INTERNATIONAL CONFERENCE ON ENGINEERING DESIGN
ICED 03 STOCKHOLM, AUGUST 19-21, 2003

TOWARDS A FRAMEWORK FOR AGENT-BASED PRODUCT MODELLING

John S. Gero and Udo Kannengiesser

Abstract

This paper presents an approach to product modelling that uses computational agents to represent product data. These agents are situated, i.e. they produce all data representations in response to the specific need in the current situation. We have explored this characteristic in the context of the communication of product data to develop a framework for agent-based product modelling. This framework complements current standardisation efforts through its ability to construct product models on the fly to flexibly adapt to changes in the environment.

Keywords: product modelling, design representations, product data handling

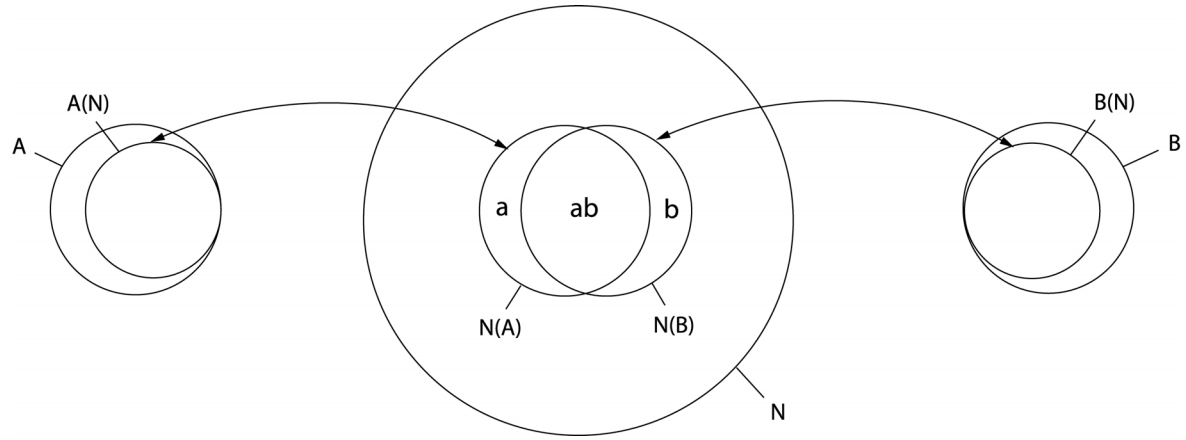
1. Introduction

There has been an increased use of computational tools to support various tasks in product development. Examples include computer-aided drafting (CAD) and manufacturing (CAM) systems and a number of specialised tools for analyses such as finite element analysis (FEA) and spreadsheet analysis. These processes require fundamentally different data about the product and different representations of that data, even though the data is concerned with the same product. Most computational systems supporting these processes have been developed independently from one another to address the specific needs of each process and use different product data representations. However, industrial product development is a process that involves a complex network of interrelated activities, each of which needs information produced or manipulated by the other. Interoperability – the ability to move data from one representation of a product to another to allow other computational processes to operate on it – has become an area of growing concern as the cost of such interchanges increases [1].

Product modelling [2] includes the notion that describes conceptual approaches to the issue of exchanging product data representations within and between companies. These approaches are commonly founded on a standard data model that is used to translate between the different native formats of the applications. Any object that needs to be made interoperable must be pre-defined in this model and encoded into a standard form. One of the best-known product models is the ISO 10303 standard, informally known as STEP (STandard for the Exchange of Product model data), developed by the International Standards Organization (ISO). The STEP product model consists of a number of partial models called Application Protocols (APs) describing classes of objects for specialised domains. All STEP APs are represented using the EXPRESS language [3]. Another well-known product modelling example is the International Alliance for Interoperability (IAI), an industry-led organization whose members have agreed upon a pre-

defined product data schema called Industry Foundation Classes (IFCs) [4], which are also represented in the EXPRESS format.

