The Role of Abilities of Agents in Re-design

Frances Brazier, Catholijn Jonker, Jan Treur, Niek Wijngaards[*]

February 27th, 1998
Artificial Intelligence Group
Department of Mathematics and Computer Science
Vrije Universiteit Amsterdam
de Boelelaan 1081a
1081 HV Amsterdam
The Netherlands
Email: `{frances, jonker, treur, niek}@cs.vu.nl`
URL: `http://www.cs.vu.nl/~wai`

Abstract. Abilities of agents and properties of their environment provide a means to describe behaviour and functionality. These abilities also provide a basis for re-design. In this paper an example is given of a prototype system for re-design of a multi-agent system in which the abilities and properties are made explicit.

Contents:

- 1. INTRODUCTION
- 2. GENERIC ABILITIES AND PROPERTIES
  - 2.1 Generic abilities of an agent
  - 2.2 Properties of a material world
  - 2.3 Properties of a multi agent system
- 3. THE ROLE OF ABILITIES IN RE-DESIGN
  - 3.1 Generic model of design
  - 3.2 Refinement of the generic model of design
- 4. THE ROLE OF ABILITIES IN A SPECIFIC RE-DESIGN PROCESS
- 5. DISCUSSION AND CONCLUSIONS
- 6. ACKNOWLEDGEMENTS
- 7. REFERENCES

1. INTRODUCTION

During the past five years extensive research has been conducted in the field of multi-agent systems. Many different notions of agency have been proposed (e.g. Nwana, 1996; Wooldridge and Jennings, 1995; Shoham, 1993). One notion of agents in which weak agency is distinguished from strong agency has been proposed by Wooldridge and Jennings (1995): weak agency is characterised by autonomy, social ability, reactiveness, and pro-activeness. In contrast the notion of strong agency is based on the characteristics of mentalistic and intentional notions (related to the stronger notion of intentional stance (Dennett, 1987).

The characteristics of weak agency defined by Wooldridge and Jennings (1995) provide a means to reflect on the tasks an agent needs to be able to perform. Pro-activeness and autonomy are related to an agent's ability to reason about its own processes, goals and plans. Reactivity and social ability are related to the ability to be able to interact with the material world and to communicate with other agents. The ability to communicate and co-operate with other agents and to interact with the material world often relies on an agent's ability to acquire and maintain its own knowledge of the world and other agents.

These abilities are essential to agent behaviour: they define an agent's behaviour. Knowledge of an agent's behaviour is needed to design *generic models* of agents, to *re-design* multi-agent systems, and to *verify and validate* multi-agent systems.
In the knowledge engineering community one of the areas of research is the characterisation of tasks and problem solving methods, e.g. the identification of *capabilities* of problem solving methods, e.g. see (Benjamins, 1993; Fensel, 1997; Fensel, Motta, Decker and Zdrahal, 1997; Harmelen and Teije, 1997; Aitken and Kingston, 1997; Orsvärn, 1996; Wielinga, Schreiber and Breuker, 1992; Breuker, 1997). Another related characterisation of problem solving methods is based on assumptions, e.g. see (Benjamins, Fensel and Straatman, 1996; Fensel and Straatman, 1997) .

The distinction between *capabilities* and *abilities* is that a capability of a model is a possible static characteristic (e.g. knowing how a car can be repaired); while an ability is a characteristic of dynamic behaviour (e.g. actually repairing a car). As dynamic behaviour is of paramount importance to multi-agent systems, abilities may also need to refer to distributed control.

The compositional development method DESIRE for multi-agent systems includes a modeling framework and software environment for the development of multi-agent systems (Brazier, Dunin-Keplicz, Jennings and Treur, 1995; 1997; Brazier, Treur, Wijngaards and Willems, 1996). Within this method generic agent models, based on desired agent abilities, are used to structure knowledge acquisition and agent design. Note that within this approach a distinction is made between agents and the material world. The material world is not considered to be an agent. As a result a distinction is made between *abilities* of agents, and *properties* of the material world or the multi-agent system as a whole.

A number of generic abilites of agents and properties of multi-agent systems are discussed in Section 2. To illustrate the role of abilities for the (automated) re-design of an existing multi-agent system, first a model for re-design is presented in Section 3. Section 4 describes the abilities involved in an example re-design process and the relationships between these abilities. A trace of the re-design process (as produced by a prototype re-design system) is shown to illustrate the role of abilities in re-design. A discussion of the results is presented in Section 5, together with suggestions for future research.

2. GENERIC ABILITIES AND PROPERTIES

A multi-agent system (MAS) is composed of interacting agents and the material world. Abilities and properties can be assigned to

- individual agents,
- the material world,
- an individual agent in relation to the agents and the world with which it interacts,
- the world in relation to the agents with which it interacts, and
- a multi-agent system as a whole.

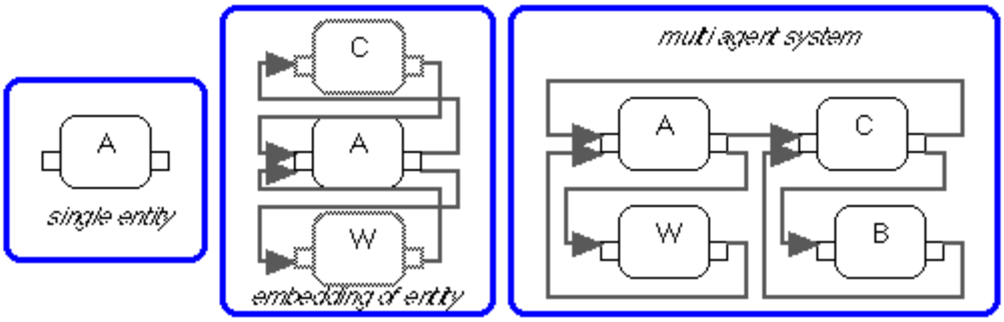

Figure 1. Aggregation levels within a multi-agent system

In fact, three different levels of aggregation can be distinguished, as shown in Figure 1: *single entity, embedded entity* and *multi-agent system*. The abilities assigned to each level are characterised as follows:

single entity	An `agent' or `material world' pur sang: what it can do, can't do, abilities or properties. Formulated in terms of an agent or material world.
embedded entity	Abilities and properties of a single agent in relation to other agents and/or the material world with which it is connected. Abilities and properties of the material world in relation to the agents with which it is connected. This includes abilities relating to communication or interaction between agents and/or the material world. Examples of the embedding relationship for Figure 1 include: embedding_of(A) ={ A, C, W }, embedding_of(B) = { B, C }, and embedding_of(W) = { W, A }.
multi-agent system	Properties in terms of the entire MAS, formulated in terms of the MAS, or `sets of agents', or `sets of agents and the material world'.

