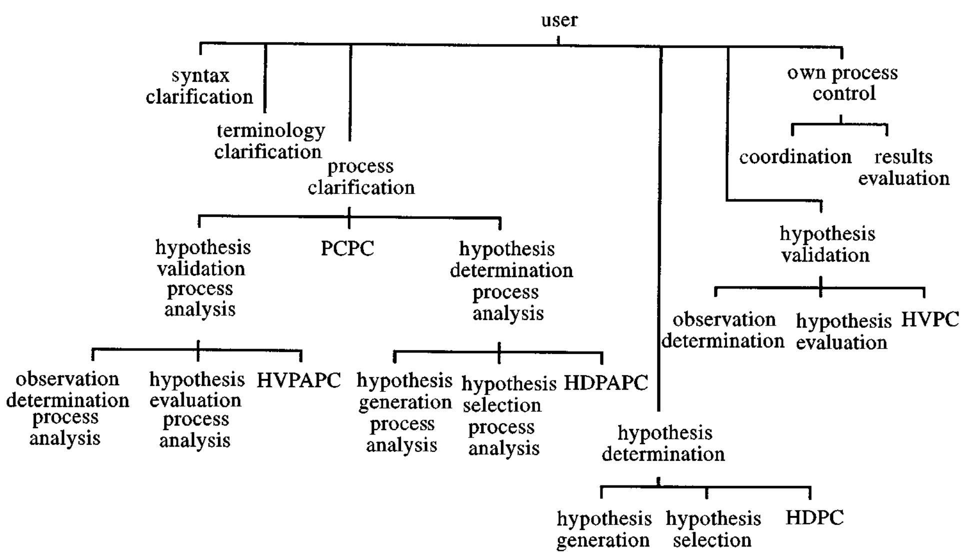

FIGURE 13. Assumed tasks of the `user`.

specific sub-task. This task composition is useful to structure the process of process clarification.

5.2.3. The tasks of the user

The tasks of the user are two-fold: diagnosis and clarification. The former is performed by the user in cooperation with the diagnostic support agent, the latter in cooperation with the clarification support agent. To obtain insight into the agents, their tasks, and their interactions, the assumed tasks of the user can also be modelled. In Figure 13 a tree representation of the assumed user task composition of the diagnostic task is depicted (it is the intersection of Figures 11 and 13). The task model for clarification within the user is shared with the clarification support agent (it is the intersection of Figures 12 and 13). Which of the tasks in a shared task model actually is performed by the one agent and which by the other (and which by both), depends on the task division. In Section 7, an example task division and interaction pattern are discussed.

5.3. MODELS FOR DIAGNOSTIC SUPPORT AGENT AND CLARIFICATION SUPPORT AGENT

In this section, the agent architectures for the diagnostic support agent and for the clarification support agent are discussed.

5.3.1. The diagnostic support agent

In contrast to most of the literature on diagnosis, the model of diagnosis used in this paper is in the first place a *process* model. It models the process of focusing on appropriate hypotheses to be investigated and determines which observations are to be performed for the focus hypotheses. Thus, the decision making in the course of the diagnostic process is modelled, whereas literature such as Reiter (1987) addresses diagnosis from a static point of view, assuming that all relevant observation results are already given.

Following the task composition explained in Section 3.1, the diagnostic support agent is composed of three components (see Figure 14): `own process control`, `hypothesis determination` (which determines the focus hypotheses: those that should be investigated) and `hypothesis validation` (which takes care of investigation whether a focus hypothesis should be confirmed or rejected). The component `own process control` is composed of two sub-components: `process coordination`

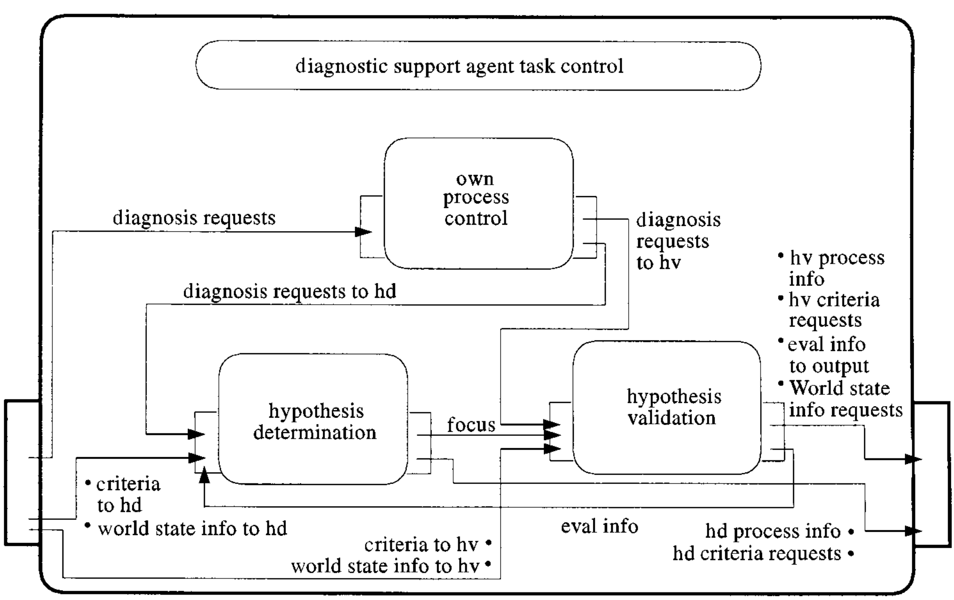

FIGURE 14. Top-level composition of the `diagnostic support agent`.

