Compositional Design of a Generic Design Agent

Frances M.T. Brazier, Catholijn M. Jonker, Jan Treur, and Niek J.E. Wijngaards*

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

Abstract

This paper presents a generic architecture for a design agent. The design agent is based on an existing generic agent model, and includes a refinement of a generic model for design, in which strategic reasoning and dynamic management of requirements are explicitly modelled. The generic architecture has been designed using the compositional development method DESIRE, and has been used to develop a prototype design agent for automated agent design.

1 Introduction

Design is a task often performed by one or more specialised agents. Other agents interact with such 'design agents' by, for example, providing qualified requirements, initial (partial) design object descriptions, and design process objectives. Specialised agents are often encountered in human society: e.g., architects are specialised agents: their area of expertise is the design of buildings. A design agent generates a design object description on the basis of the information received from other agents, and makes results of the design process available to other agents.

In this paper a generic architecture is introduced for a design agent, which was modelled using the compositional development method for multi-agent systems DESIRE (Brazier, Dunin-Keplicz, Jennings and Treur, 1997). In Section 2 this compositional development method is briefly introduced. The design agent is based on an existing generic agent model (Section 3), 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 (Sections 4 and 5). Knowledge of different types can be included, for example, knowledge that can be used to derive whether a given design object description satisfies given properties (e.g., requirements), and knowledge that can be used to derive how to refine requirements into more specific requirements. In Section 6 the application of the generic design agent to compositional system design is described. In Section 7 the results are discussed and a perspective is sketched of the use of the design agent in Electronic Commerce applications.

2 Compositional Design of Agents

The design agent described in this paper has been developed using the compositional development method DESIRE for multi-agent systems (framework for Design and Specification of Interacting Reasoning components; cf. (Brazier et al., 1997). The development of a multi-agent system is supported by graphical design tools within the DESIRE software environment. Translation to an operational system is straightforward; the software environment includes implementation generators with which formal specifications can be translated into executable code of a prototype system. In DESIRE, a design consists of knowledge of the following three types: process composition, knowledge composition, and the relation between process composition and knowledge composition. These three types of knowledge are discussed in more detail below.

2.1 Process 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.

Identification of Processes at Different Levels of Abstraction. Processes can be described at different levels of abstraction; for example, the process of the multi-agent system as a whole, processes defined by individual agents and the external world, and processes defined by task-related components of individual agents. The identified processes are modelled as *components*. For each process the *input and output information types* are modelled. The identified levels of process abstraction are modelled as *abstraction/specialisation relations* between components: components may be *composed* of other components or they may be *primitive*. Primitive components may be either reasoning components (i.e., based on a knowledge base),

* Currently at: Software Engineering Research Network, Department of Computer Science, University of Calgary, Canada

or, components capable of performing tasks such as calculation, information retrieval, optimisation. These levels of process abstraction provide process hiding at each level.

Composition of Processes. The way in which processes at one level of abstraction are composed of processes at the adjacent lower abstraction level is called *composition*. This composition of processes is described by a specification of the possibilities for *information exchange* between processes (*static view* on the composition), and a specification of *task control knowledge* used to control processes and information exchange (*dynamic view* on the composition).

2.2 Knowledge Composition

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.

Identification of knowledge structures at different abstraction levels. The two main structures used as building blocks to model knowledge are: *information types* and *knowledge bases*. Knowledge structures can be identified and described at different levels of abstraction. At higher levels details can be hidden. An *information type* defines an ontology (lexicon, vocabulary) to describe objects or terms, their sorts, and the relations or functions that can be defined on these objects. Information types can logically be represented in order-sorted predicate logic. A *knowledge base* defines a part of the knowledge that is used in one or more of the processes. Knowledge is represented by formulae in order-sorted predicate logic, which can be normalised by a standard transformation into rules.

Composition of Knowledge Structures. Information types can be composed of more specific information types, following the principle of compositionality discussed above. Similarly, knowledge bases can be composed of more specific knowledge bases. The compositional structure is based on the different levels of knowledge abstraction distinguished, and results in information and knowledge hiding.

2.3 Relation between Process Composition and Knowledge Composition

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.