Note that the aggregration levels do not have an `inclusion' relationship, i.e. abilities or properties are not automatically `inherited'. However, abilities and properties at one level, may be related to abilities and properties at another level.

In this section first the abilities and properties of single entities are described: the generic abilities of a single agent are discussed in Section 2.1, and the properties of the material world are discussed in Section 2.2. The properties of a multi-agent system are discussed in Section 2.3.

2.1 Generic abilities of an agent

Individual agents often have different abilities. A number of abilities with which an agent's behaviour may be described are the following

- co-operation,
- bi-directional communication,
- agent own process control, and
- world interaction.

Often abilities are directly dependent on other abilities. The *dependency* relationship needs to be recognised. These abilities can also, however, be refined, both with respect to their *specialisations* (refinement of the ability into more specific abilities) and with respect to their *realisation* (refinement of the ability into more fine-grained abilities related to reasoning about the ability, and more fine-grained abilities abilities related to the effectuation of the ability).

The ability of co-operation. The ability of co-operation depends on two other abilities: the ability of an agent to control its own processes and the ability to communicate with other agents. In Figure 2, the co-operation ability is described in terms of the dependency between these abilities: the dependency on the abilities of bi-directional communication and agent own process control.

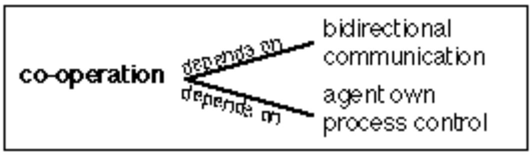

Figure 2. Dependency relationships for ability co-operation

Note that the specific abilities on which a given ability depends, is context sesnsitive: which specific abilities can be chosen depends on their availability in a given situation.

Figure 3 shows the refinement relationships for the ability of co-operation. The more specific abilities related to co-operation are the ability to *plan* co-operation and the ability to *monitor* co-operation. The abilities related to the realisation of the ability of co-operation are the ability to *reason* about co-operation and the ability to *execute* co-operation.

These abilities are further refined. The abilities related to the realisation of the ability to plan co-operation are the ability to reason about planning co-operation, and the ability to execute planning co-operation. The abilities related to the realisation of the ability to monitor co-operation are, likewise, the ability to reason about monitoring co-operation, and the ability to execute monitoring co-operation. These realisation related abilities are, in fact, specialisations of the abilities related to the realisation of the ability of co-operation. The ability to reason about planning co-operation and the ability to reason about monitoring co-operation are refinements of the ability to reason about co-operation. The ability to execute planning co-operation and the ability to execute monitoring co-operation are refinements of the ability to execute co-operation.

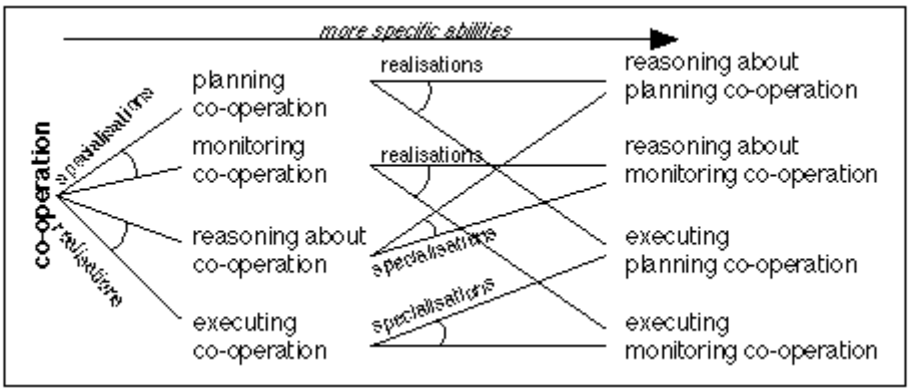

Figure 3. Refinements of the ability of co-operation.

The abilities of agent own process control and bi-directional communication influence planning and monitoring of co-operation: the agent's own decisions, and information on and from other agents, is used to manage co-operation.

The ability of bi-directional communication. The ability of bi-directional communication depends on the ability of an agent to control its own processes, as depicted in Figure 4.

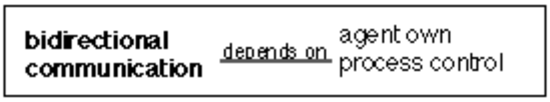

Figure 4. Dependency relationship for bi-directional communication

Figure 5 shows the refinement relationships for the ability of bi-directional communication. The more specific abilities related to bi-directional communication are the ability to communicate to others (unidirectional communication to others) and to receive communication from others (unidirectional communication from others). The abilities related to the realisation of the ability of bi-directional communication are the ability to *reason* about bi-directional communication, and the ability to *execute* bi-directional communication.

These more specific abilities are further refined, and related to the ability to reason about unidirectional communication from others, the ability to reason about unidirectional communication to others, the ability to execute unidirectional communication from others, and the ability to execute unidirectional communication to others.

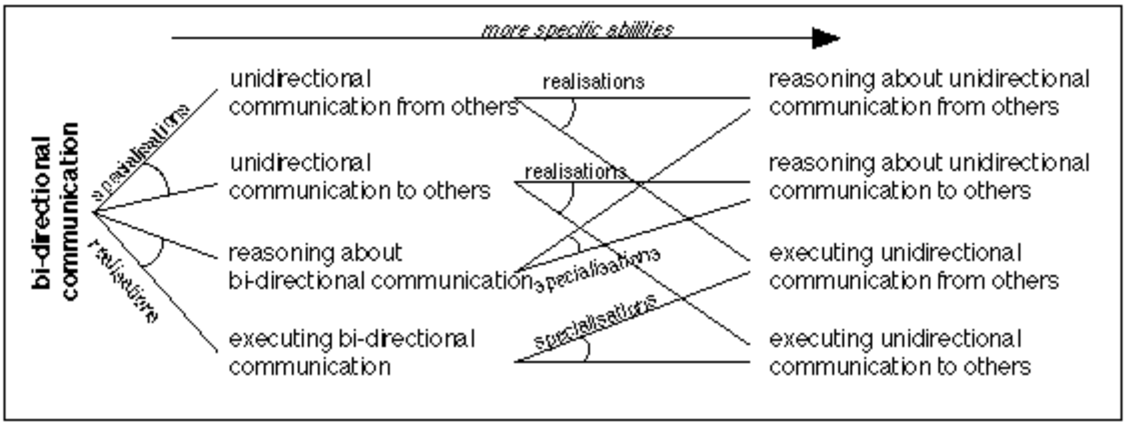

Figure 5. Refinements of the ability of bi-directional communication.

