DEVELOPMENT OF AN AGENT-BASED WORKBENCH SUPPORTING COLLABORATIVE STRUCTURAL DESIGN

Jochen Bilek and Dietrich Hartmann
Institute of Computational Engineering, University of Bochum
{bilek,hartus}@inf.bi.rub.de

SUMMARY

This paper presents an approach towards an adaptive, expandable, decentralized and flexible workbench supporting complex structural design processes using multiagent systems technology. Primarily, this novel workbench aims at assisting design experts according to their specific tasks during a project work and furthermore detecting typical deficiencies and conflicts that may occur in collaboration, cooperation and coordination between the various structural designers. The workbench consists of a set of software agents, that are designed and modeled to integrate typical organizational characteristics of a project work, engineering software and data structures in terms of product models. According to the analysis of structural design processes a theoretical concept of three agent-based submodels is suggested: i) the agent-based collaboration model, ii) the agent-based engineering software integration model and iii) the agent-based product model. In this paper we focus on the agent-based collaboration model. The three models are connected by an agent-based process model. The fundamental solution concepts of this approach are to be substantiated by analyzing the design process of an arched bridge, already erected, as a reference.

INTRODUCTION

The complexity of large engineering projects is permanently increasing and, therefore, requires the collaboration of many structural design experts. In particular, short deadlines and high quality demands enforce a close collaboration and coordination of all team members (Bretschneider 1997). But, contemporary large, project works are often composed of several participating enterprises and offices. These enterprises normally are situated in different districts, cities or even countries. Thus, the coordination and communication between the various design experts often shows deficiencies that may lead to delays in design and construction combined with an unexpected increase in costs. Another potential conflict is due to the plenitude of complex structural design processes that are carried out by the designers more and more concurrently to meet the cost pressure requirements. Obviously, interactions among the team members are necessary to resolve such conflicts. But, conventional software created in the last two decades, like finite element programs as well as CAD-systems, are not capable of performing the intricate work process carried out in practice directly.

By contrast, the multiagent systems technology provides ways and means to overcome the above mentioned deficiencies (Bilek/Hartmann 2002). Software agents are considered as autonomous problem solving units that are capable of assisting their assigned design experts and/ or other software agents with their specific domain expertise. The interaction of several spatiotemporally distributed software agents results in a flexible, adaptive, scalable, expandable and dynamic system. This is elucidated in the chapter *fundamentals* which is to give a short introduction to the multiagent systems technology with a specific emphasis on structural design.

Much fundamental work has been carried out on multiagent systems technology (Wooldridge 2002, Weiß 2000), however, only little research in this field is related to collaborative structural design. Mostly, multiagents systems research in this area covers specific areas of civil engineering like the integration of distributed engineering software tools in an agent environment (Khedro 1996, Shen 1996), specific structural engineering tasks like parallelization or optimization of FE-computations by means of software agents (Qian 2001), modeling collaborative work in structural design (Case 1996) or agent-based modeling of building data (Roseman 1999). These agent-based approaches do not meet the above mentioned deficiencies in a holistic and adequate way. There exists no approach to bring together the three important parts of structural design *collaborative team work, work on partial product models* and *usage of software engineering tools* into an overall agent-based system for structural design. To overcome this deficit this paper points out a way to adapt multiagent systems technology to the requirements that are determined by a holistic and comprehensive view on networked col-

laborative structural design processes. By that, an **agent model for collaborative structural design** is provided. This model is particularly based on the fundamentals of structural design pointed out in chapter 2. Applying agent-based software technologies, i) a representation and formalization of the human aspects of collaborative work, ii) a representation of the project management activities, iii) a representation of the structural system to be erected and iv) an integration of heterogeneous engineering software can be accomplished. The analysis of a typical structural design process yields three individual submodels, that are described later: first, the **agent-based collaboration model**, second the **agent-based software integration model**, and third, the **agent-based product model**. In this paper we focus on the agent-based collaboration model. The three models are connected by an **agent-based process model**.

FUNDAMENTALS

Characteristics of structural design

The analysis of structural design characteristics implies three aspects that have to be considered explicitly: i) characteristic organizations and actors that participate in a structural project work, ii) characteristic design processes combined with the underlying product model data and iii) the application of engineering standard software.

