Centrum voor Wiskunde en Informatica
REPORT*RAPPORT*

Deliberate evolution in multi-agent systems

F.M.T. Brazier, C.M. Jonker, J. Treur, N.J.E. Wijngaards

Software Engineering (SEN)

SEN-R9836 December 31, 1998

Report SEN-R9836
ISSN 1386-369X

CWI
P.O. Box 94079
1090 GB Amsterdam
The Netherlands

Deliberate Evolution in Multi-Agent Systems

Frances M.T. Brazier, Catholijn M. Jonker, Jan Treur[1], and Niek J.E. Wijngaards

[1]CWI
P.O. Box 94079, 1090 GB Amsterdam, The Netherlands

and

Vrije Universiteit Amsterdam, Department of Artificial Intelligence
De Boelelaan 1081a, 1081 HV Amsterdam, The Netherlands
Email: treur@cs.vu.nl URL: http://www.cs.vu.nl/~treur

ABSTRACT

This paper presents an architecture for an agent capable of deliberation about the creation of new agents, and of actually creating a new agent in the multi-agent system, on the basis of this deliberation. After its creation the new agent participates fully in the running multi-agent system. The agent architecture is based on an existing generic agent model, and includes explicit formal conceptual representations of both structures of agents and (behavioural) properties of agents that can be used as requirements. Moreover, to support the deliberation process the agent has explicit knowledge of relations between structure and properties of agents. To actually create a new agent at run-time on the basis of the results of deliberation, the agent executes a creation action in the material world, which leads to a world state update to include the new agent, after which the new agent functions within the multi-agent system. This approach enables the design of evolution processes in societies of agents for which the evolution is not a merely material process which takes place in isolation from the mental worlds of the agents, but allows for interaction between mental and material processes. A combined mind-matter approach results in which the agents in a society can deliberatively influence the direction of the evolution, comparable to the potential offered by genetic engineering. The architecture has been designed using the compositional development method DESIRE, and has been tested in a prototype implementation. It is discussed how the approach introduced here can be used as a basis for automatic evolution of multi-agent systems for Electronic Commerce.

1991 Computing Reviews Classification System: I.2.11
Keywords and Phrases: evolution, deliberation, multi-agent system, creation
Note: work carried out in theme SEN4; the NWO-project nr 612-322-316 Evolutionary Design in Knowledge Based Systems (REVISE) provided external funding for part of this work.

1. INTRODUCTION

Evolution in societies of agents is a challenging phenomenon, both from a fundamental perspective and from an applied perspective. In the literature often genetic programming approaches are used and relatively simple agents are considered, which have no deliberate influence on the direction of the process of evolution; e.g., (Cetnarowicz, Kisiel-Dorohinicki and Nawarecki, 1996; Numaoka, 1996). This results in agents that have limited autonomy viewed on a time scale covering several generations: the agents are passive with respect to the evolution process, evolution just happens. From a historical biological perspective all species have always been subject to non-autonomous evolution (with the exception of influence by selective breeding strategies). Nowadays the acquistion of knowledge on the relation between DNA structures and properties of individuals, and techniques to manipulate DNA structures opens the perspective of deliberately influencing the direction of evolution, at least partially. This perspective introduces serious, unresolved ethical dilemmas when applied to the natural society.

Artifical societies of agents make it possible to experiment with deliberate evolution processes without these ethical dilemmas. Moreover, interesting application areas have emerged: applications related to the Internet, such as Electronic Commerce can highly benefit if societies of agents can evolve automatically, and if within such an evolution process the agents can deliberatively influence the direction of the evolution.

This paper introduces a mind-matter approach to deliberate evolution. An architecture is presented for agents that are able to deliberately create new agents, or even modify or delete existing agents. For example, Internet

agents that are capable of deliberately creating new agents to assist them in information gathering, or agents that are capable of creating interface agents tuned to specific users, are agents of this type. As part of the deliberation, the type of agent introduced is able to reason about required or desired (behavioural) *properties* of an agent to be created (in a metaphoric sense comparable to the traditional biological notion of phenotype), and is able to generate the intended goal to have an agent with these properties. To achieve this intended goal, an action in the material world is needed. Since actions are only able to create agents with a given material *structure*, reasoning about the structure (in a metaphoric sense comparable to the notion of genotype) of an agent to be created is involved, and reasoning about the relation between structure and properties. Given the intended goal of creating an agent with properties as intended by the creator, an intended action is generated in order to achieve the intended goal: a creation action for a new agent with a structure such that once created, it will have the intended properties.

The architecture introduced was designed using the compositional development method for multi-agent systems DESIRE (Brazier, Dunin-Keplicz, Jennings and Treur, 1995), and implemented using the DESIRE software environment. In Section 2 a global view of the approach is described. The deliberation model within an agent is described in Section 3. The agent is based on an existing generic agent model, and includes a refinement of a generic model for design (Brazier, Langen, Ruttkay and Treur, 1994), in which strategic reasoning and dynamic management of requirements are explicitly modelled. In Section 4 the model for the actual creation of an agent is described, based on the *mind-matter approach* introduced in (Jonker and Treur, 1997). In Section 5 the results and perspectives are discussed.