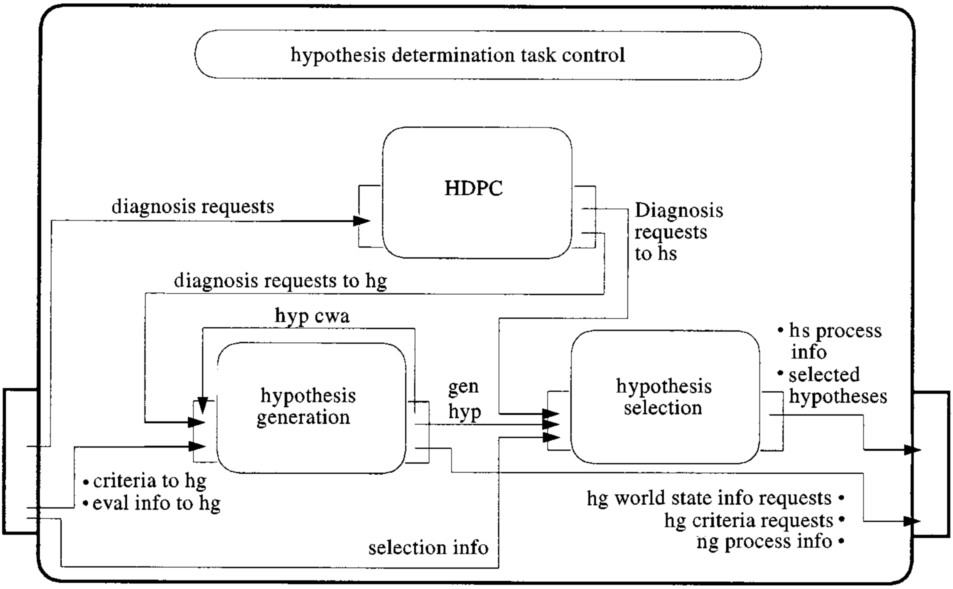

FIGURE 15. Composition of `hypothesis determination`.

and `results evaluation`. The component `hypothesis determination` is composed of the three sub-components `hypothesis determination process control`, `hypothesis generation` (which generates the options for focus hypotheses) and `hypotheses selection` (which selects one of the options); see Figure 15.

The component `hypothesis validation` is also composed of three sub-components (see Figure 16): `hypothesis validation process control`, `observation determination` (which determines the observations that are relevant to the focus hypothesis, and are not yet performed), and `hypothesis evaluation` (which evaluates the focus hypothesis in the light of the results of observations that have been performed). In the component `hypothesis evaluation` any of the static approaches to diagnosis available in the literature (e.g. Reiter, 1987) can be incorporated.

5.3.2. The clarification support agent

Composition at the top level of the clarification support agent is depicted in Figure 17. When required, the clarification support agent receives specific requests for clarification from the user. Furthermore, the clarification support agent receives information from the diagnostic support agent on the basis of which the diagnostic process can be fully (continually) monitored. The clarification support agent's `own process control` analyses an incoming request, and determines which type of clarification is required: terminology clarification, syntax clarification or process clarification. After this, the right component addresses the clarification request. Terminology clarification and syntax clarification are not discussed in further detail in this paper.

Explanation on the process can be generated by the component `process clarification` at any point in the diagnostic process. Often, clarification of the

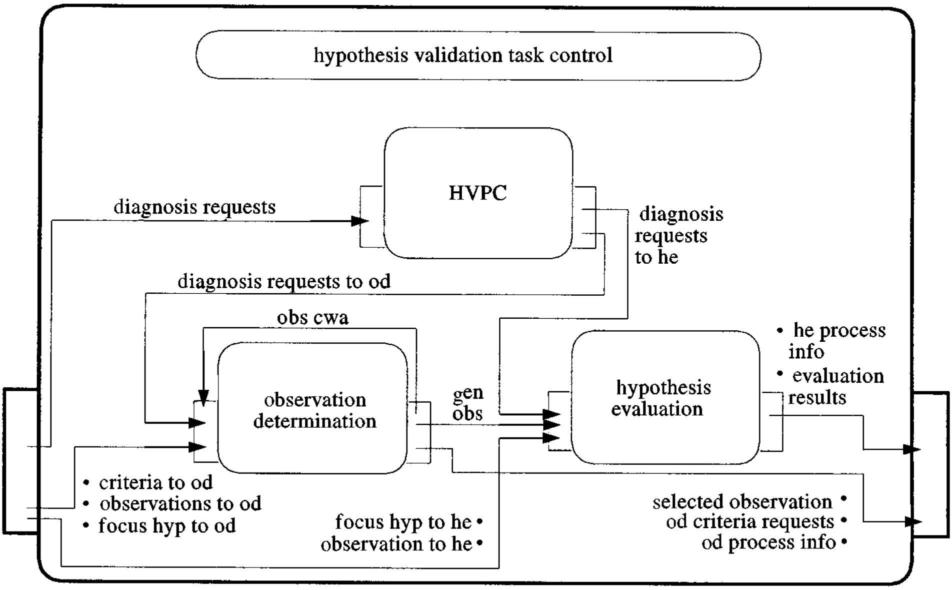

FIGURE 16. Composition of `hypothesis validation`.

