Annot Agents: Knowledge-Based Tools for Supporting Participatory Design

D Vervenne
S Rogge
W Van Laere
F Vandamme

BIKIT, University Gent
Plateaustraat 22
B-9000 Gent
BELGIUM

In this paper, we consider participatory design as aiming for full *cooperation of all actors involved in collaborative design processes. This type of co-design needs efficient tools to support the complex interactions on many levels. The paper describes our AnnotAgents which is a method and a library of tools which we are currently developing to support these complex cooperation processes.*

1 Introduction

Design tasks require flexible communication interaction between different partners who are dealing in one or another way with all live-cycle phases of a design. Whether the design is a new house, a landscape or a new steel alloy, the fast and accurate exchange of relevant information between all partners is therefor an important and crucial issue. The basis of communication during design phases becomes more and more digitised media: formal and informal documents, CAD-drawings, spreadsheets and simulation environments or virtual reality testfields.

Each industrial design is a very complex process in that it must consider a lot of constraints, desires and requirements, e.g. cost, functionalitys, legal rules, material, expertise, assembly methods, production constraints,... These requirements are not always explicit present and can even be contradictory. This makes it very hard for the operator/enduser to deal with it, the more because s/he has a high degree of freedom in designing (bridges and constructions can be build in many different was).

However neglecting one or more constraints can have a wide- range of disastrous consequences: unexpected costs, impossibility to manufacture, legal disapproval, collapsing constructions,. ..Codesign-supporting systems should thus enable customers, endusers, production engineers, budget planners,.. to have an impact on the design process by allowing them to input their requirements, desires and constraints into the system. Some of them will be brought in on a permanent basis as they are valid for several designs. Others will be specifications for a specific design: in that case they need to be entered through a kind of electronic orderbook. In general, the basic characteristic of a computer-based codesign platform is flexible communication between all actors dealing with the various phases of a design process. In this paper, we will focus on informal communication aspects of co-design; this kind of communications is supported through semi-structured annotations.

2 Components for communication in co-design platforms

Communication between actors which are involved in co-design processes, has recently been supported by many types of technology such as expert systems, critiquing modules, intelligent agents and user modelling features.

The use of explicit knowledge, usually represented as conditional rules, was original successful implemented for analyses tasks such as diagnosis and classification. In the second generation expert systems, researchers started to apply those techniques also on -more complex- synthesis tasks such as planning and scheduling, and design. This requires the system to search a solution fulfilling the requirements, represented in the form of constraints. Common in all those approaches in the engineering design domain, is the assumption that the design is decomposable in a hierarchic way. From there it is assumed that the designer is also building this design according the hierarchic object structure. In contrast to this approach is the more interactive critiquing-modules by Fischer (see Fischer 1989, Lemke 1990 and Fischer 1991).Intelligent agent technology is a new interpretation of expert systems which where introduced as a complementary style of user-interaction. In stead of user-initiated interactions, the user/designer is engaged in a cooperative process in which human and computer agents both initiate communication, monitor events and perform tasks. The metaphor used is that of a personal assistant who is co-operating and collaborating with the user in the same work environment. Such an assistant becomes gradually more effective as it learns the users interests, habits and preferences. In general, agents can assist in a range of different ways: they hide the complexity of difficult tasks or they perform tasks on the users behalf, or they can train or teach the user or monitor complex events.

Current position of this technology is illustrated by, e.g. Maes P, 1994 . She mentions two main problems to be solved in building intelligent agents:

(1) how does an intelligent agent acquires its competence?

(2) how can the enduser trust the agents so that s/he can delegate important tasks?

Both problems are currently solved by rule-based approaches whereby domainspecific knowledge about the application (a domain model) is combined with hypothetical knowledge of the user (a user model). A good illustration is UCEgo from Chin, 1992. This system helps Unix-users in solving their unix-problems; the system has a large domainmodel about the unix-universe and incorporates hypotheses about user-goals and (meta)planning.

There are many problems with this current approach (the rule-based approach):

(a) it requires a huge amount of work by the knowledge engineer in order to capture the relevant rules and

(b) the agent knowledge is fixed and static and is difficult to customise and

(c) the enduser has difficulties in trusting such rule-based module since s/he doesn't want to accept the limited explanation capacities of such an expert system approach.