The ability of agent own process control. The ability of an agent to control its own processes can be refined as shown in Figure 6.

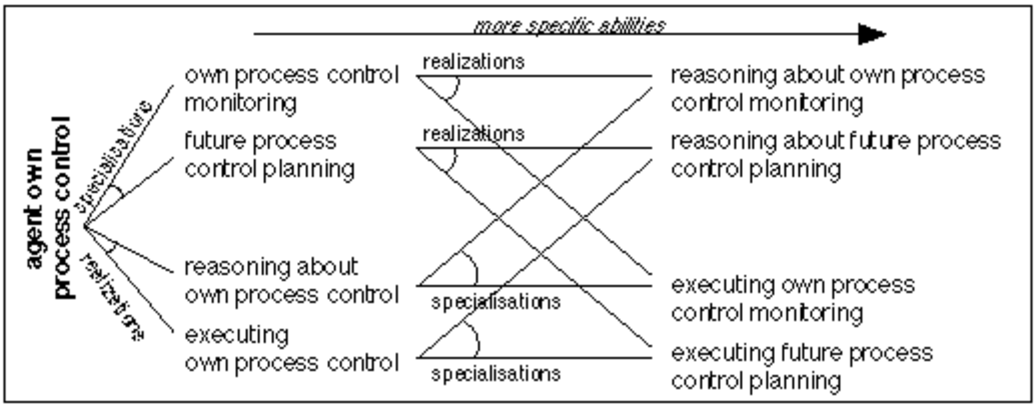

Figure 6. Refinements of the ability of agent own process control.

The two more specific abilities related to the ability of an agent's ability to control its own processes, are

- the ability to *monitor* its own processes, and

- *plan* its own future processes.

These abilities are each further refined to abilities in which realisation (reasoning and execution) is explicitly defined for each of these abilities.
The abilities related to the realisation of the ability of an agent to control its own processes are the ability of an agent to *reason* about its own processes, and the ability of an agent to *execute* its own processes. For each of these abilities two more specific abilities are depicted. Reasoning about an agent's own process control is related to the abilities to reason about current and future process control. Executing an agent's own process control is related to an agent's abilities to monitor current processes and to plan future processes.

The ability of world interaction. The ability to interact with the material world can be refined to more specific abilities: the ability to observe in the world and the ability to initiate actions in the world, as shown in Figure 7. The ability of observation is further refined into the ability of passive observation and the ability of active observation. The ability of active observation is refined into two abilities: the ability of observation initiation and the ability of processing observation results. The ability of passive observation is refined into the ability of processing observation results. These abilities are each further refined to abilities in which realisation (reasoning and execution) is explicitly defined for each of these abilities.

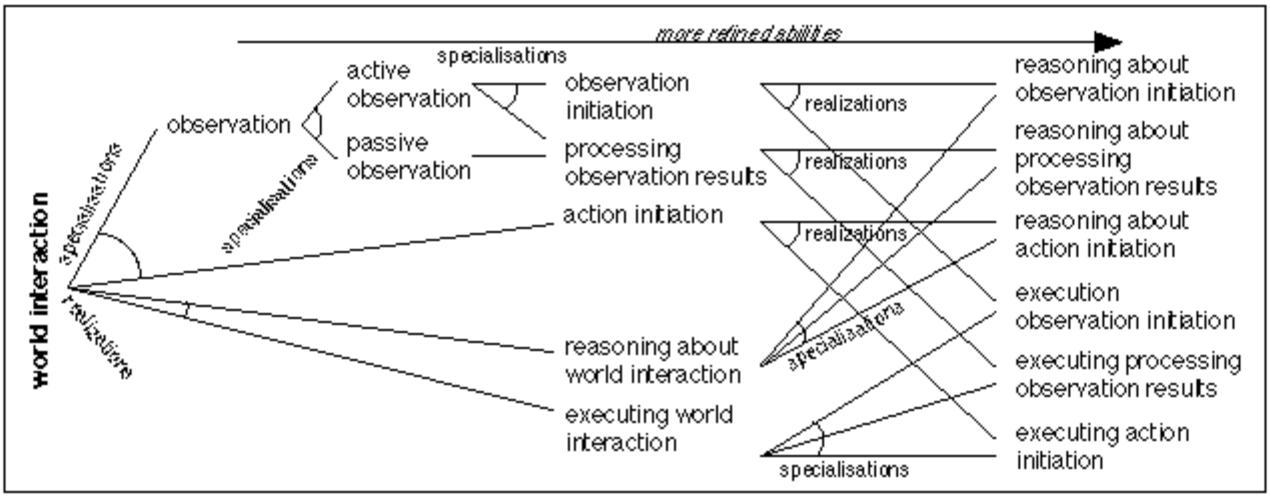

Figure 7. Refinements of the ability of world interaction.

The abilities related to the realisation of the ability of an agent to interact with the world are the ability of an agent to *reason* about its interaction with the world, and the ability of an agent to *execute* interaction with the world. For each of these abilities three more specific abilities are depicted. Reasoning about world interaction is related to the abilities to reason about observation initation, to reason about processing observation results, and to reason about action initiation. Executing interaction with the material world is related to an agent's abilities to execute observation initiation, to execute processing observation results, and to execute action initiation.

2.2 Properties of a material world

The material world, part of a multi-agent system, has properties, and not abilities. These properties are related to how "equipped" the world is to handle interaction with agents. The property or world interaction is described below.

The property of world interaction. *Properties* of material world are depicted in Figure 8. The property of world interaction (from the point of view of the material world) can be realized by two properties: the property of processing observations and the property of handling actions. These properties are the counterpart of the *ability* of world interaction: the *ability* of active observation and the *ability* of passive observation.