2. THE GLOBAL VIEW

To design an agent capable of deliberately creating or modifying agents, the following desiderata were taken as a point of departure:

A. A *model for deliberation* about agent creation or modification, covering:
   1. decisions on properties of an agent to be created or modified: generating the intended goal
   2. decisions on the structure of an agent to be created or modified, in relation to required or desired properties: generating the intended creation or modification action
   3. decisions to actually create or modify an agent according to a structure that was determined: performing the intended creation or modification action

B. A *model for effectuation* of agent creation or modification in the material world
   1. execution of a creation or modification action that is performed by the agent
   2. updating the world state to incorporate the intended effects of an executed creation or modification action

The model for deliberation is designed in such a way that the agent has:

- an explicit formal representation at a conceptual level of required or desired (behavioural) properties of agents to be created or modified (supporting A1.)
- an explicit formal representation at a conceptual level of structures of an agent to be created or modified (supporting A2.)
- knowledge to derive refinements of the required or desired properties that are sufficiently specific to be related to specific structures (supporting A1.)
- knowledge to relate specific properties to specific structures of an agent to be created or modified (supporting A2.)
- knowledge to make decisions on when to perform creation or modification actions (supporting A3.)

The model for effectuation is designed in such a way that:

- the agent executes intended creation or modification actions in the material world (supporting B1.)
- the external world has a mechanism such that upon performance of a creation or modification action, it will update the world state on the basis of the structural information conveyed in the action, thus actually creating or modifying (while the multi-agent system is running) the new or existing agent and its programme code (supporting B2.)
- the new or modified agents fully participate within the running multi-agent system in which after being created or modified by the material manipulations (supporting B2.)

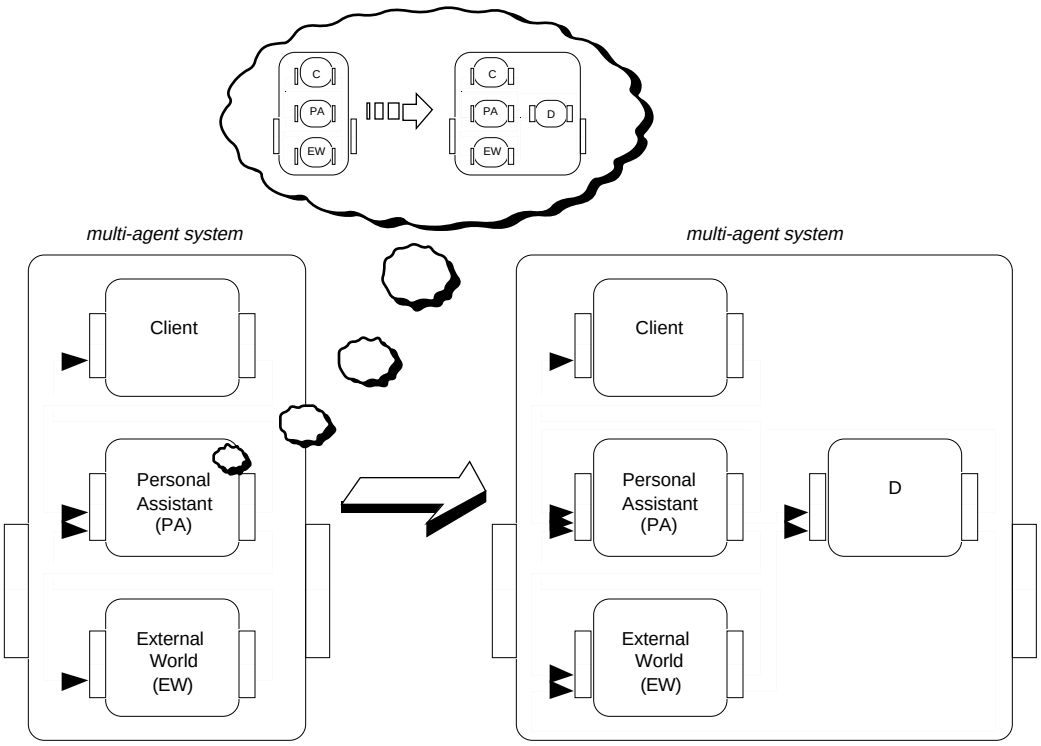


Figure 1: One agent deliberately creates another agent

In Figure 1 a sketch of the creation or modification process is depicted. The left box contains the multi-agent system before modification (consisting of the agents Personal Assistant, Client and External World), the right box after modification (with an additional agent D included). The Personal Assistant (PA) plays the role of the creating agent. It has internal (mental) representations of the multi-agent system before modification and designs a modification of the system by adding an agent D to the system. After this design process it effectuates the design by execution of the creation action in the External World, which represents all material aspects, including the material aspects of the agents.

