AWARENESS AND WORKFLOW BASED COORDINATION OF NETWORKED CO-OPERATIONS IN STRUCTURAL DESIGN

SUBMITTED: January 2006
REVISED: April 2006
PUBLISHED: July 2006 at http://www.itcon.org/2006/36/
EDITOR: P. Katranuschkov

Sascha Alda, Dipl.-Inf.
Research assistant of the Institute of Computer Science, University of Bonn, Germany
email: alda@cs.uni-bonn.de

Jochen Bilek, Dipl.-Ing.
Research assistant of the Institute of Computational Engineering, University of Bochum, Germany
email: bilek@inf.bi.rub.de

Armin B. Cremers, Prof. Dr.
Director of the Institute of Computer Science, University of Bonn, Germany
email: abc@cs.uni-bonn.de httpd: http://www.informatik.uni-bonn.de/III/

Dietrich Hartmann, Prof. Dr.-Ing.
Director of the Institute of Computational Engineering, University of Bochum, Germany
email: hartus@inf.bi.rub.de httpd: http://www.inf.bi.rub.de

SUMMARY: *Modern engineering projects in the application domain of structural design are organized as networked co-operations due to permanently enlarged competition pressure and a high degree of complexity while performing concurrent design activities. One of the major challenges of these networked co-operations constitutes the coordination of the activities of all involved participants. Recent off-the-shelf software systems, however, fail for coordinating these complex activities in this area. In the course of our research activities, we have developed two different directions for coordinating such projects: (1) a workflow-based concept regulating the activities by a global workflow model (University of Bochum) and (2) an awareness model that allows to perceive activities of other partners and to derive new activities mentally (University of Bonn). This paper describes a novel integration approach of these two models: according to the global workflow model, users can be informed such that they are enabled to detect potential inconsistencies while modeling activities at an early stage.*

KEYWORDS: *Peer-to-peer architectures, multi-agent systems, workflow, networked co-operations, structural design, Petri nets, awareness.*

1. INTRODUCTION

Large engineering projects in construction engineering necessitate the involvement and the close collaboration of different engineers and other specialists from different geographic locations. We indicate such virtual project constellations as so-called *networked co-operations*. Networked co-operations in construction engineering exhibit major benefits in particular for organizing complex planning processes (Bilek et al, 2005). During the last years, modern group-supporting software systems such as groupware systems (see (Schwabe et al, 2001) for an overview), traditional email clients (e.g. MS Outlook, Modzilla, Thunderbird), document configuration and exchange tools (e.g. CVS, Email), shared workspaces (e.g. BSCW (Fraunhofer, 2005)) or chat systems (e.g. ICQ (ICQ, 2005)) have been utilized for supporting these co-operations. These systems enable partners to share CAD documents (e.g. through Mail or BSCW), to negotiate important design decisions synchronously (e.g. chat tools, video conference) or to derive an accumulated view of the project progress (e.g. agenda systems, whiteboards).

One of the major observances we could make during the course of our research endeavours is that these state-of-the-art off-the-shelf systems are not sufficient for supporting networked co-operations in construction engineering. All mentioned systems enhance the communication among stakeholders of a co-operation but do not incorporate any way to support their coordination. The coordination of partners within a networked co-operation is, however, of major importance for guaranteeing the correct activation and completion of activities. Coordination models also have to ensure the consistency of various partial planning models that exist within a co-operation. Uncontrolled modelling would lead to inconsistencies in the global planning model. Formal process models such as Petri nets (Petri, 1973), pi-calculus (Milner, 1991), or EPK (van der Aalst, 1999) are well known in computer science. Despite their popularity, these models have not been considered for the adoption in recent groupware systems. The appreciation and implementation of these models in engineering informatics has also been pure for a long time. This circumstance can be justified by the tremendous (if not impossible) challenge to map complex *activities* (e.g. planning, structural design), *methods* (e.g. finite element analysis, object-oriented modelling), *roles and views* (e.g. structural engineer, architect, client), *documents* (i.e. different syntax, semantics), and *organizational concerns* (i.e. the trend to virtual project settings, fluctuation of partners) prevailing in this field to a formal model. Moreover, the application environments that can be found at single partner sites are of *heterogeneous* nature. Partners insist on working with proprietary software, use incompatible document exchange formats, or have various hardware infrastructure with different access abilities to the Internet (e.g. Modem, DSL). In addition, partners customize their own working practices (e.g. work at night, hide working details). A process model embracing all those different heterogeneous "parameters" is likely to become complex and unintelligible.

In the course of our common research projects at the Universities of Bonn and Bochum, we have conceived holistic solutions for improving the coordination of networked co-operations within civil and building engineering. Our work has been motivated by evaluating and identifying software architectural styles for building specific software architectures to support processes especially for distributed project settings. The project of Bonn has adopted the *peer-to-peer architectural style* for setting up a concrete architecture (CoBE), in which end-users are directly involved during the execution of software. The project of Bochum focuses on the integration of semi-autonomous behaviour by means of so-called *software agents*. In the resulting agent-oriented architecture (ACOS), agents allow for reacting on changes in their environment and to proactively pursue own activities. In addition, both projects evolved novel concepts for integrating heterogeneous resources to accomplish the unified access to software and documents of individual partners. The selection of the architectural style has also determined the utilization of a process model for supporting the coordination of partners. This has led to two different approaches. The Bochum project has proposed an *agent-based process model* to *autonomously* control and to distribute activities and project resources defined in a global *workflow model* to the appropriate partners. The workflow execution model is rigorously based on the Petri nets theory. The coordination model of CoBE strives a different motivation: any local software can be composed with the COBE AWARENESS FRAMEWORK that allows to reveal activities on planning models and to make them perceivable to other partners. Based on this gained information, new activities can be derived mentally. Both process models have been formulated in either formal or semi-formal way and have been developed as prototypes in the respective software architectures.