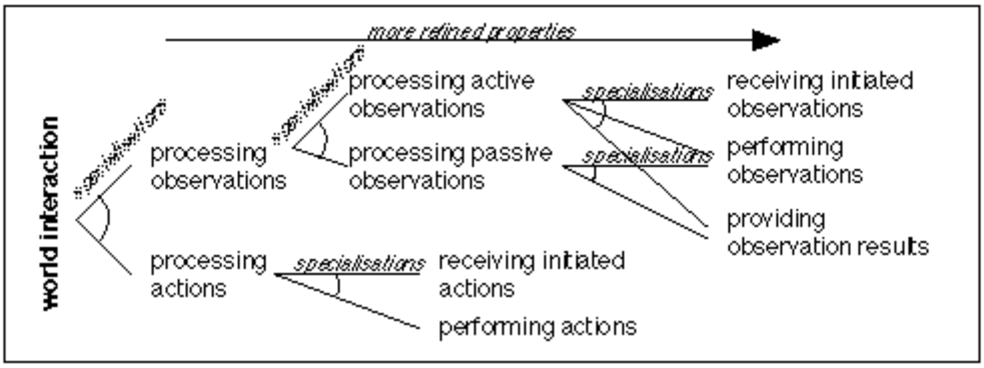

Figure 8. Refinements of the property of world interaction.

The property of processing observations can be refined into the more specific properties:

- processing active observations (i.e. counterpart for the ability active observation),
- processing passive observations (i.e. counterpart for the ability passive observation.

The property of processing actions can be refined into the properties:

- receiving initiated actions,
- performing actions.

The property of processing active observations can be refined into the property of receiving initiated observations, the property of performing observations, and the property of providing observation results. The property of processing passive observations can be refined in to the property of performing observations and the property of providing observation results.

As all of these refinements are about the *specialisation* of a property, there is no refinement for reasoning and execution as these refinements are only applicable to agents.

2.3 Properties of a multi agent system

The properties of a multi-agent system, although related to the abilities of individual agents and the properties of a material world, are properties described at the level of the multi-agent system itself. A property of a multi-agent system is, for example, the property of distributed problem solving of two agents (possibly interacting with the material world). Properties of a multi-agent system relate to the roles agents fulfil in relation to each other.

As a more detailed example, consider the multi-agent system in Figure 1. This multi-agent system *could* have the property to distributedly solve problems: for example agent D solves a subproblem for agent A: agent D provides answers to queries from agent A by making observations in the material world W. Agent D can be considered to be an information gatherer for agent A. The initial multi-agent system does not have the property yet, as there is no agent D. Section 4 extends this example by illustrating how the multi-agent system can be re-designed into another multi-agent system which does have this property.

The specific property of a multi-agent system, as described above, can be related to specific abilities and properties at the aggregation level of `embedded entity':

- the ability of agent A to have bidirectional communication with agent D,
- the ability of agent D to have bidirectional communication with agent A,
- the ability of agent D to have actively observe material world W,
- the property of material world W to handle active observation from agent D.

These abilities can be further specialised or realised by means of the descriptions given in the previous sections.

3. THE ROLE OF ABILITIES IN RE-DESIGN

A specific re-design system is used in Section 4 to illustrate the role abilities play in re-design. Before discussing the specific example, however, the re-design system is described. As re-design is an integral part of any design process (an intermediate description is often modified/re-designed to meet given requirements), the re-design process is viewed as a design process, for which a generic model of design exists.

This model of design is briefly described in Section 3.1. An application of this generic model for the re-design of a multi-agent system is described in Section 3.2.

3.1 Generic model of design

A *generic model of design*, in which reasoning about requirements and their qualifications, reasoning about design object descriptions and reasoning about the design process are distinguished, has been proposed by Brazier, Langen, Ruttkay and Treur (1994). This model is based on a logical analysis of design processes (Brazier, Langen and Treur, 1996) and on analyses of existing applications, including environmental waste modelling (Langen, Brazier, Diepenmaat and Pulles, 1995), elevator configuration (Brazier, Langen, Treur, Wijngaards and Willems, 1996) and design of environmental measures (Brazier, Treur and Wijngaards, 1996*b*). The model provides an abstract description of a design process comparable to a *design model* (Coyne, Rosenman, Radford, Balachandran and Gero, 1990). The logical analysis of design processes is comparable to a *design theory* (Smithers, 1994, for parametric design Wielinga, Akkermans and Schreiber, 1995). The model also provides 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 instantiation, involves the specification of knowledge about applicable requirements and their qualifications, about the design object, and about design strategies.

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 on appropriate default assumptions (see also (Smith and Boulanger,1994), or the result of interaction with an outside party (e.g., a client or a designer).
Figure 9 shows the first level of composition of the generic model for design. Three processes are shown at the top level, together with the information exchange.

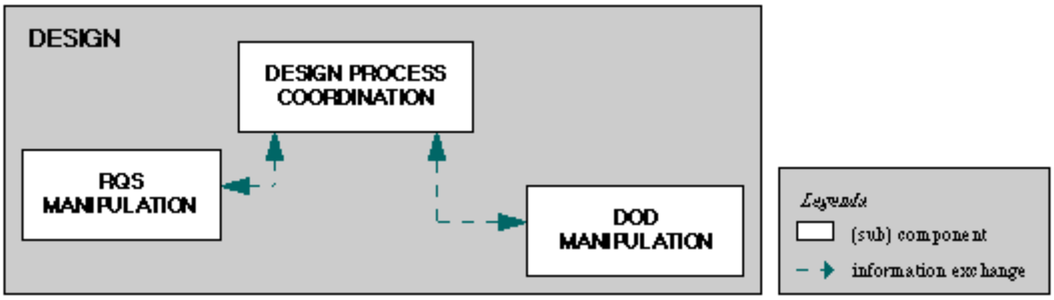

Figure 9. First level of process composition of generic model of design.

The four processes (see Figure 10) related to the process *requirement qualification set manipulation* are:

- RQS modification: the current requirement qualification set is analysed, proposals for modification are generated, compared and the most promising (according to some measure) selected,
- deductive RQS refinement: the current requirement qualification set is deductively refined by means of the theory of requirement qualification sets,
- current RQS maintenance: the current requirement qualification set is stored and maintained,
- RQSM history maintenance: the history of requirement qualification sets modification is stored and maintained.

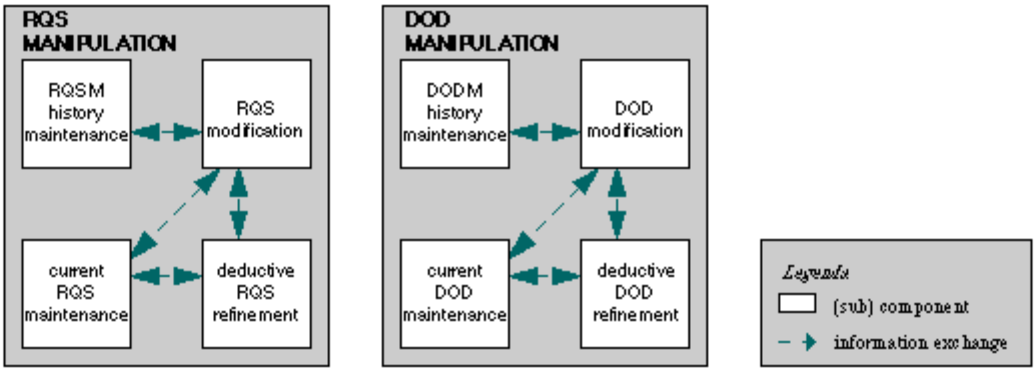

Figure 10. Process composition of manipulation processes of generic model of design.