Organizational units in computational design processes are enterprises that are often specialized in specific areas of expertise. These organizations often cooperate with each other and then establish temporary working groups. With respect to the agent model for structural design, characteristic companies (e.g. for architecture, structural engineering, construction, etc.) are termed **permanent organizations**, whereas working groups of several societies are considered as **temporary organizations**. An organization team is composed of parts within a permanent organization. Other permanent organizations that participate in a complex structural engineering process are authorities, contractors, etc.

During a design process many persons are doing portions of work – they *act*. These actors have diverse tasks, rights and liabilities depending on their role in the organization. Actors can, therefore, be characterized by the role they play. A permanent organization may employ actors with the following roles: office manager, technical or mercantile co-workers or other. If an actor is integrated in a project work (by a temporary organization) his role may change to one of the following: technical director, project manager, structural analyst, structural designer, structural engineer or technician, draftsman, etc.

Structural design is an **iterative process** during which a central product model (CPM) is transformed from one state to the next until a final state is achieved (Bretschneider 1997). Customarily, the entire design process is subdivided into several smaller subprocesses. These are:

- preliminary design,
- structural analysis,
- design/code verification,
- structural detailing.

Each of these subprocesses lead to a partial product model (PPM), which, to a certain extend overlaps partially with the partial product models of the adjacent subprocesses.

A first qualitative digital model of the structural system is created during the subprocess of preliminary design in which the main parameters of the geometry, topology and the loads of a structure are determined.

The second subprocess, the structural analysis, applies the information established within the preceding preliminary design subprocess to determine the structural response in terms of the state variables (displacements and stresses) subject to the loading cases given. Customarily, the structural response is computed by FE program systems.

The third subprocess, the design verification, determines stresses of structural elements with respect to standards. If structural analysis is verified the structural data must be modified appropriately, which leads to iterative processes.

In the fourth and last subprocess, the structural detailing, computer aided visualization of the structure, its components and details takes place. This includes the detailing of structural parts and components, their connections as well as their supports.

During a complex design process a plenitude of **heterogeneous engineering software** is used by the

structural designers. This software can be categorized into finite element (FE) programs for structural computation, CAX software for the visualization of structural elements and the preparation of technical drawings, databases to store the structural data and project information and finally office software like MS-Excel or MS-Word. The various designers, normally, have access to software that is only locally implemented on their personal computer. It would be quite useful if they could also access software that is provided by other team members or that is server-side software.

Multiagent systems technology with respect to structural design

Multiagent systems technology covers a wide range of methodologies, concepts and ideas from many disciplines like distributed artificial intelligence (DAI), computer science, sociology, organization and management science, etc. (Ferber 1999). The intention of the research work presented here is to adapt this theoretical background to a practicable, agent-based workbench for structural design.

A multiagent system is composed of several interacting autonomous software agents that may reside on different computer systems connected by the internet. To support the mapping of the above introduced permanent and temporary organization structures, the software agents for structural design have to be installed on the individual personal computers and/or organizational servers. Therefore all participating designers and organizations are required to provide a permanent internet connection.

The coordination of complex planning processes presumes human communication. Accordingly, agent interaction presumes agent communication. That is to say that agents cooperate and coordinate with each other by exchanging messages. Most popular and comprehensible are speech-act based approaches. In this project the popular agent communication specification provided by the Foundation for Physical Intelligent Agents (FIPA) is used. Sequences of messages are termed **communication protocols**. The formalization of such communication protocols is accomplished by using Agent Unified Modeling Language (AUML), which is an extension to the UML and graph-theoretical approaches.

AGENT-BASED WORKBENCH FOR STRUCTURAL DESIGN

As shown above, the three important parts of structural design (actors/organizations, product model data, engineering software) have to be incorporated in a multiagent system for structural design, connected by design processes. There are some requirements that are to be satisfied while developing the overall agent-based model for structural design:

- *reusability*: The software agents should be reusable. That means the design and implementation of the software agents should be as much generic as possible. The specific domain knowledge of agents is kept in knowledge bases that must be upgraded during the complex design process.
- *modularity*: The internal structure of the agents should consist of reusable modules. It is also requested that new modules with specific functionalities could be added easily to a software agent.
- *knowledge adoption*: Every building is unique. This fact implicates the uniqueness of every structural design process. Thus, the knowledge of the agents (especially the knowledge of those that deal with structural data) cannot be generic in all parts. As a consequence, the identification of generic and non-generic parts of a structural design process is mandatory. Reasonably, the adoption of the non-generic knowledge of agents to the specific structural design process should be easily facilitated.
- *flexibility*: The multiagent system should be extendible. Thus, it should be possible to adopt new agents having new functionalities. Consequently, this is in harmony with the requirement of modularity: If an agent with a new functionality is added to the system co-agents can utilize this new functionality only if they are familiarized with it.