The goal of this paper is to present new material that has resulted from the integration of these two coordination models. The integrated model is itself based on an integrated architectural approach, called the MAS-P2P architecture (Alda et al, 2004). The integrated coordination model tries to consolidate the strengths of both models while at the same time remedying their obvious drawbacks. This can in particular be demonstrated to the conflict management when modelling on different structural design models. While Petri nets only provide minor support for handling *unanticipated* conflicts, an awareness model turns out to be more practicable as it involves the consciousness of partners to resolve the conflict. The basic idea of our integration approach therefore is to identify potential conflicts within a given workflow model, determine the pertaining partner, and eventually to *acquaint* these partners with respect to the global awareness model. Then, partners are capable of perceiving each other's activities and to execute appropriate activities whenever a conflict becomes obvious.

The rest of the paper is structured as follows: section 2 summarizes related work, and section 3 provides background information about both involved projects and outlines the concepts of both architectural styles. In section 4, we first introduce the project specific coordination models used for the support of concurrent design activities, i.e. Petri nets and awareness-based coordination. Additionally, pros and cons of both coordination approaches are assessed. Section 5 presents the MAS-P2P architecture including an extensive description of our integrated coordination model. This section also features a demonstrative structural design scenario that demonstrates the potential and sustainability of our proposed coordination approach. Section 6 concludes the paper outlining impact and current limitations of our work and envisaged future works.

2. RELATED WORK

The development of sophisticated computer supported, collaborative process support models for the structural design domain is an emerging field of research due to the increasing availability of powerful hardware and high speed networks, particularly the worldwide internet. In the recent years, several approaches to support co-operative design processes in the structural design domain and - more generally in the AEC sector - have been made.

One group of research projects concentrate on **database systems** to exchange relevant planning information between the various, co-operating design experts. Hereby, collaborative work either is based upon a common database (blackboard) or relies upon application specific exchange formats. Fundamental research focussing on common, shared databases was for example settled in the iCSS (Juli and Scherer, 2002), COMBINE (Fenves et al, 1994) and DICE (Maxfield et al, 1995) projects. In contrast, the ToCEE/COMBI projects (Scherer and Sparacello, 1996) (Scherer et al, 1998) concatenate independent software modules with STEP/ISO 10303 exchange files. These data-based research projects evidence that it is vital to use high-level, generalized product model specifications like IFC or CIS/2 when developing collaborative design models for structural design. We take that into account with our proposed product model agents (section 5.2).

Another group of research projects considers **decision support** as nucleus for co-operation in structural engineering like IT-Code. IT-Code is a cross platform co-operation system based on internet technologies and shared workspaces (Lai et al, 2002). Fundamental concept of IT-Code is the ontology and Semantic Web based knowledge acquisition and composition for the early design stages and the transparent decision support for innovative and ho-hum design, co-operative design tasks. Kalay's P3 architecture (Kalay, 1999) enriches elementary, collaborative decision support with *world-views* and *fuzzy* constituents.

A more technically view on collaborative work have **middleware-oriented** approaches like Indusys or COMMIT. Indusys is a co-operative work supporting software system for structural engineering (Bretschneider and Hartman, 1999). It employs CORBA technologies to connect the application independent process model *Cooperate* with the inherent object-oriented product model *PlaKon*. Similarly, the COMMIT framework (Brown et al, 1996) deploys CORBA components to link the implemented collaborative design features.

Some research works primarily focus on modeling the complicated, interdependent design processes carried out concurrently or asynchronous. In this context, **workflow-based** systems have been taken into account for the coordination and control design tasks. A major research work, hereby, is the PROMISE project. PROMISE is a client-server based workflow simulator for the definition and instantiation of colored Petri nets. Workflows are classified as *structured, unstructured* and *semi-structured* structural engineering processes. PROMISE even contains a workflow engine to distribute fireable work items to SOAP-connected (Simple Object Access Protocol) clients (Greb et al, 2004). VEGA is another, important workflow based system (Zarli and Poyet, 1999). It supports co-operative teamwork by linking digital product models with WfMS concepts. The analyzed, workflow-based approaches substantiate the fundamental need for integrating sophisticated process models into collaborative software platforms in the structural engineering domain. Beyond, the deployed process models have to be affiliated to standardized, shared product models as figured out before. We incorporated workflow-based concepts and product model based approaches into specific ACOS agents as illustrated in section 3.1.

In recent years, **agent-based** projects in various engineering disciplines are on the rise. Theiss et al (2004) have conceptualized MADITA, a multiagent system for network-based fire engineering. In MADITA, agents are responsible for the accurate allocation of fire protection specific information to the participating designers during all planning phases. Similarly, Mittrup et al (2003) propose a computer-based monitoring system for safety-relevant engineering structures in which purposive and precise information distribution is delegated to interacting agents. Even in the field of finite element computations and optimization (Mueller et al, 2005) (Qian et al, 2001), multi-site scheduling (Sauer et al, 1998) or CAD support (Rosenman and Wang, 2001), (Maher et al, 2003) agents are successfully employed. Moreover, several agent-based approaches to model workflow systems have been ascertained (Purvis et al, 2004) (Hawryszkiewycz and Debenham, 1998) (Stormer and Knorr, 1998). Summarizing, agent technologies eminently support large-scale software engineering of distributed systems (Nwana et al, 1999) and exhibit an enormous potential to support collaborative, structural design, particularly in combination with sustainable process and product models as we evaluated with our integrated ACOS-CoBE architecture.