The four processes (see Figure 10) related to the process of *manipulation of design object descriptions* are:

- DOD modification: the current design object description is analysed in relation to the current requirement set, proposals for modification are generated, compared and the most promising (according to some measure) selected,

- deductive DOD refinement: the current design object description is deductively refined by means of the theory of design object descriptions,
- current DOD maintenance: the current design object description is stored and maintained,
- DODM history maintenance: the history of design object descriptions modification is stored and maintained.

The overall co-ordination of the design process together with the local co-ordination 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 (1994). The notion of design as search is also presented in (Motta and Zdrahal, 1996) in the context of parametric design.

3.2 Refinement of the generic model of design

In this section a specific refinement of the process RQS modification is explained in more detail.

The process RQS modification determines modifications to a requirement qualification set (RQS). To this purpose a number of sub-processes are performed as shown in Figure 11. The sub-process RQS modification process co-ordination is responsible for the co-ordination of the entire process within RQSM: this process determines which RQS, when and by which means a particular RQS is to be modified.

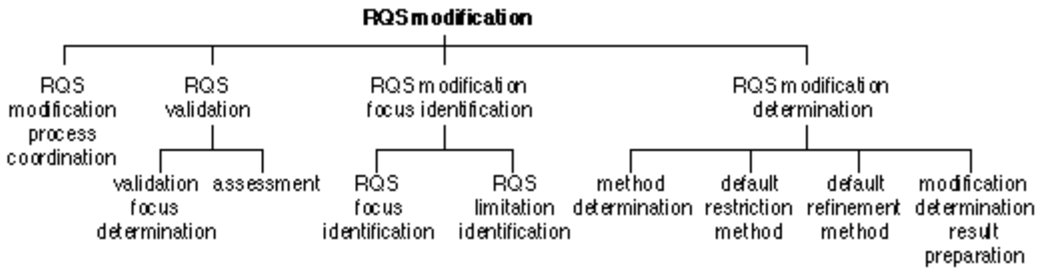

Figure 11. Partial process refinement for RQS modification.

The process RQS validation validates a certain RQS. Its sub-process validation focus determination determines which properties of (qualified) requirements need to be validated (e.g. apparent conflicts, aggregation level per (qualified) requirement, etc.). RQS refinement (see Figure 9) is given these results as goals to pursue. The sub-process assessment assesses the achievement of the validation focus on the basis of the results of the refinement.

The process RQS modification focus identification determines which qualified requirement and which requirement need to be modified, according to the given internal strategy. Two sub-processes are employed: RQS focus identification identifies one or more qualified requirements and/or requirements and RQS limitation identification identifies which constraints are related to the current focus.

The process RQS modification determination provides the actual modifications to the current RQS on the basis of given strategies from RQS modification process co-ordination, and a focus and limitations from RQS modification focus identification. This process entails four sub-processes: method determination chooses the best method corresponding to the given strategy. The sub-process default restriction method removes certain qualified requirements and requirements on the basis of the current modification focus. The sub-process default refinement method selects an appropriate refined (qualified) requirement on the basis of the current modification focus and limitations. The subprocess modification determination result preparation guarantees the correctness of the results, e.g. requirement uniqueness, before formulating the final modifications to the current RQS.

The knowledge involved within RQS modification traverses the refinements graphs of the abilities discussed in Section 2, from the `left to the right': that is, whenever a particular ability can be refined by alternative sets of more fine-grained abilities, a choice is made and one refinement is selected.

4. THE ROLE OF ABILITIES IN A SPECIFIC RE-DESIGN PROCESS

In this chapter the re-design of an example multi-agent system is used to illustrate the role of abilities in the re-design process. Figure 12 depicts the re-design process: before the re-design process, the original multi-agent system lacks a certain ability (cf. Section 2.3 and Figure 1). The new multi-agent system has that particular ability.

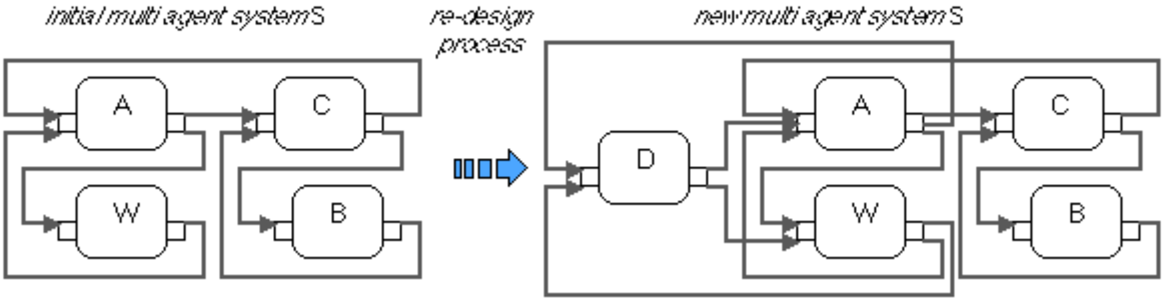

Figure 12. Re-design of a multi-agent system: `before' and `after' the re-design process.

This example focusses on the role of abilities, and not on the role of qualifications of requirements; all requirements are considered to be equally important. The required abilities on the multi agent system are shown in Table 1. The left column presents labels for these requirements.

r_m1	agent_solves_subproblem_for(agent_D, information_gatherer, agent_A)
r_m2	agent_task_explication(agent_D, information_gatherer, searching(internet, scientific_publications))

Table 1. Initially required abilities of multi-agent system mas_S.

The first initial requirement (r_m1) states that agent D is an information gatherer that receives queries from agent_A and provides answers to A and actively observes in material world W. The second initial requirement (r_m2) states that the task of agent D is to search on the internet, with as subject scientific publications.

This example is used to illustrate how initial abilities are modified during a re-design process. Table 2 shows how the initial set of required abilities is restricted to a smaller set (RQS_1): requirements that only refer to domain knowledge structures are excluded (to be later included when the task related requirements have all been fulfilled). An initial focus on the requirement r_m2 results in its deletion from the initial set, resulting in a new set containing only requirement r_m1.