3. A Model for Deliberation about Agent Creation

For a global idea of a deliberative agent, a sketch of the pattern of deliberation is as follows:

- the agent generates an intended goal to have a new or modified agent with particular desired or required properties (for example, by adopting this goal from another agent)
- on the basis of the intended goal, the agent determines refined properties for the new or modified agent, and generates more specific intended goals referring to the specific refined properties
- on the basis of the specific intended goals the agent determines a structure such that an agent with this structure would satisfy the specific properties to which the specific intended goals refer
- on the basis of this structure the agent generates an intended action: the action to create an agent with the structure as determined

This reasoning pattern is modelled in more detail by the generic model for design introduced in (Brazier, Langen, Ruttkay and Treur, 1994). A design model is applicable in this situation as intended goals and their related properties are viewed as requirements and the structure (e.g., of an agent) is the description of the object of design.

The agent architecture proposed in this paper has been designed in a compositional manner as a refinement of an existing generic agent model (Brazier, Jonker and Treur, 1997), depicted in Figure 2, by specialisation of its component Agent Specific Task using the generic model for design. Moreover, the resulting model has been instantiated with ontologies and knowledge on the domain of agents.

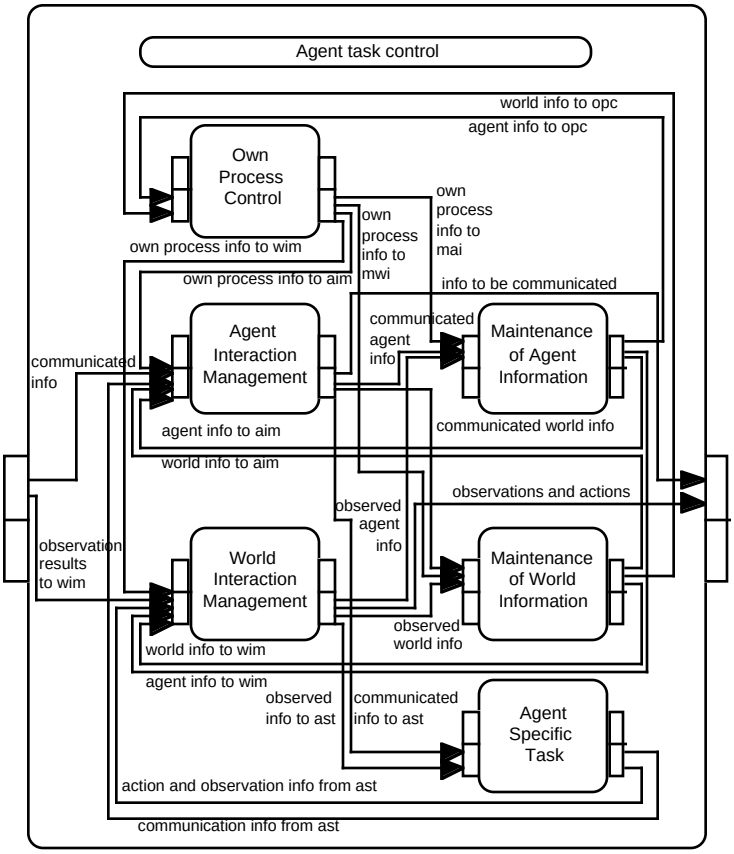


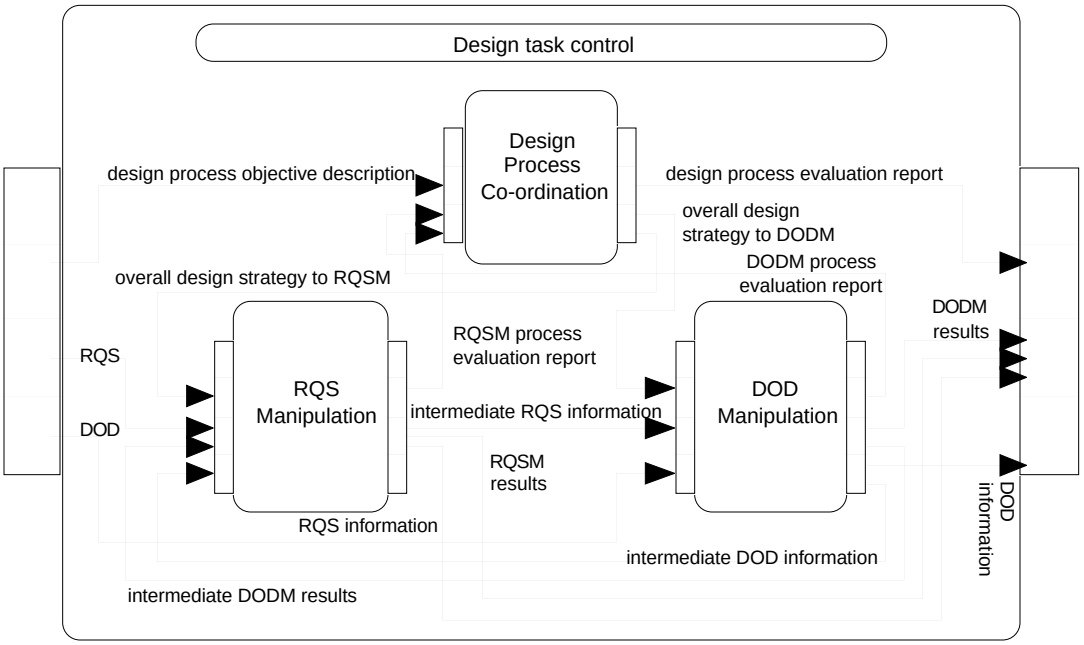
Figure 2: Generic Agent Model