First research results indicate that a separation of generic and non-generic parts is extraordinarily essential. Generic parts come across with the use of specific engineering software and the organization of project teams. Non-generic parts concern the design and preparation of the structural data itself. This has led to the following subdivision of the agent-based model for structural design: first, the **agent-based collaboration model**, second, the **agent-based software integration model** and, third, the **agent-based product model**. Each model consists of several agents with specific functionalities (see Figure 1). The agent-based collaboration model contains *cooperative agents* to handle the project management, the agent-based software integration makes use of *wrapper agents* to encapsulate engineering software and last, the agent-based product model integration comprises *product*

model agents to represent the structural data and to maintain consistency. The sum of all acting agents represents then the **workbench for structural design**.

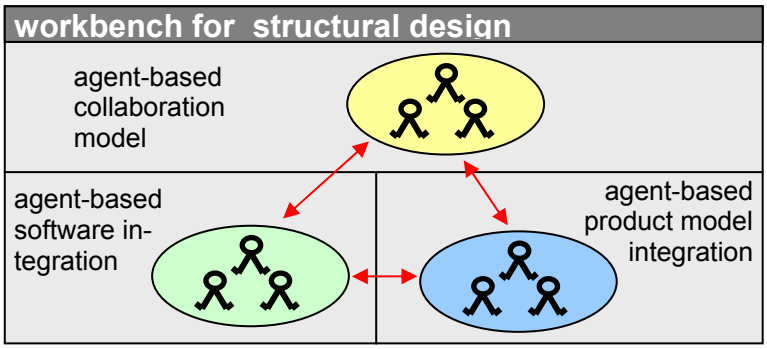

Figure 1 Basic agent model for collaborative structural engineering

Technical aspects

Selected agents of the model above have already been implemented using the XML-based agent system LARS (living agents runtime system). The required reusability is assured by using the Java Apache Avalon component framework as the basis for the internal agent structure with at least one communication module for sending and receiving messages. Additionally, each agent posseses a controller module that controls other modules and is responsible for the dynamic module loading and releasing. Dynamic module loading allows a flexible module composition of an agent not only at the start of an agent's lifecycle but during its whole lifecycle. The internal agent structure is independent of the underlaying agent system so that only the communication module has to be modified if the agent system is to be changed. LARS provides a message transport system that is capable of sending XML-encoded FIPA messages as well as serialized Java objects. Here the FIPA SL language specification is used as a basis for text messages. Agent interaction is supported by using ontology modules that depend on an agent's tasks and abilities. Ontology development follows the above mentioned models for structural design. Thus, an ontology for addressing the product model data, an ontology for collaboration and elementary ontologies to wrap software have already been implemented on a prototype level. The knowledge supplied by ontologies is kept into XML schema files that can be converted into JESS (Java Expert System Shell) facts and slots. Specifically, the JESS inference machine is applied to process rules on the knowledge bases defined. The various design experts can modify these rules and knowledge bases if necessary. Additional reusable modules, like Java Swing components for human agent interaction, are already realized as prototypes. Also, wrapper agents work as interfaces between conventional engineering software tools and, therefore, are equipped with specific modules addressing this software (s. chapter agent-based software integration).

The structural design process of a an arched bridge as a reference

The development of the three submodels introduced above is to be demonstrated by analyzing the design process of an arched bridge as a reference example. The bridge crosses the river Mulde in the city of Dessau (Germany) and was erected in 2000 connecting two precincts. The bridge is composed of four main structural elements: the steel arch (approx. 100m, slope approx. 22°) associated with 15 tension rods, the bridge deck composed of several steel panels, the arch and deck abutments founded on piles (see Figure 2).