In our recent research work we mainly focussed on conceptual and technical issues of an integrated architecture and corresponding coordination models for enhancing structural design processes. In fact, we have yet not considered **human and social implications** necessary when introducing such architecture and coordination model into concrete project settings. In the area of CSCW and groupware research, investigations have shown problems when introducing novel software infrastructures into a group of workers (Grudin, 1994). The introduction becomes in particular a problem when projects employ people who are unskilled or who are not willing to appreciate new and innovative technologies. Social methods such as ethnography have become popular in this area to understand work processes in existing groups and to derive implications how to introduce group-supporting software in a sensitive way (Hughes et al, 1994). The consideration of existing work from these social disciplines would certainly benefit our current research findings.

In this work, the *dynamic* in structural design processes is primarily identified by the fluctuating organization of such processes. In order to have a complete picture of those processes, further occurrences of dynamism should actually be incorporated as well. In particular, the project's *knowledge* is also highly dynamic as it changes and evolves in regular intervals. Project partners are therefore asked to adapt to new knowledge standards by means of **learning processes**. In this context, fundamental theories and practices of (group) learning could be reviewed for our approaches (e.g. 'reflection-in-action' learning model (Schoen, 1983), constructivist learning model (Kopp et al, 2001). A first approach how the DEEVOLVE platform could utilize the constructivist learning model in order to support construction engineering courses has been demonstrated in (Alda et al, 2005). In the RETEx II project, Schoen's learning model has been analyzed and integrated to support co-operative design in construction - see (Forgber, 1996) for more information.

3. BACKGROUND: ARCHITECTURAL STYLES FOR STRUCTURAL DESIGN PROCESSES

In the course of the priority program 1103 "Networked-based co-operative planning processes in structural engineering" funded by the German Research Foundation (DFG-PP1103, 1999), we have scrutinized possible software architecture models for enhancing networked co-operations within collaborative structural design work. In this context, two major models evidenced as feasible: 1) the peer-to-peer architecture style that was evaluated by the University of Bonn and 2) the multiagent architecture style evaluated by the University of Bochum. In the subsequent sections, we will outline fundamental concepts of both models.

3.1 Multiagent architecture

Multiagent systems (MAS) are a rapidly developing area of research that emerged from the distributed artificial intelligence (DAI) (Wooldridge 2002, Ferber 1999). Within the Bochum project a *software agent* is defined as an "*encapsulated software unit that is situated in a dynamic, heterogeneous environment, capable of solving well defined tasks autonomously and pro-actively in co-operation with other agents by order of personal (human) or non-personal principals*". A software agent, hence, is to be considered as an autonomous problem solver that is capable of proactively interacting and communicating with other agents on a high semantic, ontology-based level (Weiss, 2000). Thus, a multiagent system (MAS) is understood as a loosely-coupled network of several, autonomous and interacting problem solvers (= software agents) that work together to solve some problems being above their individual capabilities. The sum of several interacting agents' capabilities in a MAS, therefore, exceeds the sum of its individual parts.

In practice, *s*oftware agents primarily assist their assigned design experts in their activities aside the support of other software agents in the environment. Hence, it is not surprising that multiagent systems are suited to solve the above outlined problems of distributed collaborative work in structural engineering in a natural fashion. The adaptation of agent technologies to the specific needs and requirements of collaborative structural engineering implicitly requires the *decomposition* of the *entire structural design process* into adequate, domain-specific interacting agents (Bilek and Hartmann, 2003).

In our research work we identified the following four substantial domains that were incorporated into the overall *agent model for collaborative structural design*: (1) the participating specialized design experts and their associated characteristic, dynamic organizational structures - *agent-based co-operation model*, (2) the specific structural design processes - *agent-based process support model*, (3) the associated (partial) product models - *agent-based product model*, and (4) the applied engineering software - *agent-based software integration model* (see Fig. 1).

Within the agent-based co-operation model (1) human experts and their assigned, design-specific organizations (like specialized engineering offices, other building companies, and authorities) are represented by *co-operation agents*. It is vital to differentiate between two fundamental co-operation agent types, the *personal* and the *non-personal co-operation agents.*

Personal co-operation agents directly assist their assigned individuals in their design activities. For that reason they provide a graphical user interface that enables the human experts to interact with other project associates and the multiagent system.

Non-personal co-operation agents can be seen as virtual representatives for the participating organizational entities, primarily the building companies. They hold specific information about their associated organization like enlisted project co-workers, time schedules and design tasks that have to be conducted. A very special non-personal co-operation agent is the *project agent*. It represents and supports the common, intrinsic project work by supervising and controlling the project handling, administrating the project members, responsibilities and delegating design tasks.

The next fundamental agent-based submodel, the *agent-based process support model* (2), is to control and allocate the complex design processes to convenient participating designers. For that, a *workflow agent* has been modeled as a delegate to the project agent. Based on the Petri net theory, the workflow agent links design processes to project resources (product models, software, human experts). For a more detailed depiction of the incorporated Petri net based workflow concepts see section 4.2.

The third basic submodel, the *agent-based product model* (3), rests on the decomposition of the entire structural system into smaller structural subsystems. Each structural subsystem thereby is accessible via an assigned product model agent that owns and stores knowledge about several structural elements created by the participating structural designers during the design process. The product model agents adjust their knowledge and, thus, check structural dependencies and retain product model consistency. The product model specification used conforms to the CIS/2 standard (CIM steel integration standard). CIS/2 is a set of formal computing specifications that are based on ISO 10303 (STEP – STandard for the Exchange of Product model data) and allow the modeling of steel structures. CIS/2 covers the three major planning phases structural analysis, structural design and fabrication aside data management issues (Reed, 2002).

The integration of heterogeneous engineering standard software (such as CAD-, FE-programs, databases) is achieved through the implementation of wrapper agents as specified in the *agent-based software integration model* (4). Wrapper agents operate as interfaces between the MAS and locally installed software such that the software applications can be used by all other agents and/or human design experts in the agent network.