3 A Generic Agent Model

Agents are often designed to perform their own specific tasks, for example the design of an artefact. In addition, a number of generic agent tasks can be identified. This section describes a generic agent model in which such generic agent tasks are modelled. This model abstracts from the specific domain of application and can be (re)used for a large variety of agents. The model is based on the abilities associated with the notion of weak agency (Wooldridge and Jennings, 1995).

Instead of designing each and every new agent individually from scratch, a generic agent model can be used to structure the design process: the acquisition of a specific agent model is based on the generic structures in the model.

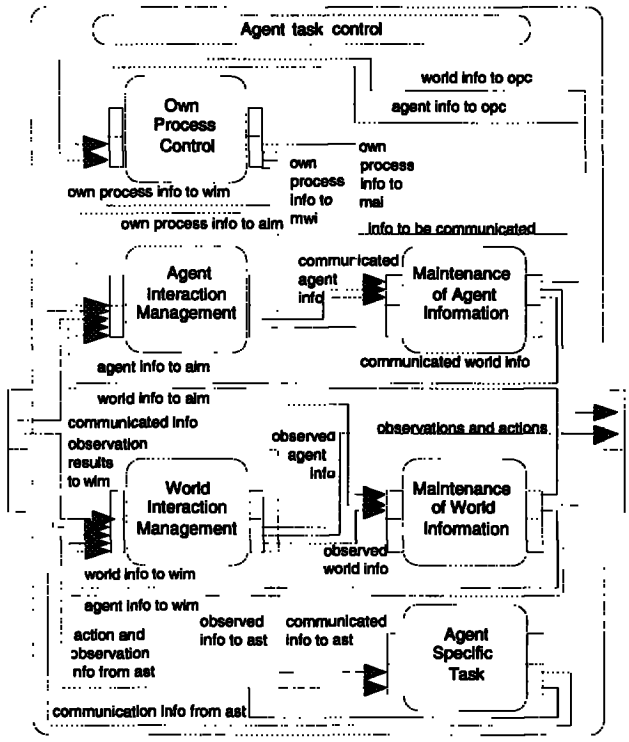

Figure 1. A Generic Agent Model.

The characteristics of weak agency 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 and to control these processes (own process control). Reactivity and social ability are related to the ability to be able to communicate with other agents (agent interaction management) and to interact with the external world (world interaction management). The ability to communicate with other agents and to interact with the external world often relies on the information an agent has of the world (maintenance of world information) and other agents (maintenance of agent information). The generic agent

model also includes an empty generic component to model the agent specific task. The tasks related to the generic abilities and agent specific tasks may be modelled by components within an agent as depicted in Figure 1. In addition to the sub-components, the model includes information links that specify which information is exchanged between components; these information links are named.

The exchange of information within the generic agent model can be described as follows. Observation results are transferred through the information link observation results to wim from the agent's input interface to the component world interaction management. In addition, the component world interaction management receives belief information from the component maintenance of world information through the information link world info to wim, and the agent's characteristics from the component own process control through the link own process info to wim. The selected actions and observations (if any) are transferred to the output interface of the agent through the information link observations and actions.

The component maintenance of world information receives meta-information on observed world information from the component world interaction management, through the information link observed world info and meta-information on communicated world information (through the link communicated world info) from the component agent interaction management. Epistemic information from maintenance of world information, epistemic world info, is transferred to input belief info on world of the components world interaction management, agent interaction management and own process control, through the information links world info to wim, world info to aim and world info to opc.

Comparably the component maintenance of agent information receives meta-information on communicated information from the component agent interaction management, through the information link communicated agent info and meta-information on observed agent information (through the link observed agent info) from the component world interaction management. Epistemic information, epistemic agent info, is output of the component maintenance of agent information, becomes input belief info on agents of the components world interaction management, agent interaction management and own process control, through the information links agent info to wim, agent info to aim and agent info to opc.

4 A Generic Model of Design

The generic model of a design agent is based on both the generic agent model discussed in Section 3, and a generic model of the design task, used to model the agent specific task component. In this section the structure of this agent specific task component for a design agent is described.