The solution of Maes F, 1994 to the above mentioned problems is by introducing machine learning technology: the hypothesis is that an interface agent can program itself by learning from the user and from other agents. This approach looks attractive but the author mentions the necessary conditions which have to be fulfilled:

(a) the application has to involve a substantial amount of repetitive behaviour and

(b) the repetitive behaviour is potentially different for different users.

This means that, if no regularities are detected by the learning module, then the agents can not program themselves. Such a hypothesis is probably valid in the field of information filtering and/or news-routing applications such as mentioned by Maes, 1994 but for design-oriented projects, this hypothesis is surely not fulfilled since design-tasks are rarely routine-based. In our solution, we introduce the interaction with the enduserdesigner through annotation processing features which supports the user in modifying and controlling the system towards his/her changing requirements.

3 Annotagents: Agents for annotation processing

Our approach is based on AnnotAgents technology, an integration of agentstechnology, user modelling and annotation processing. During participatory design Processes, each actor involved in a subprocess can add different types of annotations on designed artefact. The management and processing of such annotations are crucial for he the adequate exchange of informal communication between all actors engaged in the design process.

3. 1 Annotation-typology

Annotations are individual comments of endusers which they produce on the spot while working with their system: such comments can be expressed at each moment of a consultation and are stored in a separate annotationbase. Annotations are very valuable for adding semi-structered knowledge from various enduser types. The annotation fields are open for reading and writing from different types of users. In order to prevent that existing commentaries are overwritten of deleted by not authorised persons, a certain 'login-procedure' will be used. This procedure is responsible for the maintenance of information concerning the author of the annotation, the date, etc. These different items can be used as a kind of 'search/query' possibility for filtering and purposeful querying annotations.

The annotation management system AMS is a management module with an annotation-base and a set of annotation-agents to support the interaction and the processing with the endusers. We provide three types of three kinds of annotations which users can attach to all knowledge nodes and/or their interrelations contained in the codesign system: "Free-text" annotation, "fill-in form" annotation and "question-answering" annotation is provided. The enduser can choose between those three possibilities; s/he will experience that the degree of freedom to annotate is correlated with the processing power, e.g., to differentiate 'new' from 'old' knowledge.

From practical experience, we learned that the choice by the enduser is correlated with the learning curve through which the enduser crosses; a novice user will prefer a question-answer annotation style because the interaction there is steered by the system. The expert user will jump quickly to other annotation styles because s/he will take into account the speed by which new knowledge in the system will be processed.

Free text annotations: For "free text" annotations, some of the most common textprocessing possibilities are provided so that the enduser has no difficulties to add comments. In such a "text-window" the enduser can add his free commentaries on each existing knowledge node or to an existing annotation. Endusers can call their favourite wordprocessor and import existing reports or cut and paste relevant paragraphs into a text-annotation window.

For each annotation there is an option to fill in some keywords which refer to knowledge quality parameters that are used inside the co-design system in order to store the comments in the annotation base of AMS. A limited 'AutoLink'-function can here optionally be provided so that the author of the annotations can elaborate a first rough processing, based on individual user-thesauri. Such thesauri represent keywords that are relevant for a given enduser. Parsing free-text annotations can proceed on the basis of matching the annotation with all provided user-thesauri.

Fill-in form annotations: When the user selects a "fill-in form" type of annotations s/he then gets a fill-in form on the screen on which certain appropriate fixed fields are provided. The enduser does not have to fill in all fields and for some fields a few answers are provided from which s/he can choose. For example, an annotation can be filled in that expresses a new relation between two existing knowledge nodes: the fill-in form will then show on the screen a list of these existing nodes from which the user can select one.

When a relation is annotated to a new node, a new node will be added in the AMS-module, while the system controls on existing nodes and their synonyms. Fill-in form annotations will be interpreted by a forward-reasoning inference engine (parsing-agent) which stores the inferred results.

Question-answering annotations: When the enduser selects the "questionanswering" type of annotations, then the Annotation-Management-module starts a dialogue session. After each question a list of possible answers or an empty fill-in field is provided to the enduser. The dialogue is steered by a backward-reasoning inference engine, the QA-agent, which will do some pre-processing that has to be reported to the AMS-module

Novice users will probably prefer this question-answering type of annotations since the process is in total control of the system. On the other hand, expert users like to use free-text annotations since they have total control on the dialogue and can add and delete every type of verbal comment.