Although both STEP and IFCs have seen an increasing use in industry, there are still some unresolved practical issues. Many applications have specialised modelling capabilities that are not supported by existing STEP APs or IFCs. One of the reasons is that the development and extension of these standard models is a relatively slow procedure involving international consensus-seeking, which lags behind the advances in modelling technologies implemented in industry. In addition, many translators have been built to exchange specialised data between specialised tools. As a consequence, they are tailored to implement only a subset of the neutral model that would be capable of capturing the whole tool data. When data transfer is required between new combinations of the translators/tools, their data representations in the neutral model often mismatch leading to data loss. Figure 1 shows some inadequacies in mapping the native product models of the tools (A and B) into a neutral model as well as the capability mismatch of the translators. Interoperability among the tools is determined solely by the intersection (ab) that is implemented by both translators, while the subsets (a and b) that are supported by only one of the translators are ignored by the other.



A: product model of tool A
A(N): interoperable subset of A
↔: mapping
N: neutral product model
N(A): subset of N, translatable for tool A
N(B): subset of N, translatable for tool B
B: product model of tool B
B(N): interoperable subset of B

Figure 1. The mapping inadequacies and capability mismatches when using a neutral product model.

The common approach to solve these problems has been to accelerate the development of new standard specifications and to increase the capabilities of translators; however the success of this approach has been limited. A major reason for this is the dynamics of the industrial environment leading to new or modified applications or application domains to be integrated into product development. Extending product models and re-writing translators is time-consuming and costly and can finally result in increasing amounts of data to be handled.

Underlying most product modelling approaches is the assumption of a world of pre-existing components. However, the generation and manipulation of product data constantly redefines product models as new design concepts, new technologies and new tools are to be integrated.

This paper presents research towards a framework that extends current standardisation efforts by using computational situated agents to represent product models. This approach aims at achieving interoperability when STEP APs or IFCs are not readily available. Our assumptions for developing this framework draw from existing work in cognitive science and research in design agents.

2. Situated communicating agents

2.1 Situatedness

A characteristic that allows a system or agent to adapt its behaviour to changes in its environment is situatedness. The foundational concepts of situatedness go back to the work of Dewey [5] and Bartlett [6]. It is based on the view that "where you are when you do what you do matters", i.e. the agent does not simply react reflexively in its environment but uses its interpretation of its current environment and its knowledge to produce a response [7]. Situatedness has a particular explanatory power in the area of design research, as designing has been recognised as an activity that changes the world in which it takes place [8]. Experimental studies [9, 10] have characterised designing as an interaction of the designer with their environment: after changing the environment (e.g. by means of sketching), the design agent observes the resulting changes (i.e. the sketch) and then decides on new (sketching) actions to be executed on the environment. This means that the agent's concepts may change according to what it is "seeing", which itself is a function of what it has done. As a consequence the agent can be exposed to different environments and produce appropriate responses. A framework for situated cognition in a design agent has been developed by Gero and Fujii [11]. Based on this work, a number of design agents that embody situatedness have recently been implemented [12, 13].

If we view the notion of the environment as including both the agent's external and internal environment, then situatedness is extended to include a concept known as constructive memory. It is best exemplified by a quote from Dewey via Clancey [7]: "Sequences of acts are composed such that subsequent experiences categorize and hence give meaning to what was experienced before". The implication of this is that memory is not laid down and fixed at the time of the original sensate experience but is a function of what comes later as well. Memories can therefore be viewed as being constructed in response to a specific demand, based on the original experience as well as the situation pertaining at the time of the demand for this memory. Each memory, after it has been constructed, is added to the agent's knowledge and is now available to be used later, when new demands require the construction of further memories. These new memories can be viewed as new interpretations of the agent's augmented knowledge.