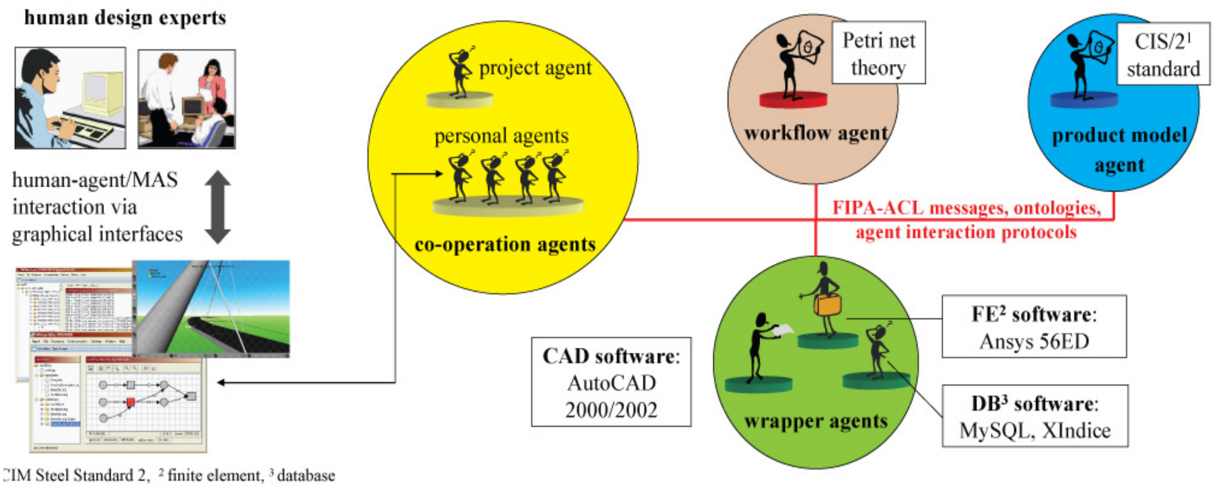
human design experts
human-agent/MAS interaction via graphical interfaces
project agent
personal agents
co-operation agents
Petri net theory
workflow agent
CIS/2[1] standard
product model agent
FIPA-ACL messages, ontologies, agent interaction protocols
wrapper agents
CAD software: AutoCAD 2000/2002
FE[2] software: Ansys 56ED
DB[3] software: MySQL, XIndice
IM Steel Standard 2, [2] finite element, [3] database

FIG. 1: The ACOS agent-based architecture.

In the course of the prototypical implementation ACOS (**A**gent-based **CO**llaborative **S**tructural Design) the delineated four submodels have been simplified such that only basic and necessary conceptual elements must be implemented into a set of design-specific software agents (see Fig. 1). These agents interact with each other by exchanging several speech-act based messages that conform to the FIPA (Foundation for Intelligent Physical Agents) agent communication language (FIPA-ACL) specifications (FIPA, 2002). The realisation of complex dialog structures is provided by task specific interaction protocols (IP). In addition, several domain specific ontologies have been developed (product model ontology, workflow ontology, etc.) and concatenated with agent specific capabilities. It has proved that the sum of several loosely coupled, design specific software agents in a dynamic environment is a convenient, flexible way to support collaborative structural design activities.

3.2 Service-oriented peer-to-peer architectures

Service-oriented peer-to-peer architectures refer to the class of software architectures that feature a set of equal nodes, so-called *peers*, where each peer is capable of functioning as a provider and as consumer of an arbitrary number of *peer services* encapsulating functions, hardware routines, or public access to documents. Consumed services can be composed to new, more complicated applications or even new services that can in turn be located and used by other third-party peers (see typical constellation in Fig. 2). This architectural style constitutes a logical enhancement of traditional peer-to-peer architectures relying on the provision of a small number of services (Shirky, 2001), (Brookshier et al, 2002) with novel concepts from the area of service-oriented architectures (Cervantes and Hall, 2005) in particular service composition and service description. In contrast to centralized service-oriented architectures, peer-to-peer architectures assume an unstable and dynamic topology as an important constraint due to the sole responsibility of peer owners to affiliate to a network. Beyond the possibility of direct resource sharing, peer-to-peer architectures enable single peers to organize into so-called *peer groups*. These self-governed communities can share, collaborate, and communicate in their own private web. The purpose is to subdivide peers into groups according to common interests or knowledge independent from any given organizational or network boundaries.

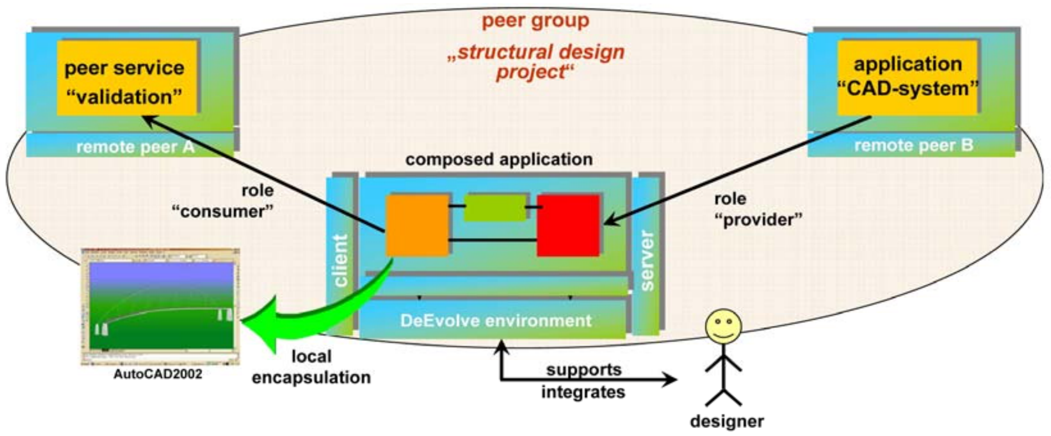

FIG. 2: The D<span style="font-variant:small-caps">e</span>E<span style="font-variant:small-caps">volve</span> *architecture.*