3.2 Annotation processing through AnnotAgents

The annotation-processing will be supported by knowledge-based agents. According to Foner, 1995, such agents have the following characteristics:

(1) Autonomy: Any agent should have a measure of autonomy from its user. This requires aspects of periodic action, spontaneous execution, and initiative, in that the agent must be able to take pre-emptive or independent actions that will eventually benefit the user. Personalizability: agents should enable people to do some task better. Ideally, there should be components of learning so the user does not necessarily have to program the agent explicitly; Discourse: For all tasks, we generally need to be assured that the agent shares our agenda and can carry out the task the way we want it done. This generally requires a discourse with the agent, a two-way feedback, in which both parties make their intentions and abilities known, and mutually agree on something resembling a contract about what is to be done, and by whom. Risk and trust: The idea of an agent is intimately tied up with the notion of delegation. We cannot delegate a task to someone or something else if we do not have at least a reasonable assurance that the entity to which we delegated can carry out the task we wanted, to our specifications. We have to balance the risk that the agent will do something wrong with the trust that it will do it right.

(2) Graceful degradation: Bound up in the notions of risk and trust, agents work best when they exhibit graceful degradation in cases of a communications mismatch (the two parties do not necessarily communicate well, and may not realise it) or a domain mismatch (one or both parties are simply out of their element, and again may not realise it).

(3) Cooperation: The user and the agent are essentially collaborating in constructing a contract. The user is specifying what actions should be performed on his or her behalf, and the agent is specifying what it can do and providing results. This is often best viewed as a two-way conversation, in which each party may ask questions of the other to verify that both sides are in agreement about what is going on.

(4) Expectations: Whenever one interacts with some other entity, whether that entity is human or cybernetic, the interaction goes better if one's expectations match reality. Agents are most useful in domains in which graceful degradation and the correct balance of risk to trust can be obtained, and users' expectations are very important in establishing this domain and making the agent useful.

Foners list of agents' basic characteristics resembles many definitions of artificial intelligence in general. In our approach, we stress the features of autonomy, personalizability and cooperation. These three aspects are crucial to assure a continuous motivation of the endusers to add and read annotations in the co-design environment. This motivation is realised by the idea of processing the contents of the annotations through specialised agents, called AnnotAgents.

We call AnnotAgents all those agents which are specialised in management of all livecycle phases of an annotation:

(a)the authoring of an annotation,

(b) the mailing and routing of annotations,

(c) the parsing and exploitation and finally

(d) the deleting or transformation of annotations.

In the next part we focus on the cooperation capabilities between AnnotAgents and endusers of the co-design platform. In the last part we focus on cooperation among AnnotAgents themselves.

4 Co-design: cooperation among endusers supported by Annotagents

Cooperation during co-design has a strategic importance. Since cooperation can proceed along semi-structured interactive annotation, we need domain specific parsing technology to process, not only the form but also the content of the annotation. If we have a representation of the annotation-content, then various agent-types can start intelligent tasks (routing-agents, alerting-agents, news-filtering, etc.). However, since parsing natural language is a very complex problem, we need to reduce the complexity by introducing domain specific thesauri which are adaptable by the enduser. For this reason, we approach the parsing problem through a user-modelling strategy. First, we describe the thesaurus approach and in the next part, we use this approach for enduser modelling.

4.1 Thesaurus-based parsing of annotations

Thesauri contain a collection of keywords and their respective meanings when used in a certain domain. The meaning of each keyword is represented by various semantic relations to other keywords: a thesaurus can therefore be considered as a type of semantic

network (see Rada et al, 1991). The classic set of semantic relations can be grouped in 3 classes which contain different relation types: syntactic relations (synonyms, plurals, etc.), hierarchic relations (broader term, narrow term, ...) and associative relations (related terms,...). An example is given in Table 1.

THESAURUS CONCEPT	STEEL
syntactic relations	
abbreviations	Fe, Fe-C alloy, Fe3C, ..
translations	Stahl, staal, acier
synonyms	iron, iron-graphite alloy
plurals	steels
hierarchic relations	
broader terms	Alloy, Materials
narrow terms	IF-steels
associative relations	
related terms	ferrite, perlite, austenite, ..
typical phrase	Steel is an alloy with ..

Table 1: example of a thesaurus record set

The kind of relations can be adapted since different domains can require more specific relations. As an example in the field of metallurgy which we introduced for parsing annotations between steel alloy designers (Vervenne & Pattyn, 1993):

METALLURGIC THESAURUS CONCEPT	ATOMIC STRUCTURE
syntactic relations	
has_synonym	Bohr model
has_plural	atomic structures
has_derivation	atomic structuration
hierarchic relations	
is_a	atomic property
has_value	...
associative relations	
is_cause_of	...
is_part_of	atoms
is_subproces_of ...	
is_used_for	...
has_role_of	...