3.1. The processes involved in the deliberation process

The compositional *generic model of design* (see Figure 3) in which reasoning about requirements and their qualifications, reasoning about design object descriptions and reasoning about the design process are distinguished, is based on a logical analysis of design processes (Brazier, Langen and Treur, 1996) and on analyses of applications, including elevator configuration (Brazier, Langen, Treur, Wijngaards and Willems, 1996) and design of environmental measures (Brazier, Treur and Wijngaards, 1996). The model 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 domain, and about design strategies.

An initial design problem statement is expressed 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.

During design the considered subsets of the set of requirements may change as may the requirements themselves. The same holds for design object descriptions representing the structure of the object to be designed. Figure 3 shows two levels of composition of the generic model for design. Three processes are shown at the top level, together with the information exchange. One of these three processes, Design Object Description Manipulation, is shown to consist of four processes and information exchange.



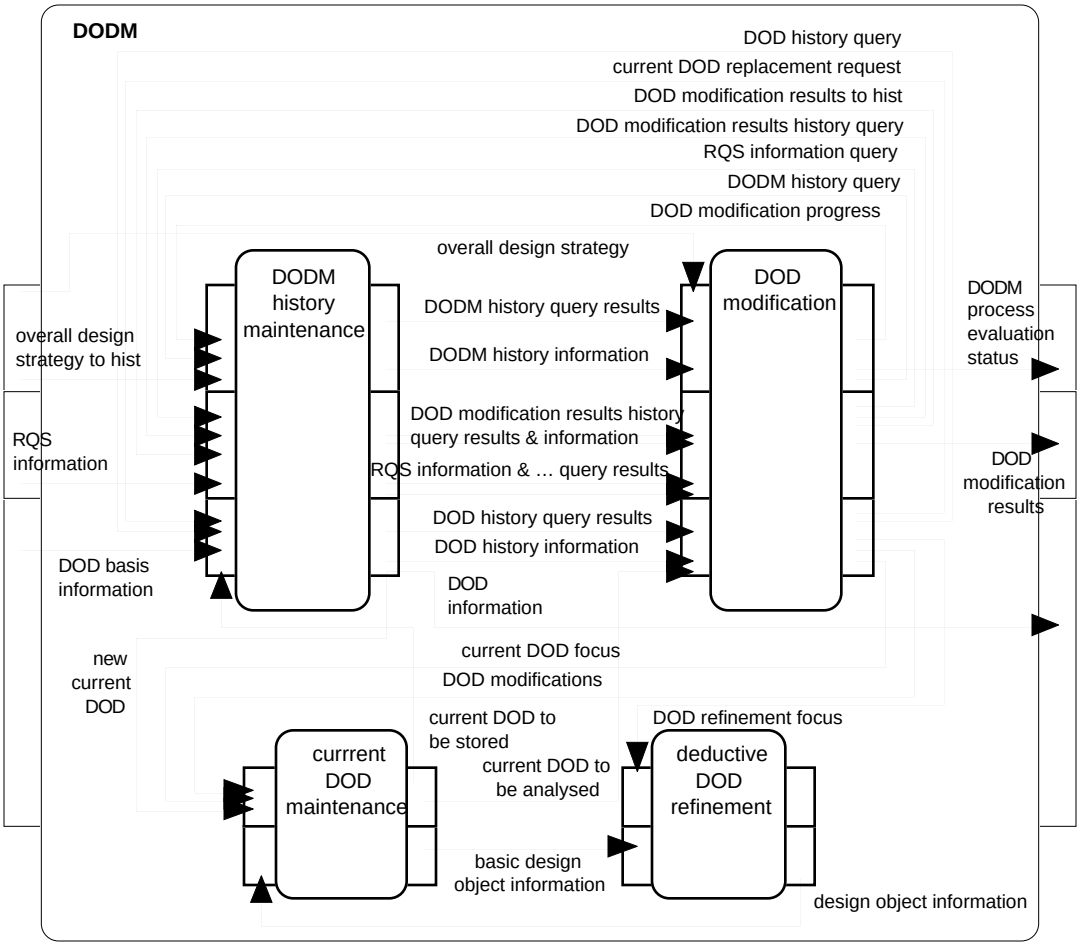


Figure 3: A generic model of design: two process abstraction levels

The four processes (see Figure 3) 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.

The four processes 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 generic design model has been specialised three more process abstraction levels deeper. For reasons of space limitation these are not presented.