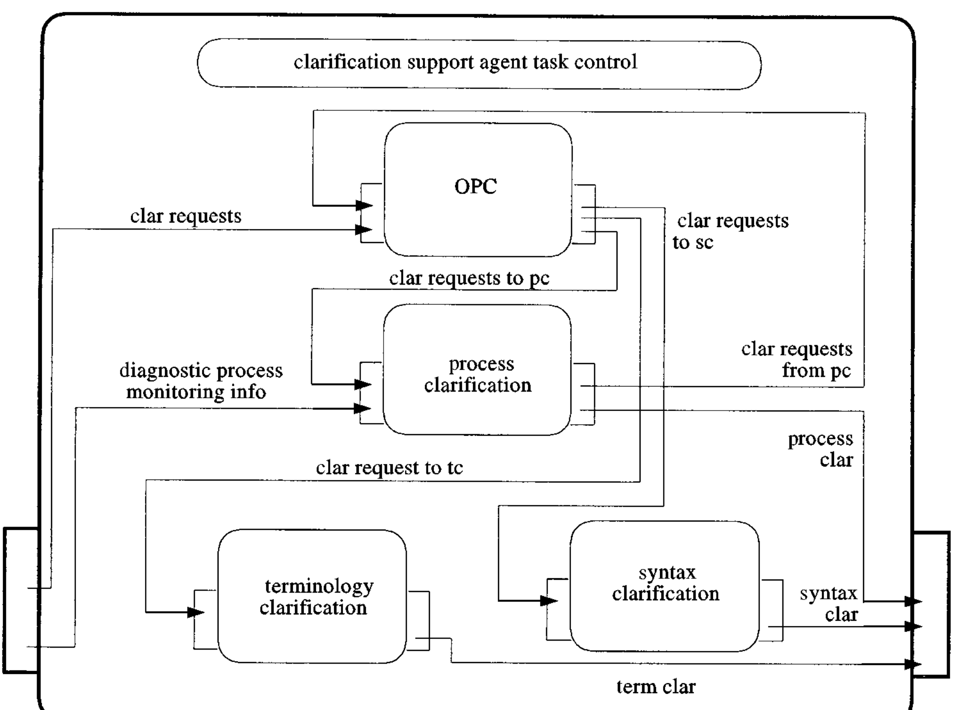

FIGURE 17. Composition of the top level of the `clarification support agent`.

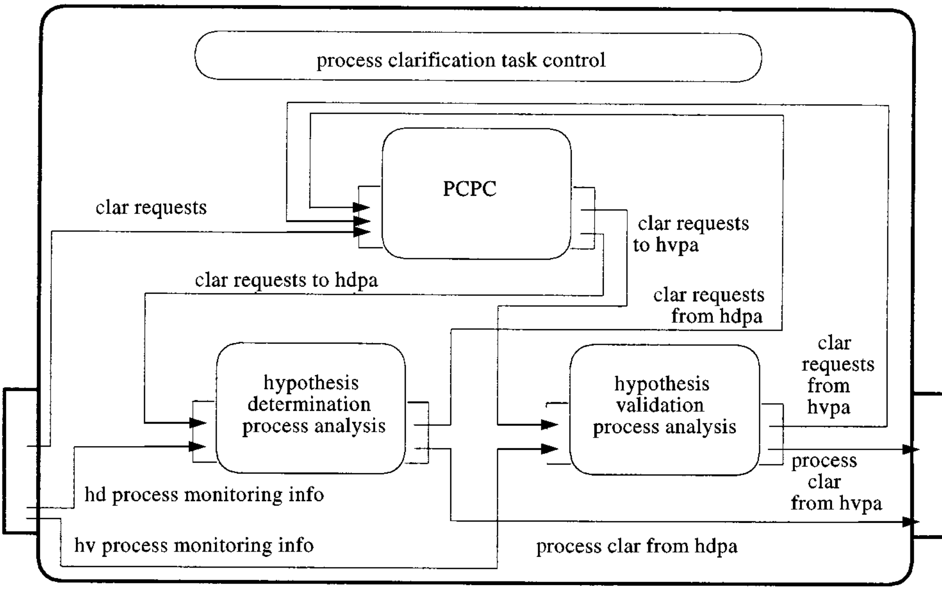

FIGURE 18. Composition of `process clarification`.

diagnostic process involves clarification of previous steps in the process in addition to the current status, and possibly includes the future process as well. A request often refers to the current step in the process, but may also refer (either explicitly or implicitly) to previous (or ubsequent) steps.

In some cases, clarification of the entire diagnostic process is requested. During process clarification a user may request further clarification on the syntax or on terms. The process clarification task coordinates the design of the contents of clarification in interaction with the user.

The architecture of the process clarification component of the clarification support agent mirrors the model of the diagnostic process within the diagnostic support agent. It is composed of three components (see Figure 18): `process clarification process control`, `hypothesis determination process analysis` and `hypothesis validation process analysis`. The component `hypothesis determination process analysis` is composed of three sub-components: `hypothesis determination process analysis process control`, `hypothesis generation process analysis` and `hypothesis selection process analysis` (see Figure 19).

The component `hypothesis validation process analysis` is also composed of three sub-components: `hypothesis validation process analysis process control`, `observation determination process analysis` and `hypothesis evaluation process analysis` (see Figure 20).