Table 2: example of an adapted thesaurus record set used by steel alloy designers

Parsing annotations of endusers is based on domain specific thesauri. Our parser scans every word of the free text annotation and associates, for each detected keyword found in the domain-thesaurus, its respective semantic related set of keywords. This set, called the annotation relation set Ars, is the input for various knowledge-based agents. A specific agent, called the profile-agent, is responsible for the modelling of each individual enduser. In the next part, we present some details about this profile-agent.

4.2 User modelling through adaptable thesauri

Since each slot-value of a thesaurus is a candidate thesaurus-concept, one can imagine that domain specific thesauri can become very complex. The maintenance of such a semantic network is a hard job and we believe that in the areas of fast evolving knowledge (such as steel alloy design), it is important to consider the involvement of thesaurus endusers in the process of maintenance.

The idea is that each enduser can control his/her own profile-agent which is represented as an adapted subset of the domain thesaurus. The adaptation happens on different levels, according to the access-permission which is defined for each individual enduser:

(a) a_level_1: weighted relations: the enduser can add weights to each slot in his/her thesaurus relation in order to express a degree of importance; table three gives an example.

(b) a_level_2: extended thesaurus values: the enduser can add new keywords to the existing relations. The most pertinent example is, e.g., personal abbreviations which can represent favourite expression or personal dialect. The enduser of this level has also access permission to level _1.

(c) a_level_3: extended thesaurus relations: on this level, the enduser can add new relation types (and respective values) which represent more domain specific knowledge. In the field of steel alloy design, e.g., we soon were confronted with the request of the endusers to add multiple types of causal metallurgic relations to differentiate multiple validation levels: relations which included mathematical formula (e.g. regression formula) have a higher value than causal relations which are qualitatively expressed. The enduser of this level has also access permission to level-2.

Of course, it depends on the hierarchical organisation structure of the department where the AnnotAgents are introduced, which users have access permission to which adaptation level. We are currently doing some fieldwork by visiting companies which use full text search technology with thesauri facilities, in order to study the interpersonal organisation structure. The idea is that cooperation between AnnotAgents could be defined through a library of organisation methods which reflect real company organisations. This idea is explained in the last part of this paper.

METALLURGIC THESAURUS CONCEPT	ATOMIC STRUCTURE	
Bob Andrews	a_level_1	
syntactic relations		
has_synonym	4	Bohr model
has_plural	1	atomic structures
has_derivation	5	atomic structuration
hierarchic relations		
is_a	5	atomic property
has_value	...	
associative relations		
is_cause_of	...	
is_part_of	3	atoms
is_subproces_of ...		
is_used_for	...	
has_role_of	...	

Table 3: example of an weighted thesaurus record set of a_level_1 used by the profile agent of Mr Bob Andrews

5 Co-design: cooperation among Annotagents supported by endusers

User modelling, as interpreted in the previous part, requires input from the enduser so that the profile-agent acquires some relevant knowledge about the individual enduser. According to the classification of Chin, (1993) this is called user-initiated user modelling. Our AnnotAgent approach however also permits automatic user-modelling (also called system-initiated user modelling) since agents can cooperate e.g. by exchanging information about partially identical Profile _agents.

User modelling can be defined as (Wahlster & Kobsa, 1989): developing, storing, maintenance and using an explicit set of assumptions on all aspects of the user that may be relevant to the dialog behaviour of the system, in order to optimise the user-computer interaction. According to this approach, we need to develop user-agents which will cooperate with the profile-agents in order to optimise the user-computer interaction.

User-agents will thus be responsible for the following tasks:

(a) developing a set of assumptions about the individual user.

(b) storing these user assumptions.

(c) detecting new user assumptions and re-evaluating old assumptions.

(d) using the set of assumptions in order to optimise the user-computer interaction.

This leads us to an architecture of an user _agent which is close to the definition of a user modelling module described in Benyon & Murray, (1993). Such a module needs 3 components: the user model, the application model (or domain model) and the interactionmodel (user interface model). According to Benyon & Murray, construction of the usermodelling module will consist of building this three components and letting them work together. Exactly the last requirement of working together is compatible with our agent-approach for a co-design environment.