3.2. Ontologies and knowledge involved in the deliberation process

The generic model of design used in the agent can, in principle, be used for any domain of application. It was applied to the domain of compositional agent design. A formal ontology of properties of agents has been developed (see Section 3.2.1). Knowledge has been identified that can be used to reason about these properties, to derive more specific properties by refining the original properties. These more specific properties play a crucial role in the design process: they guide the direction in which solutions are sought. Moreover, a specific formal ontology for the structure of agents has been defined (see Section 3.2.2). The deliberation process employs this knowledge and these ontologies to arrive at a creation or modification action on the basis of the intended goal.

3.2.1. Representation of properties of agents

Requirements are formulated in terms of abilities and properties of agents and the external world. Abilities and properties can be assigned to

- individual agents,
- the external 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.

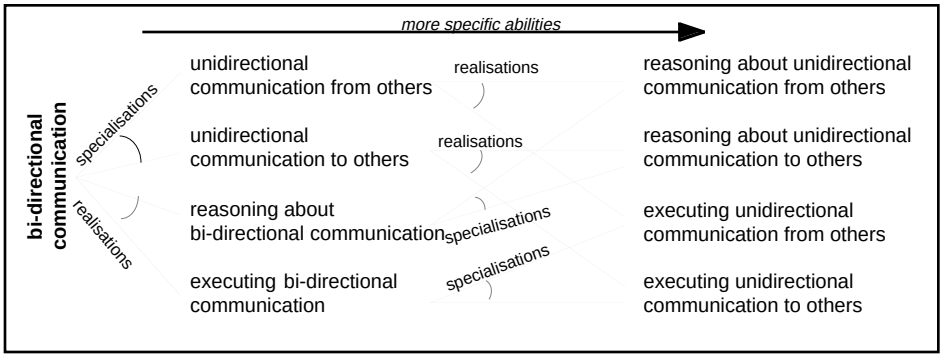


Figure 4: Refinements of the ability of bi-directional communication

Abilities of agents such as co-operation, bi-directional communication, and world interaction are often needed for agents to jointly be able to perform a certain task. In Figure 4 the ability of bi-directional communication and its refinements are depicted. For a description of other agent abilities see (Brazier, Jonker, Treur and Wijngaards, 1998). The ability of bi-directional communication can be refined, both with respect to its *specialisation* (refinement of the ability into more specific abilities) and with respect to its *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).

Figure 4 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 the ability 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.

Knowledge on refinements of the ability of bi-directional communication is formally represented as shown below. Meta-reasoning is employed to decide which refinement alternative should be employed for which ability.

EXAMPLE REPRESENTATION OF REQUIREMENTS REFINEMENT KNOWLEDGE

```
if      is_qualified_requirement_selected_as_focus( QR: qualified_requirement_name )
  and   holds( is_qualified_requirement(        QR: qualified_requirement_name,
                                                Q: requirement_qualification,
                                                R: requirement_name )
  and   holds( refers_to_requirement(   R: requirement_name,
                                                has_property( A: agent_name,
                                                        is_capable_of_bidirectional_communication_
                                                        with( A2: agent_name )) ),
                                                pos )
  and   refinement_alternative( specialisations )

then    addition_to_current_RQS(
          is_qualified_requirement(     new_name( QR: qualified_requirement_name, a ),
                                                Q: requirement_qualification,
                                                new_name( R: requirement_name, a ) ) )
  and   addition_to_current_RQS(
          refers_to_requirement(        new_name( R: requirement_name, a ),
                                                has_property( A: agent_name,
                                                        is_capable_of_unidirectional_communication_
                                                        from( A2: agent_name ) ) )
  and   addition_to_current_RQS(
          is_qualified_requirement(     new_name( QR : qualified_requirement_name, b ),
                                                Q: requirement_qualification,
                                                new_name( R: requirement_name, b ) ) )
  and   addition_to_current_RQS(
          refers_to_requirement(        new_name( R: requirement_name, b ),
                                                has_property( A: agent_name,
                                                        is_capable_of_unidirectional_communication_
                                                                to( A2: agent_name ) ) )
  and   addition_to_current_RQS(
          is_qualified_requirement(     new_name( QR: qualified_requirement_name, c ),
                                                Q: requirement_qualification,
                                                new_name( R: requirement_name, c ) ) )
  and   addition_to_current_RQS(
          refers_to_requirement(        new_name( R: requirement_name, c ),
                                                has_property( A: agent_name,
                                                is_capable_of_combining_unidirectional_
                                                communication_from_and_to( A2: agent_name ) ) );
```