The component `process clarification` first classifies the clarification request according to whether it concerns clarification of the overall tasks or clarification of the process within a specific task. This classification is performed by the component

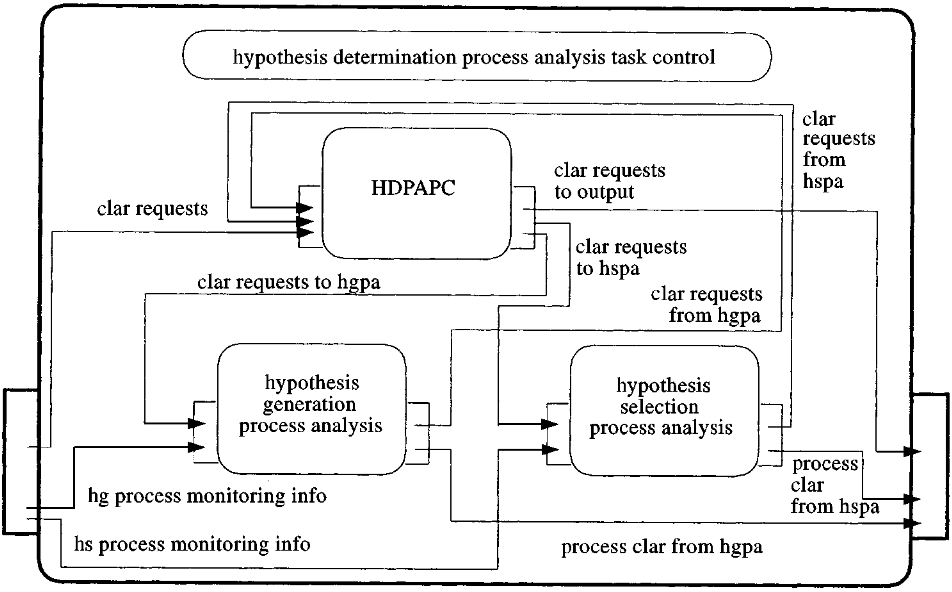

FIGURE 19. Composition of `hypothesis determination process analysis`.

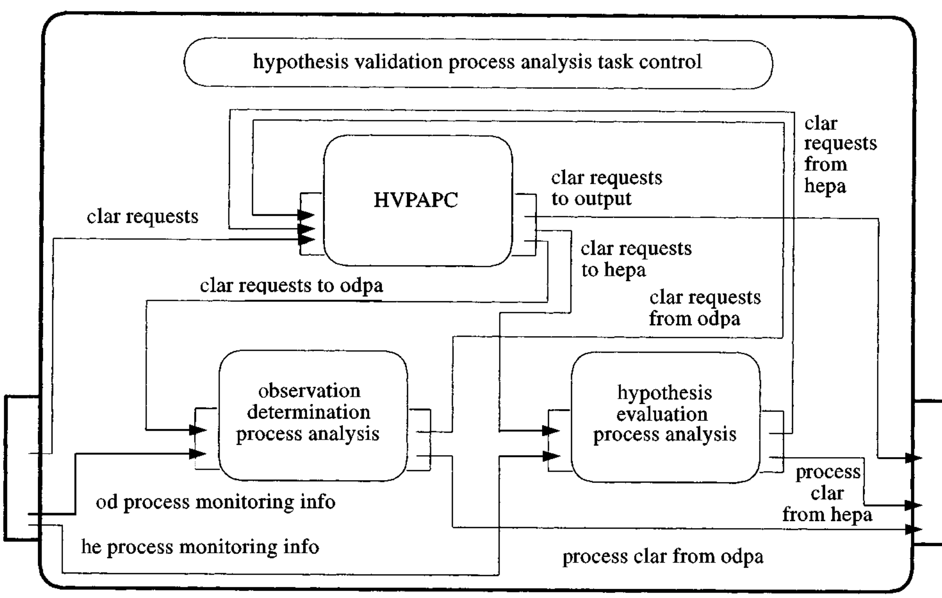

FIGURE 20. Composition of `hypothesis validation process analysis`.

`process clarification process control`. If clarification of the overall task is requested the component `process clarification process control` generates a global explanation. If clarification of the process within a task is required, this is delegated to the specific component within the clarification support agent related to that task. For example, the component `hypothesis generation process analysis` determines clarification of the process within the task hypothesis generation of the diagnostic support agent. The components `hypothesis selection process analysis`, `observation determination process analysis`, and `hypothesis evaluation process analysis` of the clarification support agent play a similar role with respect to the components `hypothesis selection`, `observation determination` and `hypothesis evaluation` of the diagnostic support agent.

Process explanation has been designed to be concise. An explanation, however, can refer to concepts that have been used in a previous task. For example, if a user asks why a specific symptom has to be observed, the explanation refers to the selected hypothesis. The user then may want to know the origin of this selected hypothesis. In this case the user may ask for additional clarification. Additional clarification is provided by the clarification agent. In this approach the clarification process can be interactively and iteratively deepened, depending on the detail desired by the user.

5.3.3. Task division and interaction between the agents

In this section, the task division and interaction pattern between the agents are discussed by means of an example pattern.

An example task division between user and diagnostic support agent. For the application to chemical process diagnosis, on which the example is based, the following task division between user and diagnostic support agent was used:

Hypothesis generation
  user: provides initial symptoms.
  diagnostic support agent: generates hypotheses on the basis of initial symptoms.
Hypothesis selection
  user: provides information on the chemical process configuration.
  diagnostic support agent: determines a first selection (which is presented to the user),
  user: selects one of the hypotheses from the presented list.
Observation determination
  diagnostic support agent: suggests an observation of a symptom to the user.
Hypothesis evaluation
  user: provides the observation result.
  diagnostic support agent: derives conclusions about the focus hypotheses.

An example interaction pattern. The user may perform both clarification tasks and diagnostic tasks and switch between them at any given moment. Both types of tasks are supported: the diagnostic tasks by the diagnostic support agent, and the clarification tasks by the clarification support agent. The following interaction patterns shows how these three agents and their contributions to the process are coordinated. In this example, the point in the process at which the user is expected to provide input on the hypothesis to be selected is discussed. The process proceeds as shown in Figure 21.