Figure 2 Arched bridge over the river Mulde in the city of Dessau (Germany)

The workflow started with a first qualitative digital model of the bridge in advance to structural analysis. Hereby, the arch and deck were assumed to be separate aggregates. Since the enterprise responsible for the erection of the steel structural elements (arch and deck) did not specialize in building real geometrical sectors, another computational concept was used in which the arch and deck had a segmental geometry. In the first qualitative model the deck was carried by means of the arch. Due to the inappropriate dynamic behavior of the bridge the structural system had to be altered. Instead, a dynamic computation was applied in which tension rods in place of ropes were used. During this iterative process building and awarding authorities had to be informed. The structural designers had to take into account the desires and requirements of the authorities.

Due to an evaluation of the design process the following participating organizations and persons (companies/ authorities, staff/ planners) have been identified:

- the city of Dessau as the awarding authority (1 public servant),
- the building authority again, in terms of the city of Dessau (1 public servant),
- the main planning office MPO (1 stress analyst for the deck, 1 project manager, who is stress analyst for the arch as well, and an office director),
- the performing steel company (1 technician for the structural steelwork),
- the performing concrete company (1 structural engineer and 1 draftsman),
- the incorporated pile company (1 engineer).

The above persons and organizations composed a new temporary organization during the project work. Summarizing, 8 actors employed to 6 permanent organizations (authorities, planning offices/enterprises) were involved in the design process.

Agent-based collaboration model

The agent-based collaboration model primarily deals with communication and coordination support. More precisely, the agent-based collaboration model integrates i) the application of modern software technologies supporting, synchronous and asynchronous communication between the spatiotemporally distributed team members and ii) the workflow coordination based upon software agents. In particular, the characteristic organizational structures of the project work have to be mapped into the agent system in an adequate fashion.

Each team member needs a graphical user interface to interact with the workbench. Hence, humans are assisted by *personal cooperative agents* “living” on their personal computer. Basically, each team member belongs to a *permanent organization* (enterprise, planning office, etc.). The mapping of this organizational structure is achieved by using *nonpersonal cooperative agents*. Each organization is represented by such a nonpersonal cooperative agent, having knowledge about the organization members and tasks that are to be carried out. Furthermore, the nonpersonal agents are given administration privileges, access rights, resources, etc.

In correspondence to the introduced *permanent* and *temporary organizations*, enterprises are represented by *enterprise agents* and *project works* analogously by *project agents*. Temporary organizations are composed of parts of permanent organizations. The permanent organization’s members (mainly technical and mercantile staff) play specific roles in their dedicated organization but their role may change if they become member of a temporary (project work) organization. Their role depends on engineering tasks they are responsible for. An engineer may also be member of different projects at the same time playing different roles.

Again, considerations of the reference example (arched bridge in Dessau) can be used for the modeling of the organization of the agents. In the Dessau project, the project work was initiated by the authorities of the city of Dessau. During the structural design process of the bridge several enterprises and authorities were charged with specific design tasks. The *enterprise MPO* (main planning office) was given the control over the project management. Thus, the MPO employed one of its engineers as the *project manager*. The project manager also accounted for the structural analysis of the arch. A further engineer of the MPO was directed to design the deck. Thus, he occupied the role of a *stress analyst*.

The piles were designed by one engineer of the *pile company* (role stress analyst). The technical drawings of the structural steelwork were created by a *tracer* as a member of the *steel inc.* The steel inc. erected the bridge. A further company that participated in the project was charged with the concrete construction. This company employed one *structural engineer* and one *tracer* to design, compute