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 introduced in (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 applications, including elevator configuration (Brazier, Langen, Treur, Wijngaards and Willems, 1996) and design of environmental measures (Brazier, Treur and Wijngaards, 1996). The model not only provides an abstract description of a design process comparable to a *design model*, e.g., (Coyne, Rosenman, Radford, Balachandran and Gero, 1990; Smithers, 1994), but also a generic structure which can be refined for specific design tasks in different domains of application. Refinement of the generic task model of design, by specialisation and 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 set of requirements. This set 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 sets of requirements may change as may the design object descriptions: they evolve during design. The strategy employed for the co-ordination 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).

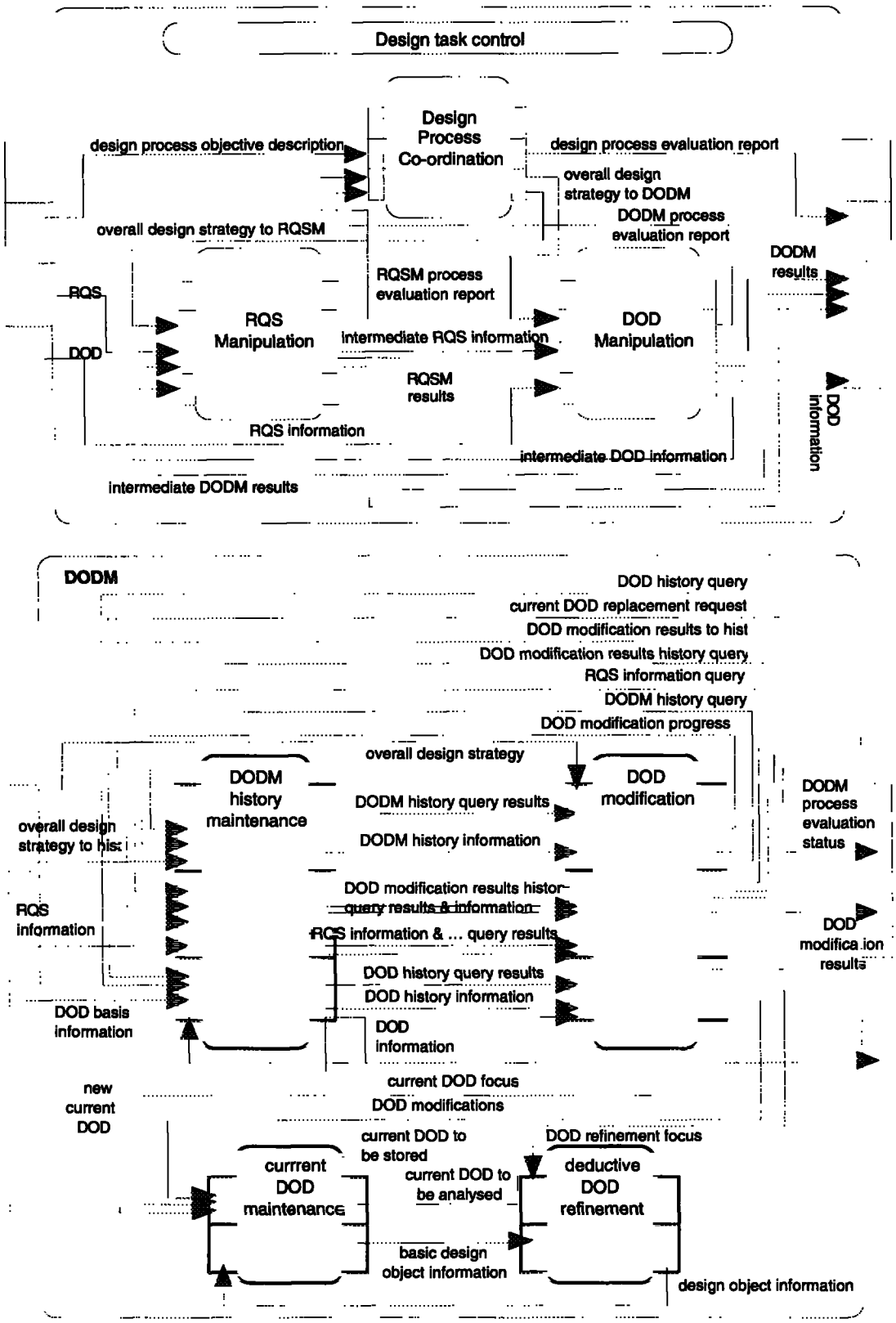

Figure 2. A generic model of design: two process abstraction levels.