5.4. CLARIFICATION PROCESSES AND REQUIRED KNOWLEDGE

In this section, the types of clarification requests that can be handled by the clarification support agent are discussed, and the clarification process is illustrated by showing how some example clarification requests are handled.

Active components and interactions	Explanation
dsa-hypothesis selection	The diagnostic support agent determines a list of hypotheses as first selection.
dsa → user	The diagnostic support agent communicates this list of hypotheses to the user.
user-process clarification	The user reads the list of hypotheses and tries to clarify the current state of the process, determines a lack of clarification, and generates a clarification request.
user → csa	The user communicates the clarification request to the clarification support agent.
csa-process clarification	The clarification support agent analyses the clarification request and generates an explanation.
user ← csa	The clarification support agent communicates the explanation to the user.
user-process clarification	The user reads the explanation and integrates it with his or her own clarifications.
user-hypothesis selection	Clarification is satisfactory, the user selects one of the hypotheses.
dsa ← user	The user communicates the selected hypothesis to the diagnostic support agent.

FIGURE 21. An example interaction pattern among the three agents.

5.4.1. Types of clarification requests that can be handled

The model for diagnostic process clarification can be used to answer, among others, the following process clarification requests (in these example questions the pointers "this" and "these" refer to specific instances that can be indicated by the user).

- *Where are we in the process?*
- *What are we doing?*
- *Why do I need to give initial symptoms here?*
- *Why do I need to give these particular initial symptoms?*
- *Why is this hypothesis in the list of displayed hypotheses?*
- *Why do I need to select a hypothesis?*
- *Why was this hypothesis focussed upon?*
- *Which hypotheses were selectable?*
- *Which selectable hypotheses have not (yet) been selected?*
- *Why is this hypothesis not in the list of displayed hypotheses?*
- *Who do I need to perform a test here?*
- *Why is this test suggested?*
- *Why is this test not suggested?*
- *How was this hypothesis found (why is this hypothesis correct)?*
- *Why was this hypothesis rejected?*
- *Which hypotheses were rejected?*
- *Which hypotheses that were selected were neither rejected nor confirmed?*
- *Why was this selectable hypothesis not confirmed?*
- *Which hypotheses were not considered?*
- *Why was this hypothesis not considered?*
- *Why was this selected hypothesis not confirmed nor rejected?*

For some clarification requests, it is shown how the clarification can be generated and which knowledge structures are used. The generated explanation instances (texts, but in some cases also graphics) are of sort `Explanation_texts`. Some of the instances are generic; they are specified as objects of the sort `Explanation_texts`: the names of the texts. In the implementation the actual explanation instance to which the name refers is displayed. Most explanation instances consist of a generic template text in which reference is made to one or more specific symptoms or hypotheses. These texts are parameterized and are specified by functions with parameters as arguments.

5.4.2. Knowledge required

To be able to provide clarification, the clarification support agent requires at least the following types of meta-information and meta-knowledge.

1. *Status information about the actual and planned diagnostic process.* Information on the diagnostic process refers to information on the diagnostic process as performed by the user and the diagnostic support agent:
   - About the previous process until the time point at which the request for clarification is put forward.
   - About the planned diagnostic process.

   This information is *dynamically* acquired by the clarification support agent during the process by monitoring the user and the diagnostic support agent during their diagnostic reasoning and interaction. For example, the generated list of hypotheses is recorded, together with information on which of them have been chosen previously, and which were confirmed or rejected.
2. *Knowledge about relations between the concepts involved in the diagnostic reasoning process.* To be able to generate explanation about the possible and actual reasoning steps in the diagnostic process the following types of meta-knowledge are required.
   - Relations between initial symptom information and generated hypotheses.
   - Relations between hypotheses and the observations required to confirm or reject them.

   This *static* knowledge is part of the specification of the clarification support agent.

5.4.3. Example clarification processes

In this section an example clarification is shown to illustrate the clarification process and the knowledge used within the clarification support agent.

At any moment in the diagnostic process, it is assumed that the user can also perform his or her clarification task, and if the clarification generated by the user is not sufficient, for example one of the following two questions can be generated.

- *Where are we in the process?*
- *What are we doing?*

These questions are input for the clarification support agent in the form

```
required_process_clar(where_are_we)
required_process_clar(what_are_we_doing)
```

For the first question the clarification support agent's component `process clarification process control` uses the knowledge

```
if required_process_clar(where_are_we)
and current_task(T : Tasks)
then selected_process_explanation(task_text(T : Tasks), where_
are_we)
```

It derives

```
selected_process_clar(task_text(t), where_are_we)
```

`where t` is the current task in the process. The text referred to by `task_text(t)` is given in the form of a graphical representation of the overall diagnostic task model, with colour or blinking to indicate which task is currently being performed, and a short description of the aim of the task (see below for instances of these task descriptions).

From the second question also the first question is derived within `process_clarification_process_control`, using the knowledge

```
if required_process_clar(what_are_we_doing)
then required_process_clar(where_are_we)
```

In addition, by the rule

```
if required_process_clar(what_are_we_doing)
then selected_process_explanation(general_task_model_text,
what_are_we_doing)
```

it is derived

```
selected_process_clar(general_task_model_text, what_are_we_
doing)
```

The explanation text referred to by `general_task_model_text` is given in the form of a graphical representation of the overall diagnostic task model, with colour or blinking to indicate which task is currently being performed, and, in addition, from each of the displayed components a hyperlink to a short text to explain the task.

Another clarification request can arise during hypothesis selection, at the instant the user receives the list of hypotheses selected by the diagnostic support agent.