In the course of the research project CoBE, we have developed DEEVOLVE (Alda and Cremers, 2004), a self-adaptable peer-to-peer architecture based on top of the JXTA standard peer-to-peer framework (Sun, 2004). DEEVOLVE incorporates the *component methodology* as another technical fundamental basis. According to the component approach, peer services can be created by the composition of single software components. A component model called FLEXIBEAN prescribes the valid interaction primitives for both the local interaction within a service and the remote interaction among distributed services. Peer services can be made available (published) to other peers by means of *advertisements*. Each service can be assigned to at least one or more group affiliations, in order to restrict the access to a service for authorized group users. Users of other peers are able to discover and use these services. Additionally, we provide the composition language PeerCAT (Alda and Cremers, 2004), which enables users to declare the composition of different peer services towards individual, higher-level applications.

In order to integrate existing legacy applications, we deploy so-called *bridge components*. Such a component allows for mediating between the standardized FLEXIBEANS interface and other interfaces such as Microsoft's COM interface. To this end, we are able to encapsulate widely used CAD applications like AutoCAD 2002 in a peer environment and to publish it as a peer service into the peer-to-peer architecture. This feature allows for directly sharing CAD-based documents among peers and serves as the basis for assimilating information about the progress of individual modeling activities to dependent peers (see AWARENESS FRAMEWORK in section 4.1).

As a self-adaptable architecture, DEEVOLVE is capable of detecting and handling exceptions such as the failure of dependent peers. The handling of exceptions is done in strong interaction with users. At selected decision points, users are able to decide which routines need to be executed for handling an occurred exception. For the actual handling procedure, the architecture uses the same component-oriented operations as for creating the initial compositions (e.g. discover and add new service, bind two services, change parameters to a service). DEEVOLVE is accompanied by a couple of auxiliary tools for the discovery, advertisement, composition of services, as well as for handling exceptions, or the management of groups.

4. COORDINATION OF CONCURRENT DESIGN ACTIVITIES

In both research projects, different process support models are favored: the agent-based project makes use of high-level Petri net-based workflow management concepts (van der Aalst, 1998) whereas the peer-to-peer-based research project falls back on an awareness-based approach. In the following sections, these models are described. Both coordination approaches reveal pros and cons that are pointed out at the end of this chapter.

4.1 Awareness-based coordination model

The theory of awareness for regulating distributed activities has become a popular research topic in the area of Computer Supported Cooperative Work (CSCW). According to the author Dourish, awareness is an understanding of the activities of others, which provides a context for your own activity (Dourish and Bellotti, 1992). With the introduction of awareness in groupware systems, each user is equipped with mechanisms, with which he can be aware about activities of other users belonging to a common activity or a common data set. For example, if a user is modifying a shared document, a predefined number of users will be informed about all subsequent changes. For receiving notification events about status changes, a user first has to subscribe to a notification list. This kind of awareness is also called *task-oriented awareness*. Awareness can be regarded as a foundation for conflict recognition and resolution based on human cognition and consciousness. We claim that the introduction of such models is in particular reasonable for the field of networked co-operations, where the occurrence of conflicts is likely. During the further assessment of the model in section 4.3, a scenario is given that demonstrates the necessity of our model.

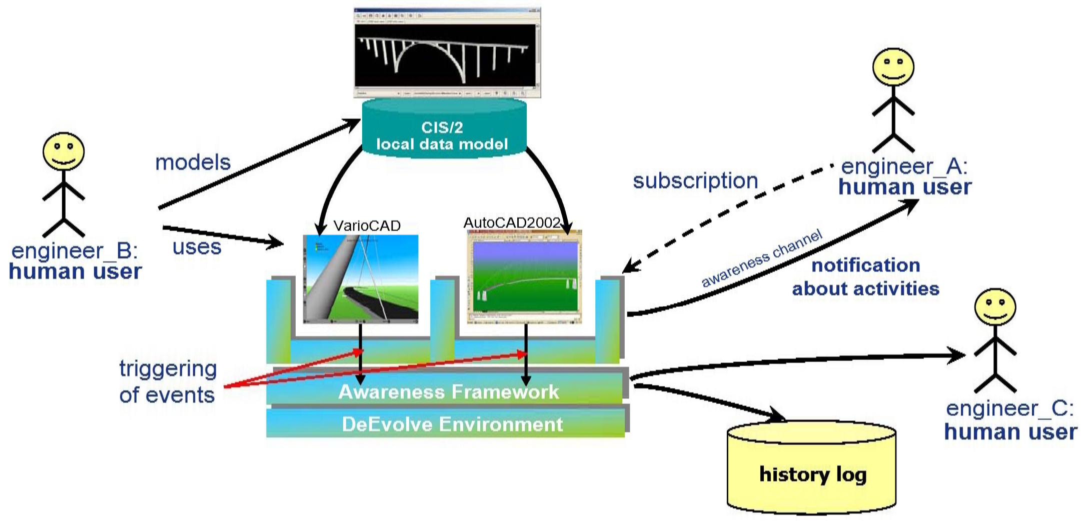

FIG. 3: The COBE AWARENESS FRAMEWORK

For the CoBE project we have developed the COBE AWARENESS FRAMEWORK (Alda et al, 2004), which realizes a decentral awareness model to coordinate the working activities within a networked co-operation. In contrast to other existing implementations of the awareness model, our system does not rely on a global server that takes over the notification of users with awareness events. With respect to the DEEVOLVE architecture, the peer-to-peer ideology has also been adopted for our awareness framework. Each peer is not only capable of acting as a publisher of awareness events, but also as subscriber who receives events from other peers through single (peer-to-peer) *awareness channels*. At any time, a peer owner (representing a partner in the co-operation) can subscribe to other partners in order to become notified about events.