We can view situatedness as the ability of an agent to construct external as well as internal representations as a function of the current situation, which is the agent's interpreted (external and internal) environment. In other words, a situated agent can be exposed to different environments (external as well as internal to the agent) and produce appropriate responses. The agent's knowledge is thus grounded in its interactions with the environment rather than pre-defined and encoded by a third party.

To exploit the adaptiveness of situated agents for the purpose of generating data representations as an input to other computational systems, we have to explore a further concept from cognitive science called common ground, which places situatedness into the context of communication.

2.2 Common ground

For successful communication it is essential that the exchanged data representation is consistent with a shared context of the agents, comprised of shared concepts and shared language. While research in computational multi-agent systems (MAS) has generally tried to implement common ontologies [14] using pre-defined schemas that are then encoded into interoperable agents, cognitive science has kept a situated stance building on the notion of common ground. Common ground has been described as the set of presuppositions "any rational participant [in a conversation] is rationally justified in taking for granted, for example, by virtue of what has been said in the conversation up to that point, what all the participants are in a position to perceive as true, whatever else they mutually know, assume, etc." [15]. Two essential ideas distinguish common ground from the current understanding and use of common ontologies in MAS. First, common ground is based on an agent's first-person constructions, thus is grounded in the interactions with other agents rather than externally encoded and static. This conforms to our situated view of communication. Second, common ground depends on which other agents are involved in the communication. As a consequence, there is different common ground between different pairs of agents, which is used by an agent to adapt its message generation as well as comprehension specifically to the other agent [16, 17].

It has been suggested that for an agent to evaluate the entire common ground it shares with another agent it would have to go through a process of infinite recursion [18]: It would have to construct a belief about the other agent's belief about the common ground, which itself entails constructing a belief about the agent's belief about the common ground, and so on. However, situated agents avoid this recursion by "satisficing" [19] common ground according to the needs of the current interaction only. The key for an agent to establish sufficient common ground is the construction of an appropriate mental model of the other agent, which is then used to adapt data generation and interpretation to that agent. This model is constructed through the agent's previous experience with that agent as well as the agent's generalised experience from interactions with other agents. The present authors [20] have recently proposed a formal representation schema for agents based on function (F), behaviour (B) and structure (S):

- F: describes the role of the agent for the observer ("what the agent is there for").
- B: describes attributes derived or expected to be derived from the agent's structure (S), which includes how the agent acts under specified conditions ("how the agent fulfils its function", from a "black-box" stance).
- S: describes the agent's components and their relationships, which includes the agent's capabilities and knowledge ("what the agent consists of").

Take the example of an agent whose function (F) is to design buildings. This function can be fulfilled by its behaviour (B) of producing appropriate design descriptions of buildings in response to certain requirements. The parts that the observer might know about the agent's structure (S) can include the agent's knowledge domains, language and sensor capabilities.

The FBS view provides a comprehensive set of constructs to model all kinds of “agents”, including simple objects or software applications. This allows the agent to treat all aspects of the world uniformly even when they appear at different levels in a conceptual hierarchy. As a consequence, the agent is able to construct generalisations from a set of individual experiences, which significantly helps to reduce complexity inside the agent. Generalised knowledge is particularly useful for communicating agents that have only little information about each other, as it readily complements their FBS models by providing additional default assumptions.

Figure 2 shows how an agent (0) has constructed function-behaviour-structure (FBS) models of other agents (1, 2, 3 and 4), nested in the FBS model of agent 0 itself. As the differently sized FBS models in the figure suggest, some agents (1 and 2) are better known than others (3 and 4), and the best-known agent is certainly agent 0 itself. When the agent wants to communicate with an agent but has too little knowledge about that agent (here agent 4) to establish sufficient common ground, it complements the existing FBS model with assumptions reflecting its generalised knowledge about agents. This generalised knowledge is derived mainly from those instances the agent (0) is most familiar with (as indicated by the different weights of the arrows in Figure 2), which principally includes the agent (0) itself. When a new, previously unknown agent (5) enters agent 0’s environment, the generalised knowledge may in most cases still suffice to construct an adequate FBS model of that agent.