Figure 2 shows two levels of composition of the generic model for design. Three processes are shown at the top level, together with the information exchange. Four processes and information exchange are shown at the second level for DODM.

The four processes (see Figure 2) related to the process *requirement qualification set manipulation (RQSM)* 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 (DODM)* 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 process *design process co-ordination* is composed in a similar manner.

5 Specialisation of the Design Agent Model

The generic model for design described in the previous chapter can be refined by specialisation and instantiation. This section addresses a specialisation of the generic model of design which has been applied in the domain of re-design of compositional (knowledge-based or multi-agent) systems. For reasons of space limitation only the specialisation of the process of requirement qualification set modification and the process of design object description modification are presented.

5.1 Specialisation of requirement qualification set modification

The process RQS modification determines modifications to a requirement qualification set (RQS). To this purpose a number of sub-processes are distinguished as shown in Figure 3. The process RQS modification process co-ordination is responsible for the co-ordination of the entire process within RQSM: this process determines whether, when and by which means a specific RQS is to be modified.

The global phases within RQS modification resemble a process control model (e.g., controlling a chemical process). A process control task usually relies on a feedback loop within which sub-tasks such as analysis, planning, and execution are distinguished.

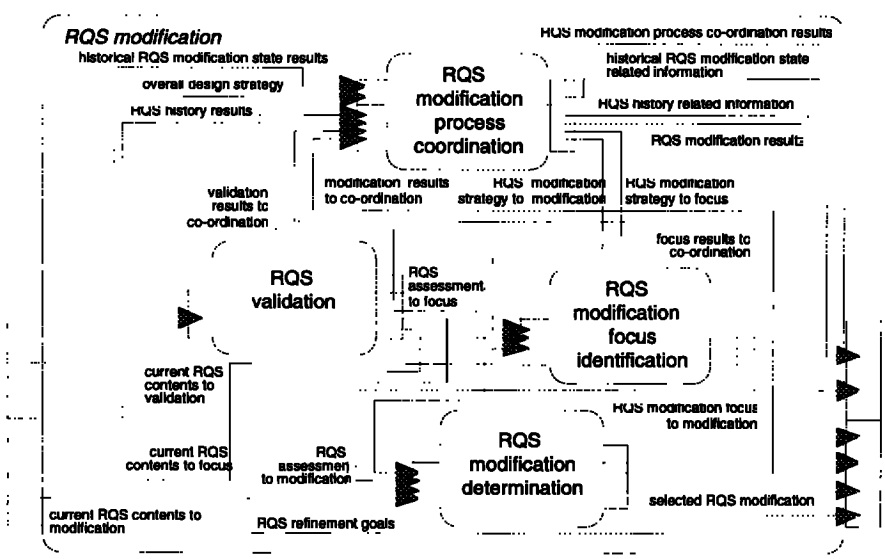

Figure 3. Specialisation of RQS modification.

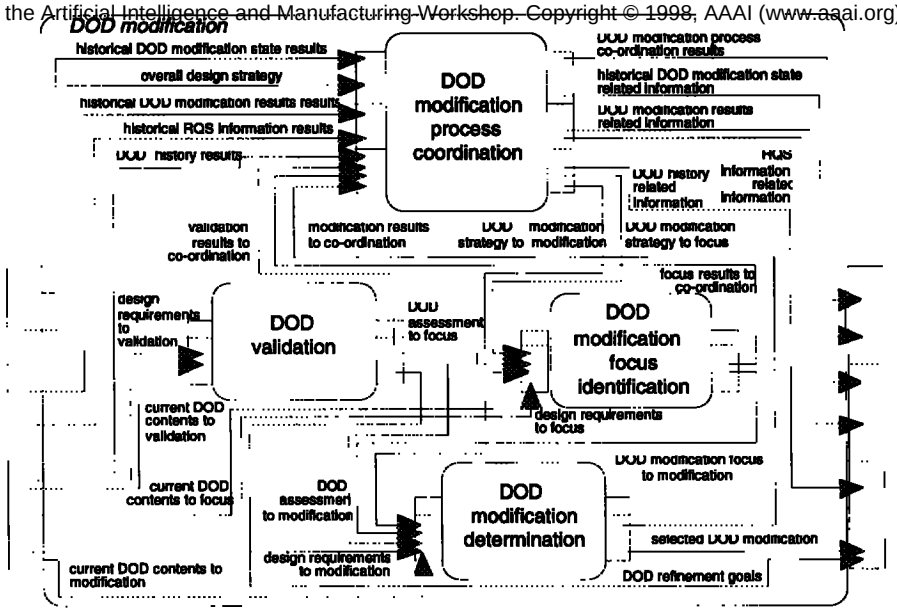

Figure 4. Specialisation of DOD modification.