- *Why is this hypothesis* `H : Hyps` *in the list of generated hypotheses*?

This question is handled by the component `hypothesis generation process analysis`; it is represented as

```
required_process_clar(why_generated(H : Hyps))
```

First, by the component `process clarification process control` the question is classified as one to be treated by the component `hypothesis generation process analysis`: Using the knowledge

```
if required_process_clar(why_generated(H : Hyps))
then required_process_clar_type(hypothesis_generation)
```

the component process clarification process control generates

```
required_process_clar_type(hypothesis_generation)
```

Note that this is a case in which the question goes back to an earlier task in the process. Next, `hypothesis generation process analysis` processes the question using the knowledge

```
if required_process_clar(why_generated(H : Hyps))
  and is_positive_support_for(S : Symptoms, H : Hyps)
  and has_been_observed(S : Symptoms)
then selected_process_explanation(hyp_generation_
text(H : Hyps, S : Symptoms), why_generated(H : Hyps))
```

For a number of symptoms, conclusions that are drawn contain the expression

```
hyp_generation_text(H : Hyps, S : Symptoms)
```

Based on these conclusions the following text is displayed.

Because of the initial symptoms `S : Symptoms` *that were observed, and the fact that hypothesis* `H : Hyps` *can cause these symptoms, the hypothesis was one of the generated possible hypotheses.*

6. Discussion

In this paper, an approach to user-centred design is presented and illustrated on the basis of the development of four applications of knowledge-intensive decision support systems. To model a task in which an expert user and an intelligent decision support system collaborate, appropriate intermediate representations of the task at hand must be designed. The different roles of a shared task model as an intermediate representation of the task (within which different levels of specificity are modelled), have been addressed in Sections 3–5.

Shared task models and descriptions of systems have been developed using the compositional development method DESIRE (Brazier *et al*., 1997, 1998). One of the advantages of DESIRE is its suitability for development of both single- and multi-agent systems. The declarative nature of specification in DESIRE was of particular importance to modelling strategic preference interaction between the user and the decision support system. Explicit, declarative representation of strategic knowledge (for which modelling primitives exist within DESIRE) allows strategic knowledge itself to be the subject of interaction, both from the user to the system (which preferences hold, which relations between preferences exist, etc., influencing the system's reasoning strategy), and from the system to the user (which preferences have been fulfilled, to which extent, etc.).

Within DESIRE, existing abstract models of generic tasks provide a means to structure initial interaction with the expert user during the acquisition of a shared task model. A number of agreed, shared task models have developed within applications (decision support systems) in different domains.

In the first application, decision support in the domain of soil sanitation, one of the existing generic task models for diagnostic reasoning based on anti-causal knowledge provided a means to structure *knowledge acquisition*. The shared task model developed for this domain was, in the end, a specialization of this existing generic task model. In the second domain discussed in the paper, diagnosis of chemical processes, two existing generic task models for diagnostic reasoning were introduced: the first one based on causal knowledge, and later in the acquisition process a model based on anti-causal knowledge. In contrast to the knowledge engineers' expectations, the model based on causal knowledge was not in line with the expert's diagnostic approach. The anti-causal model, however, was useful: the acquisition process resulted in a shared task model for diagnostic reasoning of Nylon-6 production as a specialization of this model.

In the third application a shared task model was based on an existing generic model of design. The third domain of application is environmental decision making. Three levels of *user interaction* were considered to be important by the expert users: interaction at the object level, at the level of strategic preferences and at the level of task modification. This led to the design of a user interface based on the acquired shared task model. Within this user interface all the three levels of interaction were available to expert users.

The fourth application addresses the use of a shared task model within a multi-agent system. Two software agents were designed such that each shared a task model with the user: one for a diagnosis task, and one for a clarification task. The multi-agent architecture presented has been developed to support a user both at the level of the diagnostic task he or she is performing and at the level of *clarification*. In cooperation with Dutch chemical industry (DSM), the architecture has been applied to diagnosis of chemical processes. The objective is to make it easier for a user to work with a diagnostic support system by adding the clarification support agent, and thus to contribute to more successful application of diagnostic support systems in chemical industry. The genericity of the architecture makes it reusable for various tasks and domains, for diagnosis and other tasks, within chemical industry and beyond.

To design multi-agent systems, the compositional development method DESIRE has turned out quite useful. The approach of compositional modelling from a multi-agent perspective was chosen to make explicit the different interactions of the parties involved, and to support dynamic, event-driven behaviour. The cooperating agents can behave in a reactive manner, but can also be pro-active by taking the initiative. In the perspective put forward in Dieng, Giboin, Tourtier and Corby (1992), among others, it was argued that in the interactive processes involving knowledge-based support systems, explanation is crucial and that agent-based modelling could turn out to be useful for that. In Liebermann (1997), interface agents are described that support search for information. A difference with our approach is that no explicit task model is involved and that no explanation is given on the process on the basis of such a task model. The same type of difference can be found with the approach described in Barrett, Maglio and Kellem (1997). Another difference with the references as mentioned is that in our approach the architecture is based on an explicit compositional architecture, which supports maintainabiliy and reuse.

In comparison, the CommonKADS model set (see Hoog, Martil, Wielinga, Taylor, Bright & van de Velde, 1994) includes an organization model, a task model, an agent model, an expertise model, a communication model and a design model.