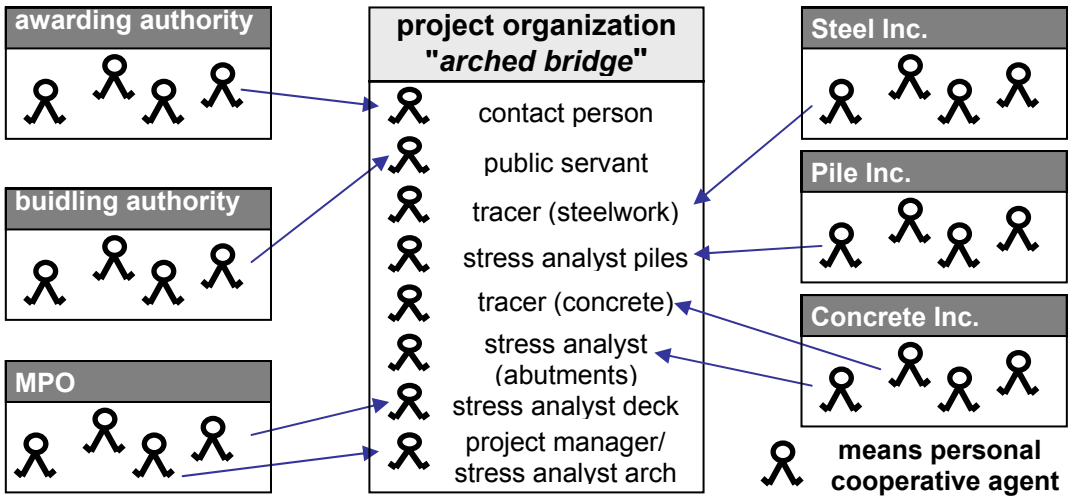

Figure 3 Organisation model “arched bridge”

and draw the abutments. Finally, the *building authority* of Dessau was involved. The building authority was represented by one of its *public servants*. As a result, 8 personal cooperative agents assisting the human actors, and 7 nonpersonal cooperative agents (6 permanent and 1 temporary organization) representing the organisational structure accomplish the organisational mapping of the structural design process (see Fig. 3). Based on this organizational structure of the agent-based system the workflow of a structural design process has been integrated into the workbench.

Conventional workflow concepts are adapted into the agent-based workbench as well (see Fig. 4). Basic parts of conventional workflow systems are the *workflow scheduler* (task coordination), the *monitoring service* (task execution control), the *worklist* (processes and activities, that have to be performed), the *dispatcher* (controls and organizes the task queue) and several *task manager* (responsible for the execution and control of tasks).

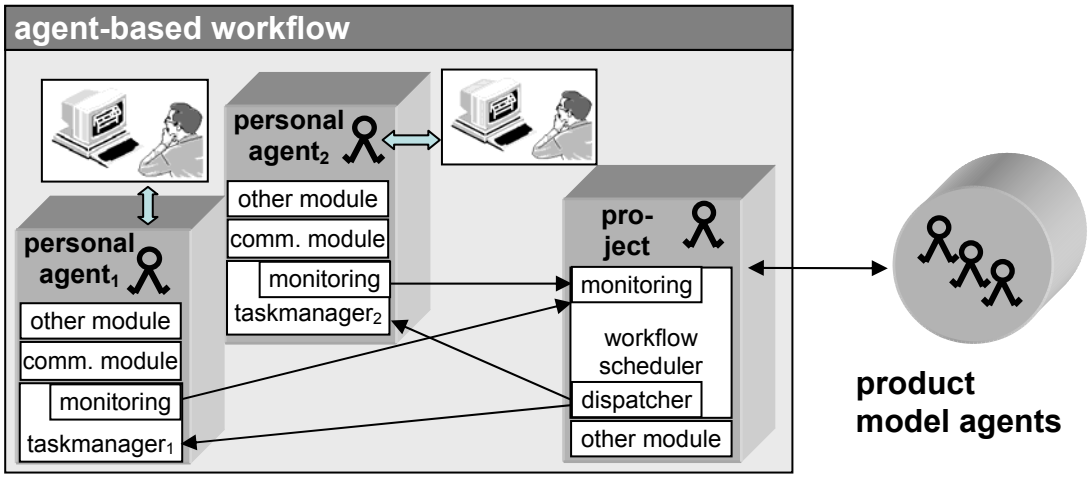

Figure 4 Agent-based workflow management

The workflow scheduler is integrated into the respective *nonpersonal cooperative project agent* in terms of a module (along other modules, like a communication module). The enhanced project agent administrates the *worklist*, the *dispatcher*, the *monitoring service* and delegates structural tasks to the personal cooperative agents of the team members. The cooperative agents, therefore, work as *task manager* for their principals. A large project work may imply thousands of workflows. Therefore, it is useful to decompose the project agent into several interacting project agents.