Similarly, within RQS modification analysis is performed by RQS validation, planning is performed by RQS modification focus identification and RQS modification determination, and execution is performed by effectuating modifications to a RQS, resulting in a new RQS in current RQS maintenance.

5.2 Specialisation of design object description modification

The process DOD modification determines modifications to a design object description (DOD) in such a fashion that a DOD is constructed which adheres to the design requirements given to DODM. To this purpose a number of sub-processes are distinguished as shown in Figure 4. The process DOD modification process co-ordination is responsible for the co-ordination of the entire process within DODM: this process determines whether, when and by which means a particular DOD is to be modified.

The global phases within DOD modification also resemble a process control model: analysis, planning, execution. Similarly, within DOD modification analysis is performed by DOD validation (including assessing requirements in the current DOD), planning is performed by DOD modification focus identification and DOD modification determination, and execution is performed by effectuating modifications to a DOD, resulting in a new DOD in current DOD maintenance.

6 Application: a Design Agent for Agent Design

The generic model of a design agent introduced can, in principle, be used for any domain of application. To obtain a proof of concept it was applied to a specific domain within compositional system design: the domain of compositional agent design. For this domain a prototype application has been designed and implemented. For this prototype system a formal ontology of requirements on agents has been developed. Moreover, knowledge has been identified that can be used to reason about these requirements, to derive more specific requirements by refining the original requirements. These more specific requirements play a crucial role in the design process: they guide the direction in which solutions are sought.

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.

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 5 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).

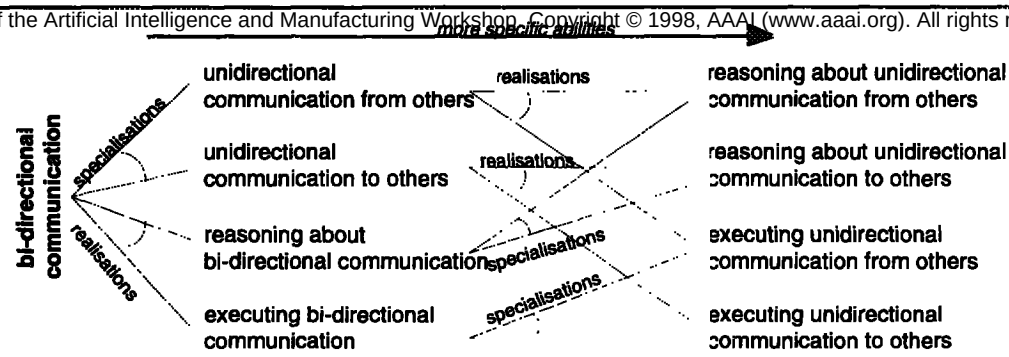

Figure 5. Refinements of the ability of bi-directional communication.

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 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 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 can be 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" ) ) )
```

Continued on the next page.

```
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 6 shows an example of (part of) such a requirements refinement hierarchy. The current prototype design 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 6. Requirement refinement hierarchy constructed by the prototype design agent.

The implication of designing (parts of) a multi-agent system, is that a multi-agent system is the object of design, and as such should be formally represented in a design object description. In this paper the design object description is assumed to be a compositional object description. 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.

7 Discussion

To design an agent capable of designing, insight is required in the agent model and the design process model to be used. The architecture of the design agent 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.

The generic agent model has been developed on the basis of experience with 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 generic design agent, which combines the generic agent model and the generic design model, was successfully applied to compositional agent design. The design agent model for this application includes formalisations of agent design descriptions and requirements on 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.

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.

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.

A principled approach to the design of the architecture is of crucial importance: a generic conceptual architecture of a broker 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.

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.