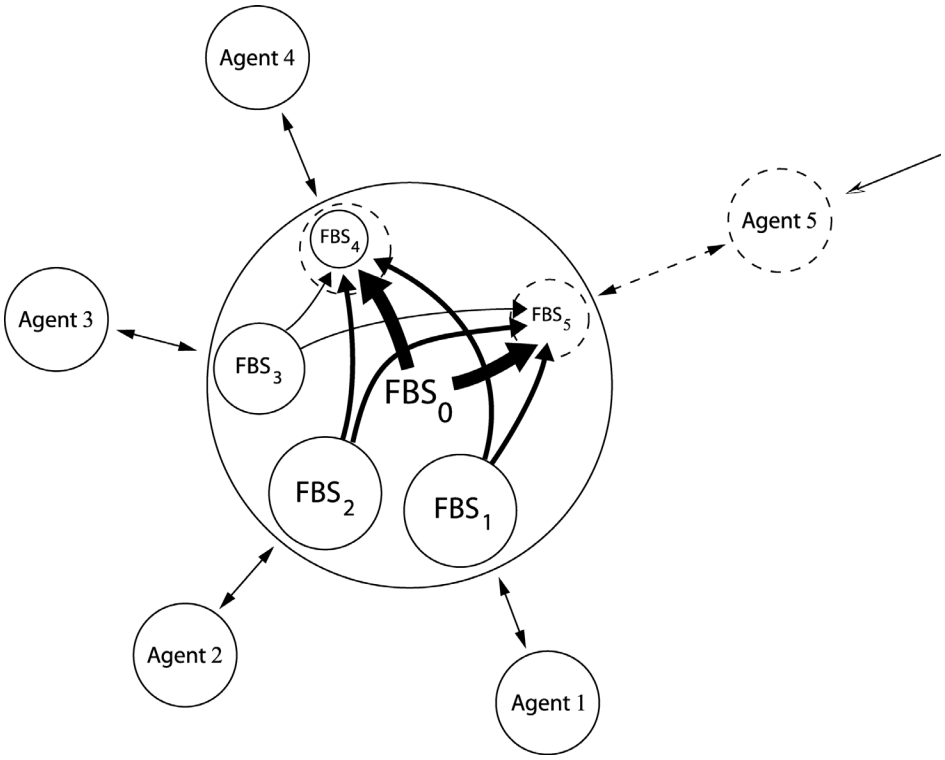


Figure 2. New agent models are constructed using previously constructed agent models.

3. Towards an agent-based system for product modelling

3.1 Product data exchange using situated communication

The ability of a situated agent to adapt its communication to other agents' capabilities has the potential to address some of the issues in product data exchange. Figure 3 illustrates how an agent can construct different product data based on its FBS knowledge about the particular tools (viewed as agents) that need the data. Specifically, the product models are adapted with respect to important aspects of the tools' structure (S), such as the concepts and the formats understood by the tools. For example, the model provided to CAD System 1 describes two-dimensional geometrical concepts in DXF format, whereas the one provided to CAD System 2 is a STEP