In structural engineering, *manual acitivities* cover an enormous amount of design activities because of the uniqueness of each structural design process. As a consequence, activities that can be automatized by invoking software applications play only a minor role. Hence, the project agent’s main task is to monitor the delegated manual activities and processes, to reveal inconsistencies and conflicts in the workflow and to forward them to the liable team members. Leading team members are qualified for modifying the worklist and workflow scheduler at any time. By that, the high dynamic and complex

characteristics of structural design processes are realized. The workflow is controlled and formalized by deploying Petri-net technologies. With respect to structural design, workflows are connected with structural product model data. Therefore, project agents serve as interfaces between workflows and the agent-based product model integration that is pointed in the chapter before last.

Agent-based software integration

Software integration is achieved by applying the *wrapper agent* concept. Wrapper agents act as an interface to standard engineering software. The workbench's software agents interact by exchanging messages. A wrapper agent converts incoming (client) messages into access commands for specific software packages and returns the results of the software. This is an analogy to the client-server paradigm: The wrapper agents represent the server while other agents represent the clients. The difference to the client-server paradigm is that a wrapper agent is not the server itself rather provides only access to it while *encapsulating* the (server-side)software. Furthermore, the encapsulated software does not need to be server-side software: wrapper agents can encapsulate any kind of software that can be used by other agents. Wrapper agents for structural design make use of several technologies to access the requested software: middleware technologies like CORBA, COM, SOAP, SQL or other (see Fig. 5). A direct program call with given parameters or input files can be carried out as well, e.g. via the Java runtime library.

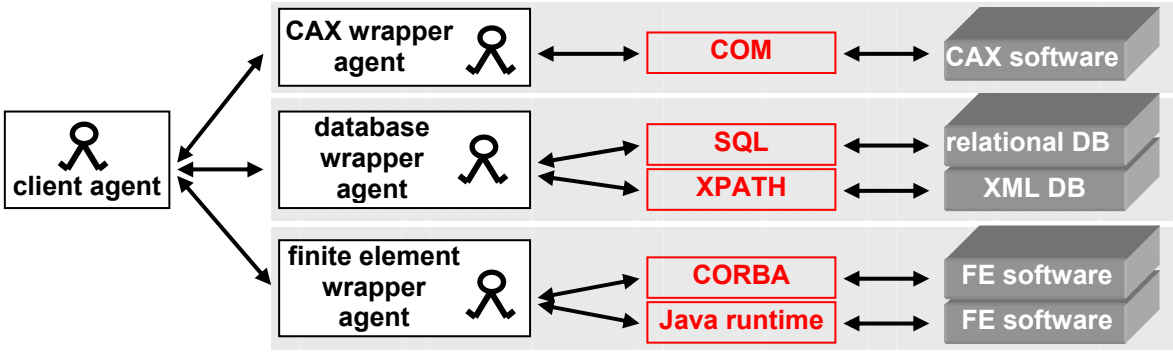

Figure 5 Software integration using wrapper agents

In the current research project, the software actually used in the reference project (Dessau bridge) is not available, due to license regulations. Instead, three wrapper agents integrating comparable software are modeled: i) a database wrapper agent that encapsulates the access to two different database systems (a relational database: MySQL using SQL, an XML-based hierarchical database Xindice using XPATH), ii) a FE wrapper agent (encapsulating the FE program systems Ansys5.6 and Felt) and iii) a CAD wrapper agent that makes use of the AutoCAD2000 COM-interface for generating technical drawings. All wrapper agents have been implemented and evaluated in the meantime. They are composed of the above mentioned generic agent structure and expanded with software specific modules and ontologies. Client agents must implement these ontologies to deploy the services that are provided by the corresponding wrapper agents. The AutoCAD wrapper agent for example can be used for processing drawings created from product model data. In total, the interaction of several AutoCAD wrapper agents represents a distributed CAD system applied by several draftsmen which work on one drawing in parallel and interactively.

Agent-based product model integration

The integration of structural data and information is gained by deploying *product model agents*. In a first step one single product model agent is created for each main structural element. The product model agents store the (partial) product model information of the data they are responsible for, manage the access to their administered structural data, and inform other subscribed agents about modifications made in the stored data. The interaction and information exchange between the individual product model agents allows for revealing inconstancies and design conflicts. Conflicts, of course, can be handled only if appropriate knowledge is captured in the agents. Such knowledge is emanated from the responsible structural designers.