An *organization model* analyses the impact of a system in and on an organization. A *task model* describes the tasks related to the realization of a function in an organization independent of the agent responsible for the performance of the tasks. A task model, however, when complete, relates each task to an agent. Agents are described in an *agent model*. The capabilities of an agent are described in an *expertise model*. Strategic knowledge is defined by interface and task aspects of the problem-solving knowledge included in an expertise model. Communication tasks, defined in a *communication model*, as specified in terms of user models (defined in an agent model) and transfer tasks (defined in an expertise model). Important decisions made during the design of an application together from the *decision model*.

Not all of the three levels of interaction (object-level interaction, interaction at the level of strategic preferences and interaction at the level of task composition) distinguished in Section 4 are easily distinguished within Common-KADS. Object-level interaction is defined by transfer tasks. How interaction at the level of strategic preferences or task model modification can be modelled is less clear. One option is to use the task layer of the expertise model, another is to use the REFLECT principle (see van Harmelen, *et al.*, 1992). Using the task layer to model these levels of interaction may not be appropriate, as domain-specific (strategic) knowledge is involved, which then would not be specified at the domain layer and inference layer of the expertise model. This is, therefore, not a very elegant solution. The REFLECT approach models an entire expertise-model in the domain layer of another expertise model. Explicit strategic reasoning can be modelled within this approach, but entails the (recursive) combination of two expertise models for this purpose.

Reasoning about states of different reasoning processes is quite common in, for example, multi-agent situations. The CommonKADS framework does not include constructs or models which can be used for this purpose. The semantics of DESIRE, however, based on temporal logic (states and transition between states) has been designed to model interaction between components (which may be tasks) by explicit specification of transition between states.

From the above it follows that the way in which the three levels of interaction are incorporated into one knowledge-based system is not as transparent in CommonKADS as in DESIRE. In DESIRE, the levels of abstraction and temporal semantics facilitate the modelling of these levels of interaction.

The role of shared task models in situations in which more than two parties (agents) are involved, is a current focus of further research. A shared task model is an agreed model: in some situations agreement may be reached among more than two parties (resulting in a situation comparable to the situation described above for two parties), but in other situations different models of a task may exist between parties, thus requiring "attunement" between parties. Such collaborative tasks are currently being analysed.

This research has been (partially) supported 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 NWO-SION within project 612-322-316, "Evolutionary design in knowledge-based systems", DSM and RIVM provided funding for parts of the reported research as well. The authors have had stimulating discussions with Rob Faessen (DSM Research), Bart Gras (DSM Research/Bolesian) and Gert Poppe (DSM Research). Arnold Wentholt and Bart Kusse (RIVM) contributed to the application discussed in Sections 4.3 and 4.4. Jos Boelens contributed to the application to soil sanitation.

References

BARNARD, P. J. (1993). Modelling users, systems and design spaces. In *Proceedings of HCl International '93*, pp. 331–336. Amsterdam, Elsevier.

BOELENS J. (1991). *Soil sanitation and strategic interaction.* Masters thesis, Department of Mathematics and Computer Science, Vrije Universiteit Amsterdam.

BARRETT, R. MAGLIO, P. P. & KELLEM, D. C. (1997). How to personalize the Web. In S. Pemberton Ed. *Human Factors in Computing Systems, Proceedings CHI'97. New York: ACM,* Reading, MA: Addison Wesley, pp. 75–82.

BRAZIER, F. M. T., DUNIN-KEPLICZ, B. M., JENNINGS, N. R. & TREUR, J. (1995, 1997). Formal specification of multi-agent systems: a real world case. In V. LESSER, Ed. *Proceedings 1st International Conference on Multi-Agent Systems ICMAS'95*, 1995, pp. 25–32. Cambridge, MP: MIT Press. Extended version in M. HUHNS & M. SINGH, Eds. (1997). *International Journal of Co-operative Information Systems IJCIS*, **6,** 67–94 (Special issue on Formal Methods in Co-operative Information Systems: Multi-Agent Systems).

BRAZIER, F. M. T., JONKER, C. M. & TREUR, J. (1998). Principles of Compositional Multi-agent System Development. In J. CUENA, Ed. *Proceedings of the 15th IFIP World Computer Congress, WCC'98, Conference on Information Technology and Knowledge System, IT&KNOWS'98, pp. 347–360.*

BRAZIER, F. M. T. LANGEN, P. H. G. VAN & TREUR, J. (1998). Strategic knowledge in design: a compositional approach. In *Knowledge-based Systems*, **11** (Special Issue on Strategic Knowledge and Concept Formation, K. HORI, Ed.), 405–416.

BRAZIER, F. M. T. & RUTTKAY, Zs. (1993). A compositional, knowledge-based architecture for intelligent query user interfaces. In S. ASHLUND, K. MULLET, A. HENDERSON, E. HOLLNAGEL & T. WHITE, Eds. *Adjunct Proceedings of the INTERCHI '93* (INTERACT'93 + CHI'93), pp. 145–146.

BRAZIER, F. M. T. & TREUR, J. (1994). User-centered knowledge-based system design: a formal modelling approach. In L. STEELS, G. SCHREIBER, & W. VAN DE VELDE, Eds. *A Future for Knowledge Acquisition, Proceedings of the 8th European Knowledge Acquisition Workshop, EKAW'94,* Lecture Notes Artificial Intelligence, Vol. 861, pp. 283–302. Berlin: Springer-Verlag.

BRAZIER, F. M. T., TREUR, J. & WIJNGAARDS, N. J. E. (1996*a*). The acquisition of a shared task model. In N. SHADBOLT, K. O'HARA & G. SCHREIBER, Eds. *Advances in Knowledge Acquisition, Proceedings of the 9th European Knowledge Acquisition Workshop. EKAW'96, Lecture Notes* Artificial Intelligence, Vol. 1076, pp. 278–289. Berlin: Springer Verlag.