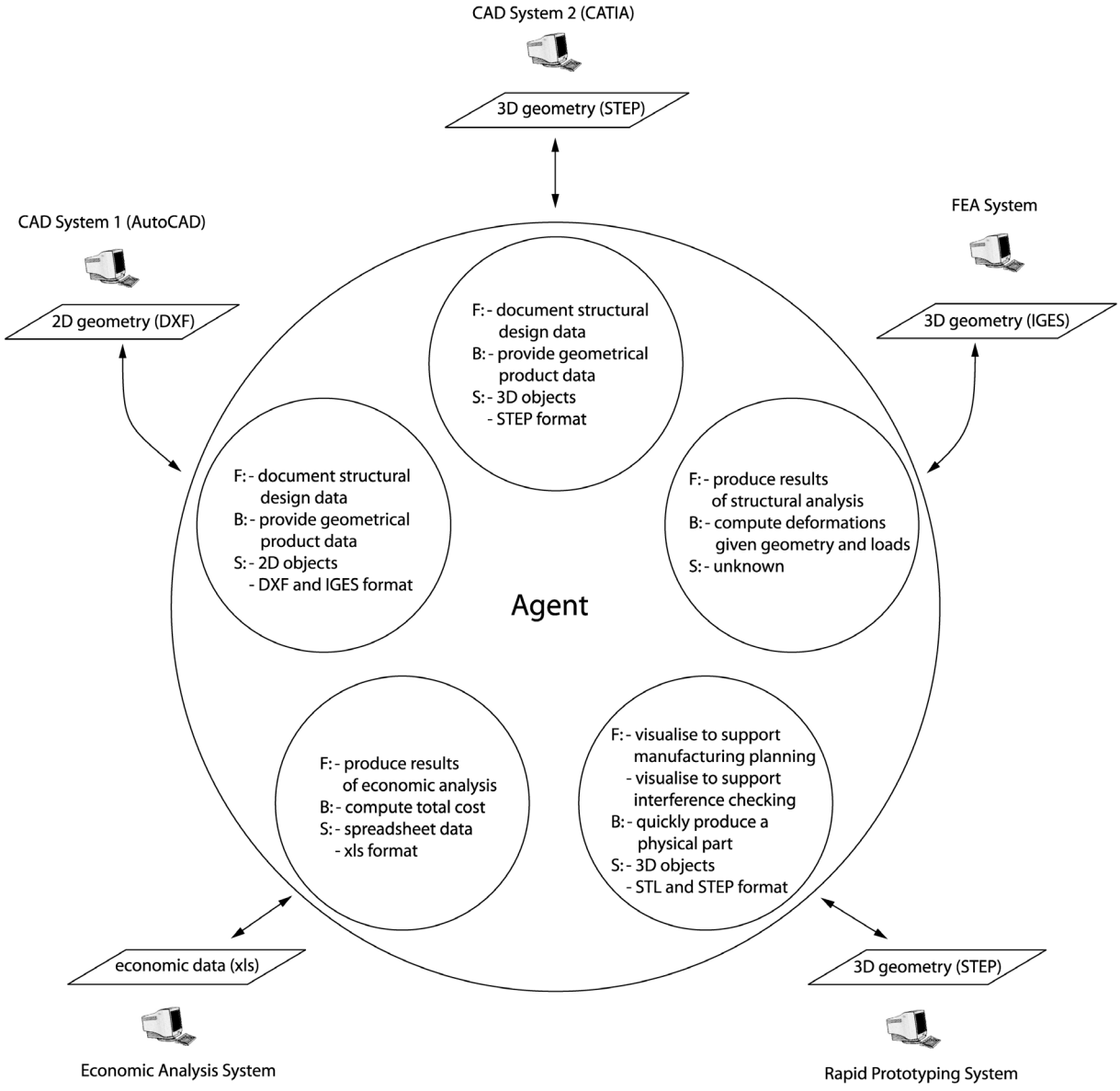


Figure 3. The agent adapts the data concepts and formats to its knowledge about the tools.

representation of 3D geometry. The agent can also account for data concepts that no longer refer to geometry, such as the spreadsheet data required by the economic analysis system.

Most importantly, it can be seen in Figure 3 how the agent-based system can be used to establish data exchange when the main standard model (e.g. STEP or IFCs) is not available. For example, if the agent has read STEP data from one tool (CAD system 2) to be passed on to another tool (CAD system 1) that does not have a STEP translator, the agent can (translate and) represent the data in a format that this tool understands (here DXF). This quick and flexible adaptation to the given circumstances can avoid the immediate need for new translators, which often arises when the environment is modified by adding or substituting tools. The agent not even has to know all of the properties of a new tool. Take the example of the FEA system: Its structure (S) in terms of data concepts and formats are firstly unknown; however, the agent has some amount of other, more general information about that tool, referring to the typical function (F) and behaviour (B) of an FEA system. This knowledge can then be used to derive the needed information about the tool's structure (S) from generalisations of previous experiences with similar kinds of tools. In our example, the agent infers that both the use of 3D geometrical data and their representation in the IGES format are likely to be meaningful for the new FEA tool.