The current version of the COBE AWARENESS FRAMEWORK can be used by any component-based application that is deployed in the DEEVOLVE environment (see Fig. 3). The purpose of this framework is actually to make the local behaviour of components explicit for other users. Apart from our own developments (simple CAD tool, shared whiteboards), we could also integrate sophisticated applications like AutoCAD 2002 or Visio. For extracting modeling activities within these applications, we use their provided COM-based event notification channels. These channels fire events whenever documents are modified, stored or opened. The awareness framework takes these events, interprets them and conveys them to subscribed users. In addition, the producer of awareness events can annotate auxiliary comments, documents and so on. Awareness channels themselves are published as peer services that can be located within a peer-to-peer architecture. In order to restrict the accessibility, peer services (i.e. awareness channels) can be associated to peer groups, so that only authorized partners can be informed about modeling activities.

Users obtaining information about occurred interactions are thus accomplished to derive further actions based on the sole awareness of precedent activities stemming from other users who, for instance, belong to the same co-operation. Users can either be notified directly on the screen or, given that they are unavailable, asynchronously via email. In order to avoid any violation of privacy issues of individual co-operation members, each member is capable of defining so-called *filter agents*. The purpose of these agents is to pre-select events of activities that should not be made explicit for other users. In addition, agents allow the selection of non-relevant events stemming from other members.

Any awareness event produced is stored locally in a history log. These logs can later on be browsed by new partners in order to become acquainted with the history of the project. This eases their entry especially to long-lasting projects. It also gives project managers and individual planers an overview of their completed activities and, hence, an impression on the project progress.

4.2 Workflow-based coordination model

In the Bochum project the coordination of collaborative design processes is handled by the *workflow agent*. In principle the delegation and coordination of design tasks is accomplished as follows: The workflow agent receives a process template for a given sequence of activities either from the project agent or from the project manager. Afterwards, the process template is instantiated with given boundary values. Then, the workflow agent delegates “fireable” design activities (work items) to subscribed design experts subject to the current state of the workflow until the complete workflow is finished.

The applied process support model, thereby, has to fulfil several major requirements that derive from the specific characteristics of collaborative structural design. On the one hand the process model has to support concurrency; on the other hand it has to facilitate time dependent activities e.g. a given task has to be completed at a given point in time. Moreover, the process model has to provide an opportunity to integrate any kind of resource (like product model data, engineering software e.g. for the (semi-)-automatic execution of processes, designers, etc.).

We analyzed different Petri net (PN) types with respect to the proposed requirements. As a result high-level, colored, timed, hierarchical Petri nets (HCTPN) turned out to fulfil our pre-conditions in a convenient way. In particular PNs are well suited for systems in which communication, synchronisation and resource sharing are of great importance like in the structural design domain. Another advantage of colored, timed Petri nets is that they can be described with the standardized Petri net markup language (PNML) and provide means for an intuitive graphic representation. Furthermore, it is possible to concatenate several process chains by using the more complex hierarchical colored Petri nets.

Basically, a Petri net is a graphical and mathematical modeling tool for discrete, distributed systems. The structure of a PN consists of places (the passive part of a PN, representing system states), transitions (the active part of a PN, representing events, activities, etc.), and directed arcs. Arcs connect a place to a transition and vice versa. Mathematically, the structure of a low-level PN is represented by a quadruple $PN = (P, T, A, M_0)$ where $P = \{p1,...,p_n\}$ is a finite set of places, $T = \{t1,...,t_m\}$ is a finite set of transitions, A is a finite set of arcs and M_0 is a finite set of initial markings $M_0 = \{ m_0^1,\dots, m_0^n \}$. *Markings* (or sometimes called *tokens*) are attached to places. Tokens are responsible for the execution and, thus, dynamics of the system. The current state of the modeled system (the marking) is given by the number of tokens in each place. If the system state changes at least one transitions "fires". Transitions are only allowed to fire if they are enabled, which means that all the preconditions for the activity must be fulfilled (enough tokens must be available in the input places). When a transition fires, it removes tokens from its input places and adds some at all of its output places. Fig. 4a) shows a simple, cyclic Petri net with four places and five transitions. Transition t_1 is actually able to fire for it consumes three tokens and its input set provides exactly three tokens.

The major enhancement of *colored* Petri nets compared to low-level Petri nets is that they are characterized by distinguishable, high-level tokens of different data types, i.e. places are marked by structured tokens where information is attached to them (Jensen, 1998). More complex high-level Petri nets add individuals, their changing properties and relations to the ordinary (low-level) Petri nets. In other words, tokens have an identity and arcs are labeled with variables, and the transitions may be annotated with a formula. The formula limits the conditions when the transition can fire.

In structural design timing and scheduling is important. *Timing* can be added to a Petri net in different ways. Places, transitions or even arcs may be associated with a predefined, constant or stochastic time delay for which no event/firing/movement of a token can occur. For our work it is sufficient to attribute only transitions either with a constant delay time or with an interval based delay time $I(t)=(a[t];b[t])$ where $a[t]<b[t]$ ($a[t]$ defines the static *earliest firing time* and $b[t]$ the static *latest firing time* of a transition t).

A large project work may imply thousands of different, partially interlinked activities. *Hierarchical* Petri Nets introduce a facility for building a PN out of decomposable subnets or modules. The idea behind hierarchical Petri net theory is to allow the construction of a large process model by using a number of small, manageable, design specific PNs, which are related to each other in a well-defined way. In our approach, a transition t may represent a complete subnet PN_{sub}. If t's preconditions are fulfilled, the assigned subnet PN_{sub} is invoked. Then, t pushes its markings not until PN_{sub} is finished.