Top-level requirements are refined into more specific requirements during a design process. The result is the construction of a specific hierarchy of requirements, which adheres to the requirements ontology and refinement knowledge. Figure 5 shows an example of (part of) such a requirements refinement hierarchy. The current prototype agent makes extensive use of the requirements ontology, generic models and design object building blocks. The design process is fairly linear, in the sense that few options are generated and selected. The most refined requirements are almost directly operationalisable by building blocks for design object descriptions. A

specific design requirement, currently in focus in DOD modification, is broken up (i.e., refined) into smaller properties: assessment points. These assessment points can be tested for, and when not yet realised, building blocks related to an assessment point can be added to the current design object description. The generation of options for sets of qualified requirements and design object descriptions involving explicit strategic knowledge can be incorporated in the design model, as described by (Brazier, Langen, and Treur, 1998).

```
has_property( agent_D, is_capable_of_active_observation_in( world_W ) )
    has_property( agent_D, is_capable_of_processing_observation_results_from( world_W ) )
        has_property( agent_D, is_capable_of_reasoning_about_processing_observation_results_from( world_W ) )
        has_property( agent_D, is_capable_of_executing_processing_observation_results_from( world_W ) )
        has_property( agent_D, is_capable_of_combining_reasoning_about_and_executing_processing_observation_results( world_W ) )
    has_property( agent_D, observation_initiation_in( world_W ) )
        has_property( agent_D, is_capable_of_reasoning_about_observation_initiation_in( world_W ) )
        has_property( agent_D, is_capable_of_executing_observation_initiation_in( world_W ) )
        has_property( agent_D, is_capable_of_combining_reasoning_about_and_executing_observation_initiation_in( world_W ) )
    has_property( agent_D, is_capable_of_combining_processing_observation_results_and_observation_initiation_in( world_W ) )
```

Figure 5: Requirement refinement hierarchy constructed by the prototype design agent

3.2.2. Representation of the structure of agents

The implication of designing (parts of) a multi-agent system, is that a multi-agent system is the object of design, and as such its structure should be formally represented in a design object description. In this paper the design object structure is assumed to be a compositional. The assumption underlying this decision is that a compositional structure facilitates the process of (re-)design. The compositional formal specification language underlying DESIRE forms an adequate basis for such a design object description representation.

The description of the compositional structure is augmented with a description relating existing structures to generic models. This provides information useful for documentation purposes and it also provides valuable information for the identification of abilities and properties.

EXAMPLE REPRESENTATION OF AN AGENT DESIGN

The agent needs a representation of a multi-agent system including agents and the external world. Part of the top level of the multi-agent system can be represented as follows:

```
is_top_level(c_00)
corresponds_with(c_00, mas_S)
corresponds_with(c_01, agent_A)
corresponds_with(c_04, world_W)
has_characterisation(c_00, generic, multi_agent_system)
has_characterisation(c_01, generic, agent)
has_characterisation(c_4, generic, external_world)
corresponds_with(lm_01, active_observations)
has_subcomponent(c_00, c_01)
subcomponent(c_00, c_04)
has_information_link(c_00, lm_01)
has_source_component(lm_01, c_01)
has_destination_component(lm_01, c_04)
```

Unique identifiers are assigned to components and information links so that names of links and components can be reused in several parts of the composition.

4. A MODEL FOR EFFECTUATION OF AGENT CREATION

After the deliberation on the creation process has been completed, and the agent decided to effectuate the modifications, a number of steps take place.

4.1. Preparation of the effectuation

As discussed in (Jonker and Treur, 1997) effectuation of the modification of the design can be modelled by changing the material representation of the multi-agent system within the external world.

> EXAMPLE CHANGING THE MATERIAL REPRESENTATION OF THE MULTI-AGENT SYSTEM.
>
> The resulting design object description, dod_55, contains the complete set of modifications that are to be made to the multi-agent system mas_S (including the creation of a new agent). The design object properties that together form dod_55, are represented by statements such as:
>
> has_DOD_characteristic(dod_55, corresponds_with(c_05, agent_D))
>
> has_DOD_characteristic(dod_55, has_subcomponent(c_00, c_05))
>
> These statements reside at a meta-level with respect to design object description statements. The second argument of each statement expresses relationships within the design object description.

The modification action, and the structural changes in the material representation of the multi-agent system to which it refers changes are transferred from the agent to the external world.

> EXAMPLE EFFECTUATION ACTION FOR MODIFYING THE MULTI-AGENT SYSTEM.
>
> Within the agent an intended action is generated to effectuate the modification of the current multi-agent system:
>
> to_be_performed(modify_according_to(dod_55));

4.2. Execution of the creation action within the external world

To be able to execute the creation action in the external world, the external world needs to have certain properties. These properties are related to how "equipped" the world is to handle interaction with agents. There are two generic properties needed for the interaction of agents with the external world: the processing of observations and the processing of actions. Observation of the external world was needed to inform agent A of the current material representation of the multi-agent system.The property of processing actions can be refined into the properties:

- the external world can receive initiated actions, and the related information, and
- the external world can perform actions (effectuation of the physical effects of actions).

To change the number of agents and their characteristics, the external world has to adapt the executable specification of that system while the system is running. This implies that the parts of the system that are affected by the modifications need to be interrupted, their information states stored, after which the executable specification of those parts need to be modified, and the modified system need to be reactivated with the correct information states.

> EXAMPLE RESULT OF THE EFFECTUATION ACTION.
>
> The external world world_W effectuates the creation action and modifies the multi-agent system according to the given modifications.
>
> As an agent in the multi-agent system, agent D and receives a request from agent A: would it like to find out more about YYY?. The agent D gathers information on subject YYY by initiating observations in the world W, and interpreting the observation results. Once the answer is found, agent D reports its findings to agent A. Agent A can then which finally answer the question of the client.