BRAZIER, F. M. T., TREUR, J. &. WINJGAADS, N. J. E. (1996*b*) Modelling interaction with experts: the role of a shared task model. In W. WAHLSTER, Ed. *Proceedings of the 12th European Conference on Artificial Intelligence, ECAI'96,* pp. 241–245. New York: John Wiley & Sons.

BRAZIER, F. M. T., TREUR, J., WIJNGAARDS, N. J. E. & WILLEMS, M. (1999). Temporal semantics of compositional task models and problem solving methods. *Data and Knowledge Engineering*, **29,** 17–42.

BROWN, D. C. & CHANDRASEKARAN, B. (1989). *Design Problem Solving: Knowledge Structures and Control Strategies*, Research Notes in Artificial Intelligence, London: Pitman.

CASTELFRANCHI, C. (1995). Commitments: From individual intentions to groups and organizations. In V. LESSER, Ed. *Proceedings of the 1st International Conference on Multi-Agent Systems, ICMAS'95*, pp. 41–48. Cambridge, MA; MIT Press.

CHANDRASEKARAN, B. (1986). Generic tasks in knowledge-based reasoning: high-level building blocks for expert system design. *IEEE Expert*, **1,** 23–30.

CHANDRASEKARAN, B. (1990). Design problem solving: a task analysis. *AI Magazine*, **11,** 59–71.

DIENG, R., GIBOIN, A., TOURTIER, P. A. & CORBY, O. (1992). Knowledge acquisition for explainable, multi-expert, knowledge-based design systems. In Th. WETTER, K. D. ALTHOFF, J. BOOSE, B. R. GAINES, M. LINSTER & F. SCHMALHOFER, Eds. *Current Developments in Knowledge Acquisition, Proceedings of the 6th European Knowledge Acquisition Workshop, EKAW'92.* Lecture Notes in Artificial Intelligence, Vol. 599, pp. 298–317. Berlin: Springer Verlag.

FORD, K. M., J. M. BRADSHAW, J. R. ADAMS-WEBBER & N. M. AGNEW (1993). Knowledge acquisition as a constructive modelling activity. In K. M. FORD & J. M. BRADSHAW, Eds. *Knowledge Acquisition as Modeling*, *International Journal of Intelligent Systems*, **8,** 9–32.

HARMELEN, F. VAN, B. WIELINGA, B. BREDEWEG, G. SCHREIBER, W. KARBACH, M. REINDERS, A. VOß, H. AKKERMANS, B. BARTSCH-SPÖRL & E. VINKHUYZEN (1992). Knowledge-Level Reflection. In B. LE PAPE & L. STEELS, Eds. *Enhancing the Knowledge Engineering Process — Contributions from ESPRIT*, pp. 175–204. Amsterdam, The Netherlands.

HOOG, R. de. R. MARTIL, B. WIELINGA, R. TAYLOR, C. BRIGHT & W. VAN DE VELDE (1994). *The CommonKADS model set*. Deliverable DM1.1x of ESPRIT Project P5248 "KASA-II".

JENNINGS, N. R. (1995) Controlling co-operative problem solving in industrial multi-agent systems using joint intentions. *Artificial Intelligence Journal*, **74**.

JOHNSON, H. & JOHNSON, J. (1991). Task knowledge structures: Psychological basis and integration into system design. *Acta Psychologica*, **78,** 3–26.

JOHNSON, H. & JOHNSON, P. (1993). Explanation facilities and interactive systems. In W. GRAY, W. HEFLEY & D. MURRAY, Eds. *Proceedings of the International Workshop on Intelligent User Interfaces*. New York: ACM.

JONKER, C. M. & TREUR, J. (1998). A generic multi-agent architecture for interactive diagnostic tasks with clarification. In J. CUENA, Ed. *Proceedings of the 15th IFIP World Computer Congress, WCC'98 Conference on Information Technology and Knowledge Systems, IT&KNOWS'98* pp. 361–374.

LIEBERMAN, H. (1997) Autonmous interface agents. In S. PEMBERTON, Ed. *Human Factors in Computing Systems, Proceedings CHI*'97, pp. 67–74. New York: ACM, Reading, MA: Addison Wesley.

REITER, A. (1987). A theory of diagnosis from first principles. *Artificial Intelligence*, **32,** 57–95.

RAO, A. S. & GEORGEFF, M. P. (1991). Modeling rational agents within a BDI architecture. In R. FIKES & E. SANDEWALL, Eds. *Proceedings of the 2nd Conference on Knowledge Representation and Reasoning*. pp. 473–484. Los Altos, CA: Morgan Kaufman.

TREUR, J. (1993). Heuristic reasoning and relative incompleteness. *International Journal of Approximate Reasoning*, **8,** 51–87.

WOOLDRIDGE, M. & JENNINGS, N. R. (1995*a*). Agent theories, architectures, and languages: a survey. In: M. WOOLDRIDGE & JENNINGS, Eds. *Intelligent Agents. Proceedings of the 1st International Workshop on Agent Theories, Architectures and Languages, ATAL'94,* Lecture Notes in Artificial Intelligence, Vol. 890. Berlin: Springer Verlag, pp. 1–39.

WOOLDRIDGE, M. & JENNINGS, N. R. (Eds.) (1995b). *Intelligent Agents. Proceedings of the 1st International Workshop on Agent Theories, Architectures and Languages, ATAL'94,* Lecture Notes in Artificial Intelligence, Vol. 890. Berlin: Springer Verlag.