Beyond their intuitive, graphical notation Petri nets potentiate the substantiated, mathematical analysis, verification and simulation of a given net. The mathematical foundation is given with the *incidence matrix* defining the static structure of a net and the *state equation* describing the dynamic behavior of net. Based on the mathematical fundamentals the analysis of a given net's dynamic (reachability graph, liveness, boundedness, etc.) and static (place or transition invariants, starvation, deadlocks, etc.) characteristics may prognosticate potential design errors and conflicts in advance.

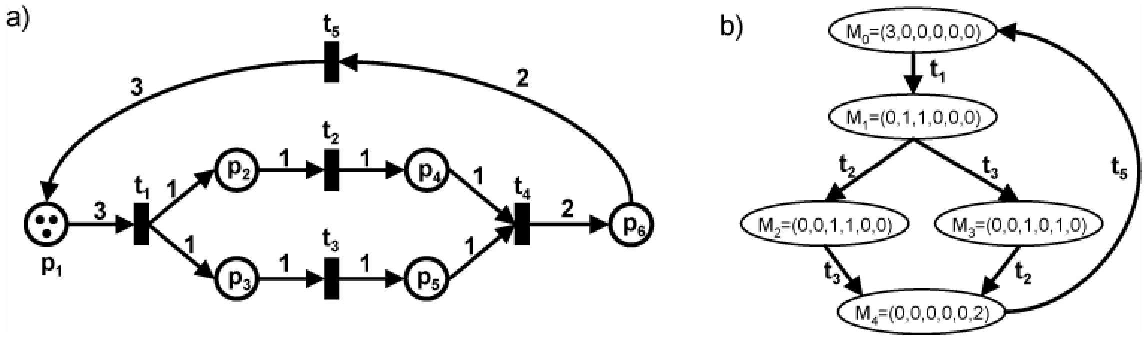

FIG. 4: Simple place/transition (P/T) Petri net with place invariant (a) and related reachability graph (b).

The implemented process support model incorporated in the workflow agent consists of basic high-level Petri net features as depicted before. For that, the workflow agent embodies a HCTPN based workflow engine that dispatches available activities proactively to appropriate designers and their personal, co-operative agents respectively. The co-operation agents, therefore, work as task manager for their principals. An activity, hereby, is a transition coupled with a specific role the design participants may exhibit or not. With that, a sophisticated and expedient allocation of activities to be executed is realized. However, beyond the role based transitions the workflow agent currently supports even more transition types: interval and delay timed transitions, subnetting transitions that instantiate specified, modularized sub-workflows automatically. Furthermore, we created an activity interface for arbitrary, (semi-) automatic application based tasks controlled by the *activity manager*. The activity manager provides the possibility to link any fireable transition to an implemented, application dependent activity with customized, advanced features e.g. for the automatic shipping of emails. Besides, arcs can be labelled with formulas that make use of global system variables designed and controlled within the *attribute manager*.

4.3 Assessment of both coordination models

Both explained coordination models exhibit individual strengths for supporting planning processes. The awareness model is an efficient way for detecting and handling any kind of unanticipated conflict that may encounter during the collaboration of different partners within a networked co-operation. By unanticipated conflicts, we suggest any kind of exceptional cases that cannot be foreseen and captured during design time. We believe that such a flexible way for handling exceptional cases is in particular important for supporting networked co-operations in civil and building engineering. This opinion can be justified by the dynamic behaviour of project structures that can be found in this area. Considering the case, for instance, where partners tend to leave and/or join a co-operation arbitrary either due to project-related circumstances or due to individual reason (e.g. insolvency). Also, partners might become unavailable whilst an important decision need to be made (e.g. in disaster cases). Such exceptional cases need to be considered but can hardly be modeled rigorously by formal process models. Recent inquires have shown that almost 80% of the time spent in building business processes is dissipated in exception management (Peltz, 2002).

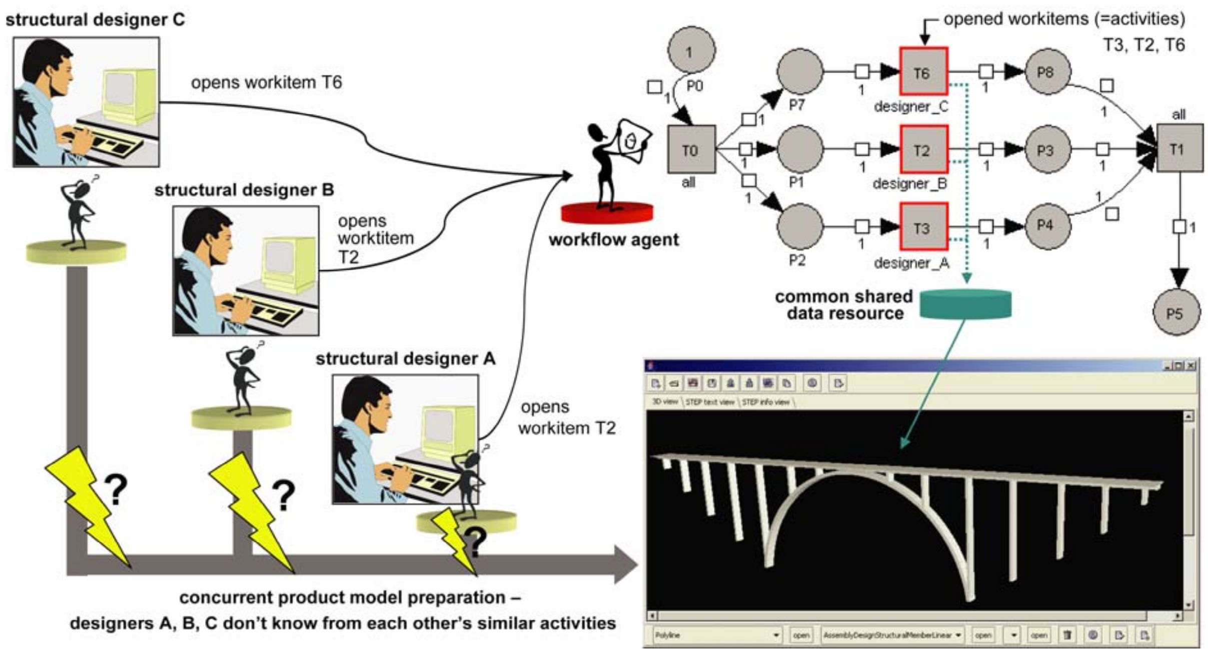

FIG. 5: Conflicting use case: designers A, B and C have to operate on the same bridge model objects concurrently.