5. DISCUSSION

Research within multi-agent systems research has focussed on the behaviour of individual agents and their interaction. The dynamic creation of new agents within an existing multi-agent system, on the basis of the identification of newly required functionality and behaviour, is an area on which little research has focussed. Most of the research in the area of dynamic agent creation is based on a *genetic programming approach*; e.g., (Cetnarowicz, Kisiel-Dorohinicki and Nawarecki, 1996; Numaoka, 1996). These approaches do not take into account the possibility that agents deliberately influence their own evolution. The approach taken in this paper is that an existing agent is capable of deliberately designing a new agent tuned to the needs as perceived and then be capable of bringing this agent to life. As genetic programming approaches in principle are based on manipulation of the material representations of the agents within the material world (outside the mental world of the agents), the *mind-matter approach* introduced here exploits the duality between the potential of deliberation on the one hand, and material manipulation possibilities on the other hand. In this sense the approach introduced adds to the genetic programming approach the possibility of a mental perspective from (individual or societies of) agents.

To design an agent capable of deliberately creating new agents, insight is required in the type of agent to be designed and the deliberation model to be used. The architecture of the agent introduced here is based on an existing generic agent model, and includes a refinement of a generic model of design. It combines results from the area of Multi-Agent Systems and the area of AI and Design. In this paper a compositional approach to agent design has been followed. An agent's abilities are related to the tasks an agent is able to perform. These abilities are the means with which both the existing agents' abilities are expressed. In addition, the properties of the multi-agent system and the external world are of importance. As such, this work is related to the properties distinguished with respect to problem solving methods (Benjamins, Fensel and stratman, 1996; Breuker, 1997; Fensel, Motta, Decker, and Zdrahal, 1997). Within the field of Knowledge Engineering properties of problem solving methods are used to support knowledge engineers during the design process: providing a means to describe existing generic components that may be used, modified or refined during a design process, depending on their applicability in a given situation. The Knowledge Engineering community has not focussed on abilities and properties of agents and their interaction, as was done in this paper.

The generic agent model has been developed on the basis of experience with the design of agent models of different kinds; for example, models for information gathering agents, co-operative agents for project co-ordination, BDI-agents, negotiating agents, broker agents, and agents simulating animal behaviour. The generic design model was developed and evaluated on the basis of experience with design applications in a number of domains; for example, design of sets of measures for environmental policy, aircraft design, and elevator design.

The refined model includes formalisations of the (compositional) structure of agents and (desired or required) properties of agents, and formalisations of agent design knowledge. After this proof of concept, the approach introduced will be applied in the domain of Electronic Commerce. Electronic Commerce necessarily involves interaction between human users in different types of organisations, and very dynamic automated environments, in which the parties involved are not known beforehand, and often change. In such environments human users can be supported by Personal Assistant Agents, which in turn make use of existing broker agents and other task-specific agents. Co-operation between these (human and computer) agents is to the advantage of all. To cope with the dynamic character of the environment, frequently new agents need to be created, or existing agents need to be modified, for specific purposes. Such frequent modification of an environment necessitates almost continuous maintenance. On the basis of the approach introduced here, a generic multi-agent Electronic Commerce environment will be developed in which a broker agent can dynamically reconfigure (parts of) the multi-agent system by adding or modifying Personal Assistant agents, broker agents and additional agents. More specifically, the aim is to develop a multi-agent broker architecture with a number of co-operative broker agents, Personal Assistant agents, and task specific agents. Each broker agent can dynamically configure and implement new agents or modify existing agents as part of the multi-agent system as follows:

- if new users (clients) subscribe to a broker agent, Personal Assistant agents tuned to the requirements imposed by this user, may be created, or existing Personal Assistant agents may be modified, due to changed requirements.
- if required in view of the load of an existing broker agent, new broker agents can be added to distribute the load (and avoid overload of the existing broker agent), or existing broker agents can be modified.
- if opportune, or requested, new agents may be created to perform specific tasks, fulfilling certain dynamically imposed requirements; for example, for searching the Internet for specific types of information, or shadowing information at a specific site.

Recently a few applications of broker agents have been addressed for this area; see, for example (Chavez and Maes, 1996; Chavez, Dreilinger, Guttman and Maes, 1997; Kuokka and Harada, 1995; Martin, Moran, Oohama, Cheyer, 1997; Sandholm and Lesser, 1995; Tsvetovatyy and Gini, 1996). However, these applications have been implemented without an explicit design at a conceptual level, and without taking into account the dynamic requirements imposed by the domain of application and the maintenance problem implied by this dynamic character.

A principled approach to the design of the architecture is of crucial importance: a generic conceptual architecture of an agent is needed to support the (re)design process needed for dynamic creation or modification of agents based on dynamically imposed requirements. An approach in which conceptual design is the basis for structure-preserving (formal) detailed design and operational design, can provide the means to model, specify and implement the flexible structures required, as shown by the prototype implementation.