Our AnnotAgents represent a collection of co-operating agents:

(a) profile _agents describe the domain such as it is observed and modified by each individual enduser; they represent the individually adapted thesauri,

(b) domain-agents represent the subset of all those thesaurus-nodes which all profile _agents have in common: the result is a departmental knowledgebase which in many cases contains very valuable company know-how (cf the notion of company repository),

(c) user-agents will observe the behaviour of the enduser and construct user_assumptions. Before storing these assumptions, the respective user will be asked for adding confirmation weights. User_agents can, e.g., observe endusers in the way they use the three annotation types, mentioned in 2.1, (to whom endusers are sending their annotations and what annotation style they are using)

(d) parsing_agents will scan the content of an annotation, searching for keywords which can be linked to associated keywords from the thesaurus. The output of the parser depends on the contents of the thesaurus; if e.g. causal relations are representing qualitative domain knowledge, the parsing_agents can identify new relations if a causal pattern is detected for which the antecedent or the consequent-part are new.

(e) routing-agents will support the various mailing phases of annotations.

Based on the output of the parsing-agents, the routing can be based e.g. on detected keywords which have a high weighting factor in those individual thesauri of potential interested endusers. The routing-agent will then suggest to the annotation-author to mail the annotation to that enduser who will be asked for validating the relevance of the mailed annotation.

6 Conclusion

In this paper, we focused on informal cooperation of actors which are involved in complex design processes. The informal cooperation is based on various types of annotations that are created by the endusers/designers. We defined a family of agents who support all phases of the annotation live cycle: we called them AnnotAgents. The power of the AnnotAgents is based on domain-oriented thesauri which can be modified by the endusers. Cooperation is thus not only viewed as a relation between endusers, but also as a communication process between agents and endusers and amongst agents themselves.

Future developments will be oriented on testing of the current prototypes in Lotus Notes and Wordwide Web environments.

7 References

(1) Benyon D., & Murray D., 1993, Applying user modelling to human-computer interaction design in Artificial Intelligence Review vol 7 nos. 3-4

(2) Chin, D. N. 1992. Intelligent Inferfaces as Agents. In Sullivan, J. and Tyler, S. (Eds.), Intelligent User Interfaces, ACM Press, New York, pp.17?- 206

(3) Chin D., 1993, Acquiring user models in Artificial Intelligence Review vol 7, nos 3, 4

(4) Fischer G., & Girgensohn A., 1989 End-user modifiability in design environments, Report at Univ; of Colorado, Boulder

(5) Fischer G., Grudin J., Lemke A., McCall R., Ostwald J., & Shipman F., 1991, Supporting asynchronous collaborative design with integrated knowledge-based design environments, Univ; of Colorado at Boulder

(6) Fischer G., Lemke A., & Rathke C., 1987, From design to redesign Proceedings of 9th Intern. Conf. on Software Engineering

(7) Fisher G., & Nakakoji, 1991, Empowering designers with integrated design environments, Proceedings of Al in Design, Edinburgh.

(8) L. Foner 1995, MIT, agent research notes on http://foner.www.media.mit.edu/people/foner/

(9) Gosling J., 1983, Algebraic constraints PhD Thesis CM Univ

(10) Kobsa A., (1989) A taxonomy of beliefs and goals for user models in dialog systems

(11) Maes P., 1994, Agents that reduce work and information overload, in Communications of ACM, jul

(12) Rada R., 1991, Hypertext; from text to expertext

(13) Vervenne D.1(1994a) "Cooperative maintenance of expertext through structured annotations', in CD-Rom Poceedings of the 2nd World Congress on Expert Systems, Estoril Portugal.

(14) Vervenne D., (1994b) Interactive annotation processing for extraction of strategic knowledge, Conference Proceedings of Management of Industrial and Corporate Knowledge, 26-27 October, Compiegne, France

(15) Vervenne D., & Fransen F., (1992) "SPIK, a knowledge-based system to support steel-alloy design", Proceedings of the Twelfth International Conference on Expert Systems in AVIGNON '92, June 1-6., Specialised conference on Materials.

(16) Vervenne D., & Pattyn K., (1993) "Maintenance of knowledge-based systems trough structured annotation by the endusers", in Proceedings of the 4th Workshop on Artificial Intelligence and Knowledge-based Systems for Space, may 17-19th, ESA - ESTEC, The Netherlands.

(17) Wahlster & Kobsa, 1989 User models in dialogue systems in Kobsa & Wahlster (eds) User models in dialogue systems, Spinger Verlag London.