Table 3 shows that the set RQS_1 is first analysed and requirements are found at the `multi-agent system' level of aggregation. The focus is on the requirement r_m1 (the only requirement) and for this requirement more refined abilities and properties are inferred: abilities and properties at the `embedded entity' level of aggregation. The resulting set of requirements is labelled RQS_2.

Table 4 briefly depicts how the set RQS_2 is modified by adding a refinement to the ability of bi-directional communication between agents A and D.

Task	**Information changes**
RQS modification process co-ordination	Continue with the initial set of requirements. This initial set should be validated.
RQS validation: validation focus determination	Results include: whether each requirement is restricted to domain ontology or not.
deductive RQS refinement	Results include: requirement r_m1 does *not* only use domain knowledge structures, r_m2 only uses domain knowledge structures.
RQS validation: assessment	No apparent conflicts or apparent inconcistencies are detected.
RQS modification process coordination	The following strategy is formulated: RQS_modification_strategy(restrict_to, non_knowledge_structures_requirements);
RQS modification focus identification	requirement_selected_as_focus(r_m2); No limitations
RQS modification determination	*modification method determination* selected_RQS_modification_subtask(default_restriction);
default restriction method	is_selected_design_requirement_for_deletion(is_requirement(r_m2, ...), pos);
Resulting RQS_1	is requirement(r_m1, has_ability(mas_S, agent_solves_subproblem_for(agent_D, information_gatherer, agent_A)));

Table 2. Manipulating the initially required abilities: restriction to a subset.

Task	**Information changes**
RQS modification process coordination	Continue with the last set of requirements: RQS_1. This set should be validated.
RQS validation: validation focus determination	Results include: whether each requirement is restricted to domain ontology or not, determine level of aggregation for each requirement, determine whether a requirement is already refined or not.
deductive RQS refinement	Results include: requirement r_m1 is *not* restricted to domain knowledge structures and requirement r_m1 is at the aggregation level of multi_agent_system, r_m1 is not

	refined yet.
RQS validation: assessment	No apparent conflicts or apparent inconcistencies are detected.
RQS modification process coordination	There is only one requirement, at the aggregation level of multi_agent_system, thus: RQS_modification_strategy(refine, non_refined_non_is_a_refinement_qr); RQS_modification_strategy(refinement_extent_includes, structure_type_identification);
RQS modification focus identification	requirement_selected_as_focus(r_m1); No limitations
RQS modification determination	*modification method determination* selected_RQS_modification_subtask(default_refinement);

default refinement method	is_selected_design_requirement_for_addition(is_requirement(r_e1, has_ability(agent_A, bi-directional_communication_with(agent_D))), pos);
	is_selected_design_requirement_for_addition(is_requirement(r_e2, has_ability(agent_D, bi-directional_communication_with(agent_A))), pos);
	is_selected_design_requirement_for_addition(is_requirement(r_e3, has_ability(agent_D, active_observation_in(world_W))), pos);
	is_selected_design_requirement_for_addition(is_requirement(r_e4, has_property(world_W, processing_active_observations_by(agent_D))), pos);

Resulting RQS_2	is requirement(r_m1, has_ability(mas_S, agent_solves_subproblem_for(
	agent_D, information_gatherer, agent_A)));
	is_requirement(r_e1, has_ability(agent_A, bi-directional_communication_with(agent_D)));
	is_requirement(r_e2, has_ability(agent_D, bi-directional_communication_with(agent_A)));
	is_requirement(r_e3, has_ability(agent_D, active_observation_in(world_W)));
	is_requirement(r_e4, has_property(world_W, processing_active_observation_by(agent_D)));
	is_refined_by(r_m1, { r_e1, r_e2, r_e3, r_e4 });

Table 3. Manipulating the RQS_1: determining a refinement for r_m1.

Task	**Information changes**
RQS modification focus identification	requirement_selected_as_focus(r_e1); No limitations

RQS modification determination	*modification method determination* selected_RQS_modification_subtask(default_refinement);

default refinement method	is_selected_design_requirement_for_addition(
	is_requirement(r_s1, has_ability(agent_A,
	unidirectional_communication_from(agent_D))), pos);
	is_selected_design_requirement_for_addition(
	is_requirement(r_s2, has_ability(agent_A,
	unidirectional_communication_to(agent_D))), pos);

Resulting RQS_3	is requirement(r_m1, has_ability(mas_S,
	agent_solves_subproblem_for(
	agent_D, information_gatherer, agent_A)));
	is_requirement(r_e1, has_ability(agent_A, bi-
	directional_communication_with(agent_D)));
	is_requirement(r_e2, has_ability(agent_D, bi-
	directional_communication_with(agent_A)));
	is_requirement(r_e3, has_ability(agent_D,
	active_observation_in(world_W)));
	is_requirement(r_e4, has_property(world_W,
	processing_active_observation_by(agent_D)));
	is_refined_by(r_m1, { r_e1, r_e2, r_e3, r_e4 });
	is_requirement(r_s1, has_ability(agent_A,
	unidirectional_communication_from(agent_D)));
	is_requirement(r_s2, has_ability(agent_A,
	unidirectional_communication_to(agent_D)));
	is_refined_by(r_e1, { r_s1, r_s2 });

Table 4. Manipulating the RQS_2: determining a refinement for r_m2.

The final refinements for the requirement r_m1 are shown in Figure 13:

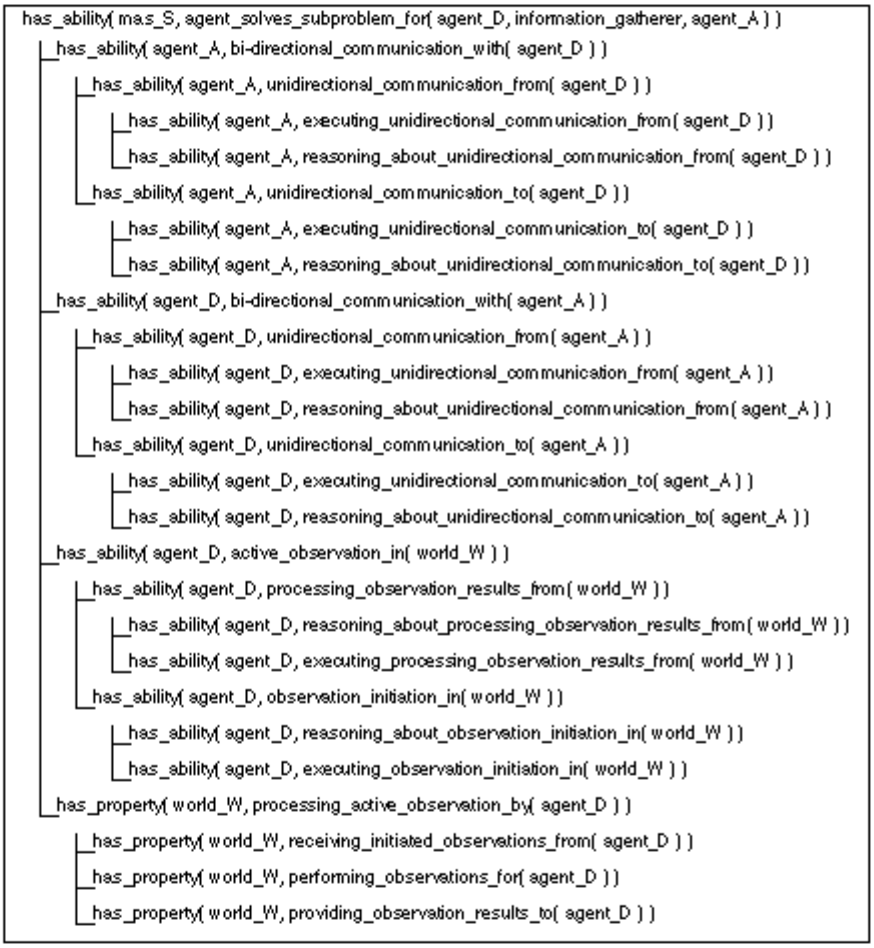

Figure 13. Result of determining refinements for r_m1.

5. DISCUSSION AND CONCLUSIONS

In this paper several generic abilities of agents, and properties of multi-agent systems and the material world have been introduced. On the basis of aggregation levels of multi agent systems the following abilities and properties are introduced:

- ability of co-operation,
- ability of bi-directional communication,
- ability of agent own process control,
- ability of world interaction, and
- property of world interaction with agents (a property of the material world),
- property of distributed problem solving of several agents (a property of a multi-agent system).

Knowledge on these abilities and properties has been provided: dependencies among abilities and properties, and refinements (both specialisations and realisations) of abilities and properties have been outlined.

To illustrate the role these abilities and properties can play, an example re-design process has been discussed. An existing multi-agent system lacking a particular property is re-designed into a different multi-agent system supporting that particular property.

The prototype re-design system and multi-agent systems used as examples in this paper have been modelled and specified in the DESIRE modelling framework. Within this modelling framework dynamic and static aspects of multi-agent systems are made explicit thereby facilitating the identification and realization of abilities and properties. However, the abilities and properties are defined without any *reference to* the DESIRE framework and can be used within different modelling frameworks.

Other areas where these abilities and properties can play a significant role are: reuse of parts of a multi-agent system, verification and validation of (part of) multi-agent systems and design rationale of multi-agent systems.

Future research can focus on a number of issues. The role of these abilities with respect to brokering of parts of agents via the WWW could be investigated. Initial thoughts on brokering parts of a problem solving method are introduced by Fensel (1997*b*). Verification and validation is another line of research closely related (Cornelissen, Jonker and Treur, 1997; Jonker and Treur, 1997). Additional abilities of agents and multi-agent systems have to be described, e.g. abilities concerning beliefs, desires and intentions, co-operation patterns or protocols, et cetera. The current prototype re-design system should be augmented to include knowledge on these additional abilities as well.

6. ACKNOWLEDGEMENTS

The authors wish to thank Pieter van Langen for his contributions to the generic model of design, and Lourens van der Meij and Frank Cornelissen for their support of the DESIRE modeling and specification tools. This research has been (partially) supported by NWO-SION within project 612-322-316: "Evolutionary design in knowledge-based systems" (REVISE).

7. REFERENCES

AITKEN, S. and KINGSTON, J. (1997). Capability Descriptions of Problem-Solving Methods. *Working Paper*, 10 pages.

BENJAMINS, V.R. (1993). *Problem Solving Methods for Diagnosis*. PhD Thesis, University of Amsterdam, Amsterdam.

BENJAMINS, R., FENSEL, D., and STRAATMAN, R. (1996). Assumptions of Problem-Solving Methods and their Role in Knowledge
Engineering. In: WAHLSTER, W. (ed.). *Proceedings European Conference on AI (ECAI '96)*, Wiley and Sons, Chichester.

BRAZIER, F.M.T., DUNIN-KEPLICZ, B.M., JENNINGS, N.R. and TREUR, J. (1995) Formal Specification of Multi-Agent Systems: a Real World Case, In: LESSER, V. (Ed.), *Proceedings First International Conference on Multi-Agent Systems, ICMAS'95*, MIT Press, pp. 25-32.

BRAZIER, F.M.T., DUNIN-KEPLICZ, B.M., JENNINGS, N.R. and TREUR, J. (1997), DESIRE: modelling multi-agent systems in a compositional formal framework, In: HUHNS, M. and SINGH, M. (Eds.), *International Journal of Co-operative Information Systems*, *IJCIS* vol. 6 (1), special issue on Formal Methods in Co-operative Information Systems: Multi-Agent Systems, pp. 67-94.

BRAZIER, F.M.T., LANGEN, P.H.G. VAN, RUTTKAY ZS., and TREUR, J. (1994). On formal specification of design tasks. In GERO, J.S., and SUDWEEKS, F. (Eds.) *Artificial Intelligence in Design '94* Kluwer Academic Publishers, Dordrecht, pp. 535-552.

BRAZIER, F.M.T., LANGEN, P.H.G. VAN, and TREUR, J. (1996). Logical theory of design. In: *Advances in Formal Design Methods for CAD*, J.S. Gero (Ed.), Chapmann & Hall, New York, 1996, pp. 243-266.

BRAZIER, F.M.T., LANGEN, P.H.G. VAN, TREUR, J., WIJNGAARDS, N.J.E. and WILLEMS, M. (1996). Modelling an elevator design task in DESIRE: the VT example. In: SCHREIBER, A.TH., and BIRMINGHAM, W.P. (Eds.), Special Issue on Sisyphus-VT. *International Journal of Human-Computer Studies*, 1996, **44**, pp. 469-520.

BRAZIER, F. M. T., TREUR, J., and WIJNGAARDS, N. J. E. (1996b). Interaction with experts: the role of a shared task model. In: WAHLSTER, W. (ed.). *Proceedings European Conference on AI (ECAI '96)*, pp. 241-245. Wiley and Sons, Chichester.