3.2 Possible system architectures

The product modelling approach as outlined so far has shown that the data translations from one tool to another are no longer carried out by fixed translators but by one situated agent. The single-agent approach, however, is only one out of several possible agent configurations. Below we present three different system architectures based on the number of agents and their relative autonomy.

1. One agent modelling the whole product:

   Figure 4 depicts an agent that is specialised to know everything about a particular type of product (here a transmission). The agent (called the "transmission agent") communicates all aspects of the product to a tool. A problem of this architecture is likely to be the amounts of data that this single agent has to manage and translate from one format to another.

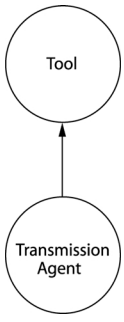


Figure 4. One agent modelling the whole product.

2. Emerging the product model by hierarchically structured agents:

   Figure 5 shows how agents are specialised to represent the product on several levels of aggregation. Every agent constructs representations for the agent on the next-higher level in the hierarchy, thus constructing the product model from the components via sub-

assemblies to the whole product. This bottom-up architecture has all the advantages of distributed, robust computation, while preserving coherence through a controlled, step-by-step "assembly" of the product.

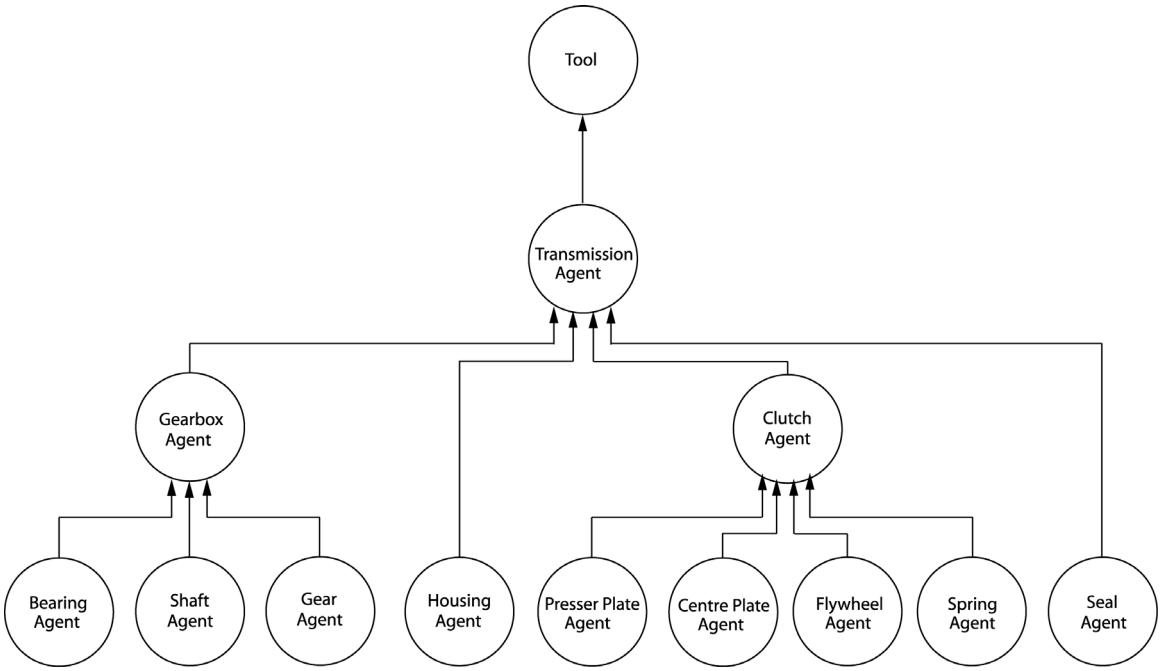


Figure 5. Emerging the product model by hierarchically structured agents.