Product model agents are organized hierarchically according to the structural decomposition used. On top of the hierarchy, a supervisory product model agent resides that acts as an interface to the structural data and has knowledge about the dependencies between structural elements. Thus, the effec-

tive amount of communicative load is decreased. All data must pass the supervisory product model agent that makes use of its knowledge to check dependencies and find inconsistencies. Obviously, because of the limited knowledge of the supervisory agent, at present, not all inconsistencies can be detected.

In the reference application considered here, four main structural elements can be constituted: the arch, the deck, the arch abutments and the deck abutments. Consequently, this results in four product model agents controlled and supervised by one product model agent. In a first step, the product model data is provided by a specific agent that uses XML for data representation. The XML product model data is determined by an XML schema file. Ongoing research will focus on the further development of the agent-based product model integration.

CONCLUSIONS AND FUTURE WORK

The agent-based workbench for structural design points out ways and means to bring together the three important parts of structural design *collaborative team work, work on partial product models* and *usage of software engineering tools* into a unified agent-based system for structural design. The workbench provides much more efficient computer assistance for collaborative structural design than a set of single agent-based approaches that only cover specific facets of structural design. The approach is adaptable, scalable and expandable due to the generic structure of the agents used, the concept of dynamic module loading chosen and the possibility of adding new agents to the workbench by runtime.

Some agents of the workbench have already been implemented as prototypes. Particularly, the agent-based software integration has substantial advantages: engineering software can be used easily by other agents and design experts in various scenarios. The CAD wrapper e.g. enables several draftsmen to work on the same drawing interactively and in parallel. The agent-based collaboration model is not yet fully implemented but the conception of the model shows that an adaptation of cooperative agents to other project works is possible at moderate expense. Within the next year, the most substantial agents of the workbench will be prototypically implemented.

REFERENCES

Bilek, J. and Hartmann, D. (2002) *Collaborative Structural Engineering based on Multiagent Systems*, Advances in Intelligent Computing in Engineering, EG-ICE Workshop 2002, Darmstadt, Germany.

Bilek, J. and Hartmann, D. (2002) *Kooperative Tragwerksplanung basierend auf Multiagentensystemen*, p. 287-306, Bauen mit Computern, VDI reports 1668, Düsseldorf.

Bretschneider, D. et al. (1997) *Modelling collaborative engineering in object orientated design systems*, Mouchel Centenary Conference. Cambridge.

Case, M.P., Lu, ScC.-Y. (1996) *Discourse Model for collaborative Design*, Computer-Added Design – Special issue: Computer-aided concurrent design, 28(5): p. 333-345.

Ferber, J. (1999) *Multi-Agent Systems: An Introduction to Distributed Artificial Intelligence*, Addison Wesley, Edinburgh, UK.

Khedro, T (1996) *AgentCAD: a distributed cooperative CAD environment*, In B. Kumar, A. Retik, editor. Volume Information Representation and Delivery in Civil and Structural Engineering Design', p. 15-19., Civil-Comp Press, University of Strathclyde, Glasgow.

Norrie, D. H. et al. (2000) *Multi-Agent Systems for Concurrent Intelligent Design and Manufacturing,* Taylor & Francis, London, UK.

Qian, Z., Xue, C., Pan, S. (2001) *FEA agent for multidisciplinary optimization*, Struct Multidisc Optim 22, p. 373-383. Springer-Verlag. Berlin.

Rosenman, M. and Wang, F. (1999) *CACOM: A Component Agent-based Design-Orientated Model for Collaborative Design*, Research in Engineering Design11, p. 193-205.Springer-Verlag. London.

Shen, W., Barthes, J.-P. (1996) *An experimental Multiagent Environment for Engineering Design*. International Journal of Cooperative Information Systems, 5(2-3), pp 131-151.

Weiß, G. (2000) *Multiagent Systems – a Modern Approach to Distributed Artificial Intelligence*, Massachusetts Institute of Technology.

Wooldridge, M. (2002) *An Introduction to Multiagent Systems*, John Wiley & Sons, Chichester, England.