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 software environment. This research has been (partially) supported by NWO-SION within project 612-322-316: 'Evolutionary design in knowledge-based systems' (REVISE).

REFERENCES

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), 408-412, Wiley and Sons, Chichester.

Brazier, F.M.T., Dunin-Keplicz, B.M., Jennings, N.R. and Treur, J. 1997. Formal Specification of Multi-Agent Systems: a Real World Case, In: Lesser, V. (ed.), *Proceedings First International Conference on Multi-Agent Systems, ICMAS'95*, 25-32, MIT Press. Extended version in: Huhns, M. and Singh, M. (eds.), *International Journal of Co-operative Information Systems, IJCIS* vol. 6(1):67-94, (1997), special issue on Formal Methods in Co-operative Information Systems: Multi-Agent Systems.

Brazier, F.M.T., Jonker, C.M., Treur, J. and Wijngaards, N.J.E. 1998. The Role of Abilities of Agents in Re-design. In: Gaines, B. and Musen, M. (eds). *Proc. of the 11th Knowledge Acquisition Workshop, KAW'98*. Banff. University of Calfgary. URL: http://ksi.cpsc.ucalgary.ca/ KAW/KAW98/brazier2.

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*, 535-552, Kluwer Academic Publishers, Dordrecht.

Brazier, F.M.T., Langen, P.H.G. van, and Treur, J. 1996. A logical theory of design. In: *Advances in Formal Design Methods for CAD*, J.S. Gero (ed.), 243-266, Chapmann & Hall, New York, 1996.

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, 44:469-520.

Brazier, F. M. T., Treur, J., and Wijngaards, N. J. E. 1996. Interaction with experts: the role of a shared task model. In: Wahlster, W. (ed.). *Proceedings European Conference on AI (ECAI '96)*, 241–245,Wiley and Sons, Chichester.

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, 19-35.

Cetnarowicz, K., Kisiel-Dorohinicki, M., and Nawarecki, E. 1996. The Application of Evolution Process in Multi-Agent World to the Prediction System. In: Tokoro, M. (Ed.) *Proceedings of the Second International Conference on Multi-Agent Systems*, ICMAS'96, 26-32, AAAI press, Menlo Park CA.

Chavez, A., Maes, P. 1996. Kasbah: An Agent Marketplace for Buying and Selling goods. In: *Proc. of the First International Conference on the Practical Application of Intelligent Agents and Multi-Agent Technology*, PAAM'96, 75-90, The Practical Application Company Ltd, Blackpool.

Chavez, A., Dreilinger, D., Gutman, R., Maes, P. 1997. A Real-Life Experiment in Creating an Agent Market Place. In: *Proc. of the Second International Conference on the Practical Application of Intelligent Agents and Multi-Agent Technology*, PAAM'97, 159-178, The Practical Application Company Ltd, Blackpool

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.

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, 1319:113-128, Springer, Berlin.

Jonker, C.M., and J. Treur 1997. Modelling an Agent's Mind and Matter. In: *Proceedings of the 8th European Workshop on Modelling Autonomous Agents in a Multi-Agent World, MAAMAW'97*, Lecture Notes in AI, 1237:210-233, Springer Verlag.

Kuokka, D., Harada, L. 1995. On Using KQML for Matchmaking. In: V. Lesser (ed.), Proc. of the First International Conference on Multi-Agent Systems, ICMAS'95, 239-245, MIT Press, Cambridge, MA.

Martin, D., Moran, D., Oohama, H., Cheyer, A. 1997. Information Brokering in an Agent Architecture. In: *Proc. of the Second International Conference on the Practical Application of Intelligent Agents and Multi-agent Technology*, PAAM'97, 467-486, The Practical Application Company Ltd, Blackpool.

Numaoka, C. 1996. Bacterial Evolution Algorithm for Rapid Adaptation. In: Van de Velde, W. and Perram, J.W. (Eds.) *Proceedings of the 7th European Workshop on Modelling Autonomous Agents in a Multi-Agent World*, MAAMAW'96, Lecture Notes in Artificial Intelligence, 1038:139-148, Springer.

Sandholm, T., Lesser, V. 1995. Issues in Automated Negotiation and Electronic Commerce: Extending the Contract Network. In: V. Lesser (ed.), *Proc. of the First International Conference on Multi-Agent Systems*, ICMAS'95, 328-335, MIT Press, Cambridge, MA.

Smith, I.F.C., and Boulanger, S. 1994. *Knowledge representation for preliminary stages of engineering tasks* Knowledge Based Systems, 7:161-168.

Smithers, T. 1994. On knowledge level theories of design process. In Gero, J.S., and Sudweeks, F. (eds.) *Artificial Intelligence in Design '96*, 561-579, Kluwer Academic Publishers, Dordrecht.

Tsvetovatyy, M., Gini, M. 1996. Toward a Virtual Marketplace: Architectures and Strategies. In: *Proc. of the First International Conference on the Practical Application of Intelligent Agents and Multi-Agent Technology*, PAAM'96, 597-613, The Practical Application Company Ltd, Blackpool.

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