3. Emerging the product model by self-organising agents:

   Figure 6 shows a system architecture where each component is an agent. The agents self-organise on the basis of their interactions with each other, where these interactions are driven by the goals and requirements of the individual agents.

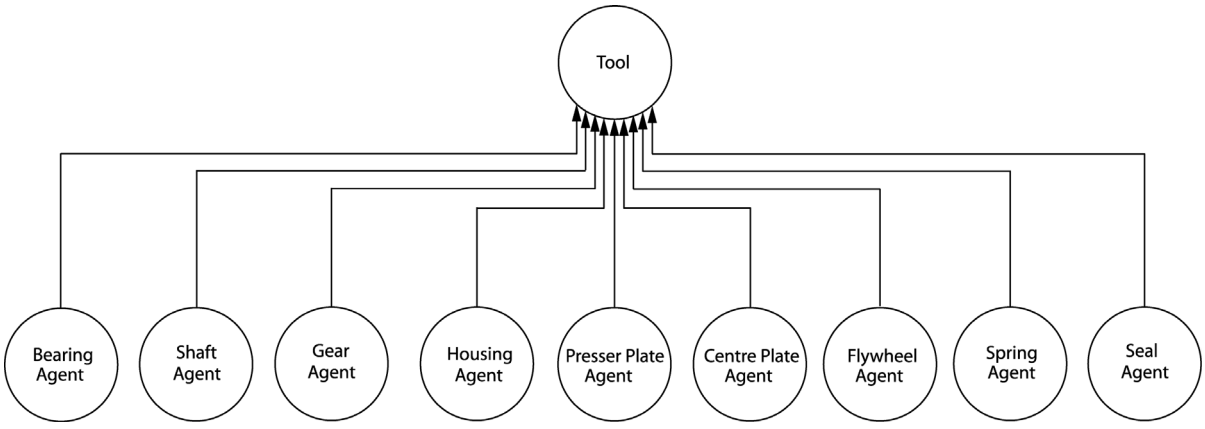


Figure 6. Emerging the product model by self-organising agents.

We are currently working on the development and implementation of a system architecture based on hierarchically structured agents. At this stage, we believe that this architecture is well suited to meet the requirements of coherence and robustness of the product model.

4. Discussion

We have proposed an approach towards a framework for agent-based product modelling, which has the potential to complement current standardisation efforts like STEP and IFCs. The advantage of this approach is that when tools are not interopable-ready they can still re-represent data in the necessary form in an effective manner. This is often required as new tools and new technologies are to be integrated in product development. Letting situated agents generate the right data in the right form as needed by the analysis tools has considerable advantages over attempting solely to maintain fixed interoperability in a dynamic industrial environment.

The approach described in this paper lays the foundations for a different class of tools. These tools are more "intelligent" not in terms of the specialised knowledge they contain but in terms of their ability to learn by being used. Their use is based on their interaction with other agents including agents they have not necessarily come in contact with previously. They are able to develop a form of interoperability based on the development of their communication as they attempt to interact with other agents. This interaction is the basis of their negotiation of a common ground.

Acknowledgements
This work is supported by a University of Sydney Sesqui Research and Development grant and by an International Postgraduate Research Scholarship.

References

[1] NIST, "Interoperability cost analysis of the US automotive supply chain", Planning Report #99-1, NIST Strategic Planning and Economic Assessment Office, Gaithersberg, Maryland, 1999.

[2] Eastman C., "Building Product Models: Computer Environments Supporting Design and Construction", CRC Press, Boca Raton, FL, 1999.