BRAZIER, F.M.T., TREUR, J., WIJNGAARDS, N.J.E. and WILLEMS, M. (1996). Temporal semantics and specification of complex tasks. In: GAINES, B.R., and MUSEN, M.A. (Eds.), *Proceedings of the 10th Banff Knowledge Acquisition for Knowledge-based Systems workshop (KAW'96)*, Calgary: SRDG Publications, Department of Computer Science, University of Calgary, pages 15/1-15/17. To appear in: *Data and Knowledge Engineering*, 1998.

BREUKER, J.A. (1997). Problems in indexing problem solving methods. In: FENSEL, D. (Ed.) *Proceedings of the Problem Solving Methods for Knowledge Based Systems workshop*, IJCAI'97, pp. 19-35.

CORNELISSEN, F., JONKER, C.M., and TREUR, J. (1997). Compositional Verification of Knowlegde-Based Systems: A Case-Study for Diagnostic Reasoning. In: PLAZA, E. and BENJAMINS, R. (Eds.). *Knowledge Acquisition, Modeling and Management*, proceedings of the 10th European workshop, EKAW'97, Lecture Notes in Artificial Intelligence, **1319**, Springer, Berlin, pp. 65-80.

COYNE, R.D., ROSENMAN, M.A., RADFORD, A.D., BALACHANDRAN, M., and GERO, J.S. (1990). *Knowledge-based design systems* Addison-Wesley Publishing Company, Reading.

DENNETT, D.C. (1987). *The Intentional Stance*. MIT Press, Cambridge.

FENSEL, D. (1997). The Tower-of-Adapters Method for Developing and Reusing Problem-Solving Methods. In: PLAZA, E. and BENJAMINS, R. (Eds.). *Knowledge Acquisition, Modeling and Management*, proceedings of the 10th European workshop, EKAW'97, Lecture Notes in Artificial Intelligence, **1319**, Springer, Berlin, pp. 97-112.

FENSEL, D. (1997*b*). An Ontology-based Broker: Making Problem-Solving Method Reuse Work. In: *Proceedings of the Workshop on Problem-Solving Methods for Knowledge Based Systems at the 15th International Joint Conference on AI (IJCAI'97)*, Nagoya, Japan.

FENSEL, D., MOTTA, E., DECKER, S., and ZDRAHAL, Z. (1997). Using Ontologies for Defining Tasks, Problem-Solving Methods and their Mappings. In: PLAZA, E. and BENJAMINS, R. (Eds.). *Knowledge Acquisition, Modeling and Management*, proceedings of the 10th European workshop, EKAW'97, Lecture Notes in Artificial Intelligence, **1319**, Springer, Berlin, pp. 113-128.

FENSEL, D. and STRAATMAN, R. (1997). The Essence of Problem-Solving Methods: Making Assumptions to Gain Efficiency. In press: IJHCS 1997.

HARMELEN, F. VAN and TEIJE, A. TEN (1997). Characterising Problem Solving Methods by gradual requirements: overcoming the yes/no duality. Submitted to KEML'98 workshop.

JONKER, C.M. and TREUR, J. (1997). Compositional Verification and Validation of Multi Agent Systems: A Formal Analysis of Pro-Activeness and Reactiveness. In: DE ROEVER, W.P., LANGMAACK, H. and PRUELI, A. (Eds.) *Proceedings of the international symposium on Compositionality* (COMPOS'97), Springer Verlag, to appear.

LANGEN P.H.G. VAN, BRAZIER F.M.T., DIEPENMAAT H.B., and PULLES M.P.J. (1995). Inventarisatie van de milieubelasting van industriele processen vanuit kennistechnologisch perspectief (Inventarisation of environmental consequences of industrial processes from a knowledge intensive perspective). In Dutch, Technical Report number TNO-MW-R95/139, TNO Milieuwetenschappen, Delft, 54 pages.

MOTTA, E. and ZDRAHAL, Z. (1996). Parametric Design Problem Solving. In: GAINES, B.R., and MUSEN, M.A. (Eds.), *Proceedings of the 10th Banff Knowledge Acquisition for Knowledge-based Systems workshop (KAW'96*), Calgary: SRDG Publications, Department of Computer Science, University of Calgary, pages 9/1-9/20.

NWANA, H.S. (1996). Software agents: an overview. In: The Knowledge Engineering Overview, Cambridge University Press, **11**(3), pp. 205-244.

ORSVäRN, K. (1996). *Knowledge Modelling with Libraries of Task Decopmosition Methods*. PhD Thesis, Royal Institute of Technology & Swedish Institute of Computer Science, SICS dissertation series 22, Kista, Sweden.

PETRIE, C J, CUTKOSKY, M R, and PARK, H (1994). Design space navigation as a collaborative aid. In: GERO, J S, and SUDWEEKS, F (Eds.) *Artificial Intelligence in Design '94* Kluwer Academic Publishers, Dordrecht, pp 611-623.

SHOHAM, Y. (1993). Agent-oriented programming. *Artificial Intelligence*, **60**, pp. 51-92.

SMITH, I.F.C., and BOULANGER, S. (1994). *Knowledge representation for preliminary stages of engineering tasks* Knowledge Based Systems, **7**, pp. 161-168.

SMITHERS, T. (1994). On the nature of theory and design. In SMITHERS, T. (Ed.) *Workshop Notes of the AID '94 Workshop on the Role and Nature of Theory in AI in Design Research*.

WIELINGA, B.J., AKKERMANS, J.M. and SCHREIBER, A. TH. (1995). A Formal Analysis of Parametric Design Problem Solving. In: GAINES, B.R., and MUSEN, M.A. (Eds.), *Proceedings of the 9th Banff Knowledge Acquisition for Knowledge-based Systems workshop (KAW'95*), Calgary: SRDG Publications, Department of Computer Science, University of Calgary, pages 37/1-37/15.

WIELINGA, B.J., SCHREIBER, A.TH., and BREUKER, J.A. (1992). KADS: a modelling approach to knowledge engineering. *Knowledge Acquisition*, **4**, pp. 5-53.

WOOLDRIDGE, M.J. and JENNINGS, N.R. (1995). Intelligent Agents: Theory and Practice. In: Knowledge Engineering Review, **10**(2), pp. 115-152.

---

[*]

Current address: KSI Lab, Department of Computer Science, University of Calgary, 2500 University Drive NW, Calgary Alberta, T2N 1N4 Canada, niek@cpsc.ucalgary.ca

---

Page generated by Niek Wijngaards, February 28th.