[3] ISO, "ISO 10303-11 Product Data Representation and Exchange – Part 11: Description methods: The EXPRESS language reference manual", International Standards Organization, Geneva, Switzerland, 1994.

[4] IAI, "Industry Foundation Classes", International Alliance for Interoperability, 2000, http://eande.lbl.gov/btp/iai/copyright_ifc2x.html.

[5] Dewey J., "The reflex arc concept in psychology", Psychological Review, Vol. 3, 1896 reprinted in 1981, pp.357-370.

[6] Bartlett F.C., "Remembering: A Study in Experimental and Social Psychology", Cambridge University Press, Cambridge, 1932 reprinted in 1977.

[7] Clancey W.J., "Situated Cognition", Cambridge University Press, Cambridge, 1997.

[8] Gero J.S., "Conceptual designing as a sequence of situated acts", in I. Smith (ed.), Artificial Intelligence in Structural Engineering, Springer-Verlag, Berlin, 1998, pp.165-177.

[9] Schön D. and Wiggins G., "Kinds of seeing and their functions in designing", Design Studies, Vol. 13(2), 1992, pp.135-156.

[10] Suwa M., Gero J.S. and Purcell T., "Unexpected discoveries and s-inventions of design requirements: A key to creative designs", in J.S. Gero and M.L. Maher (eds), Computational Models of Creative Design IV, Key Centre of Design Computing and Cognition, University of Sydney, Sydney, Australia, 1999, pp.297-320.

[11] Gero J.S. and Fujii H., "A computational framework for concept formation for a situated design agent", Knowledge-Based Systems, Vol. 13(6), 2000, pp.361-368.

[12] Kulinski J. and Gero J.S., "Constructive representation in situated analogy in design", in B. de Vries, J. van Leeuwen and H. Achten (eds), CAADFutures 2001, Kluwer, Dordrecht, 2001, pp.507-520.

[13] Smith G. and Gero J.S., "Situated design interpretation using a configuration of actor capabilities", in J.S. Gero, S. Chase and M.A. Rosenman (eds), CAADRIA2001, Key Centre of Design Computing and Cognition, University of Sydney, Sydney, 2001, pp.15-24.

[14] Gruber T.R., "A translation approach to portable ontologies", Knowledge Acquisition, Vol. 5(2), 1993, pp.199-220.

[15] Karttunen L. and Peters S., "Conventional implicature of Montague grammar", in C. Cogen, H. Thompson, G. Thurgood, K. Whistler and J. Wright (eds), Proceedings of the First Annual Meeting of the Berkeley Linguistic Society, University of California, Berkeley, CA, 1975, pp.266-278.

[16] Clark H.H. and Murphy G.L., "Audience design in meaning and reference", in J.F. Le Ny and W. Kintsch (eds), Language and Comprehension, North-Holland Publishing Company, Amsterdam, 1982, pp.287-299.

[17] Fussell S.R. and Krauss R.M., "The effects of intended audience on message production and comprehension: Reference in a common ground framework", Journal of Experimental Social Psychology, Vol. 25, 1989, pp.203-219.

[18] Clark H.H. and Marshall C.R., "Definite reference and mutual knowledge", in A.K. Joshi and B. Webber (eds), Linguistics Structure and Discourse Setting, Cambridge University Press, Cambridge, 1981, pp.10-63.

[19] Simon H.A., "The Sciences of the Artificial", MIT Press, Cambridge, MA, 1969.

[20] Gero J.S. and Kannengiesser U., "Function-behaviour-structure: A model for social situated agents", in R. Sun (ed.), Workshop on Cognitive Modeling of Agents and Multi-Agent Interactions, International Joint Conference on Artificial Intelligence'2003, Acapulco, Mexico, 2003 (to appear)

John S. Gero
University of Sydney, Key Centre of Design Computing and Cognition, Sydney NSW 2006, Sydney, Australia
Tel.: +61 2 9351 2328 Fax: +61 2 9351 3031 Email: john@arch.usyd.edu.au