The following example demonstrates, how conflicts in real life structural design scenarios can occur and what kind of problems naturally arise. In fact, many design activities in structural design require the preparation of (at least partial) product model data that in many cases cannot be integrated into the Petri net based process model directly. For example, in an instantiated workflow different design activities (fireable transitions/ work items) are attached to the common, shared product model objects that are controlled by the product model agent. Let us assume three of these transitions fire simultaneously and their attached work items are delegated to three different design experts (designer A, B, C) subsequently (see Fig. 5). Then, in the near future, the designers A, B and C have to operate on the same product model objects concurrently without knowing from each other's similar work items. One of the designers (designer A) may start to manipulate some product model objects in co-operation with the product model agent. Let us suppose he finishes its activities before the other designers, (designer B and C) start their intended operations. In this case, the already permitted operations of individual A can totally differ from the intended operations that designer B and/or C are willing to perform. This use case may result in severe planning conflicts. Although we could explain the evolution of this conflict scenario reasonably, it is definitely a big challenge to elicit all exceptional scenarios that could occur and to abstract from all these towards a generalized representation of the use case “handling exceptions during co-operative design”. The reason for that is the huge number of parameters one has to take into regard, which are unknown in such project settings. Typical parameters are, for instance, the number of engineers taking part as well as model-related concerns such as the granularity, the grad of accuracy, the distribution, or the availability of models.

Awareness models are therefore a much more efficient option (in terms of costs, design time) to incorporate and to handle such conflicting cases. This, say, benefit of the awareness model is in contrast a common weakness of formal process models like Petri nets. Here, every case has to be grasped at design time, that is, unanticipated exceptional cases cannot be included and handled thoroughly. The benefits of a Petri net-based workflow model as proposed in the last section is obvious, anyway. The *core* processes consisting of activities, roles, their dependencies, concurrency aspects and so on can be modeled soundly. So, the assignment of tasks, duties and responsibilities can be declared. What is more, a process model incorporating (agent-based) autonomous behaviour ensures that certain critical activities are executed instantly. This provides for the continuous flow within a given model especially at significant activity paths. The instant or guaranteeing reaction on perceived events cannot, however, be ensured in an awareness model as proposed in this work up to now. The reaction or the execution of activities is up to the mental attitude of the receiving user. Hence, conflicts – although made explicit – can simply be ignored or not recognized. Another problem of an awareness model is the lack of an initialization process. Such a process would ensure that selected partners are connected through awareness channels and become acquainted. This becomes in particular a problem in a distributed setting as conceived for the CoBE Awareness Framework. Owing to the idea of decentralization, centralized entities are missing that could otherwise take over the establishment of an awareness network consisting of linked partners. The creation of such an initialization process should not be underestimated, because many aspects need be considered, as for instance, the different project constellations and so on.

The above-illustrated scenario can be improved by deploying an awareness model that enables dependent partners to become acquainted and, from then on, to perceive each other’s activities. This leads to an integration of both, the Petri net-based workflow and the CoBE awareness models. The authors believe that such an integration constitutes the most reasonable and the most efficient way for supporting planning process in structural design. The integrated model features the advantages of both models while eliminating the apparent drawbacks that have been outlined in this section. The specific “awareness-workflow” integration is described in the next section.

5. MULTIAGENT AND PEER-TO-PEER BASED WORKFLOW-CONTROL

5.1 Preliminary work: MAS-P2P architecture

In the course of our common research activities, we have already proposed and developed an integrated Multiagent and peer-to-peer software architecture (MAS-P2P) for supporting collaborative structural design processes within networked co-operations. According to this integrated platform, not only human experts, but also software agents are capable of emitting and perceiving awareness events that correspond from planning activities. This approach decreases the probability that inconsistencies in partial models remain undetected by human actors. Undetected inconsistencies are typically caused by two reasons: (1) human actors are swamped to tackle with too much information and (2) human actors still possess the freedom how and when to react on events. The advantage of agents being able to receive awareness events is motivated by their semi-autonomous characteristic: agents are able to react immediately and autonomously on events and, accordingly, to react directly on occurred inconsistencies in partial models. On the other side, human experts are now able to perceive activities resulting from software agents that operate on a data model. This way, typical agent-oriented operations such as the autonomous consistency check on building models can be tracked by end-users. More information about the basic assumptions of this platform can be found in (Alda et al, 2004). To some extend, more information is also provided in the next section.

5.2 Fundamental integrated coordination model

Fig. 6 depicts the overall design of the integrated architecture (MAS-P2P) featuring fundamental aspects of both the peer-to-peer and the agent architecture model, respectively. Beyond, the figure points out the incorporated enhanced Petri net based workflow control mechanism and the CoBE awareness model. What one can figure out from the first view is that each participating engineer (e.g. designer A and B) is equipped with an integrated

MAS-P2P environment. The MAS part represent the agent-oriented environment (ACOS) in which an ACOS *personal software agent* is deployed. The P2P part constitutes the CoBE peer-to-peer runtime environment, in which component-based applications such as the awareness framework and arbitrary CAD software can be deployed (see also Fig. 3 for more concise presentation).

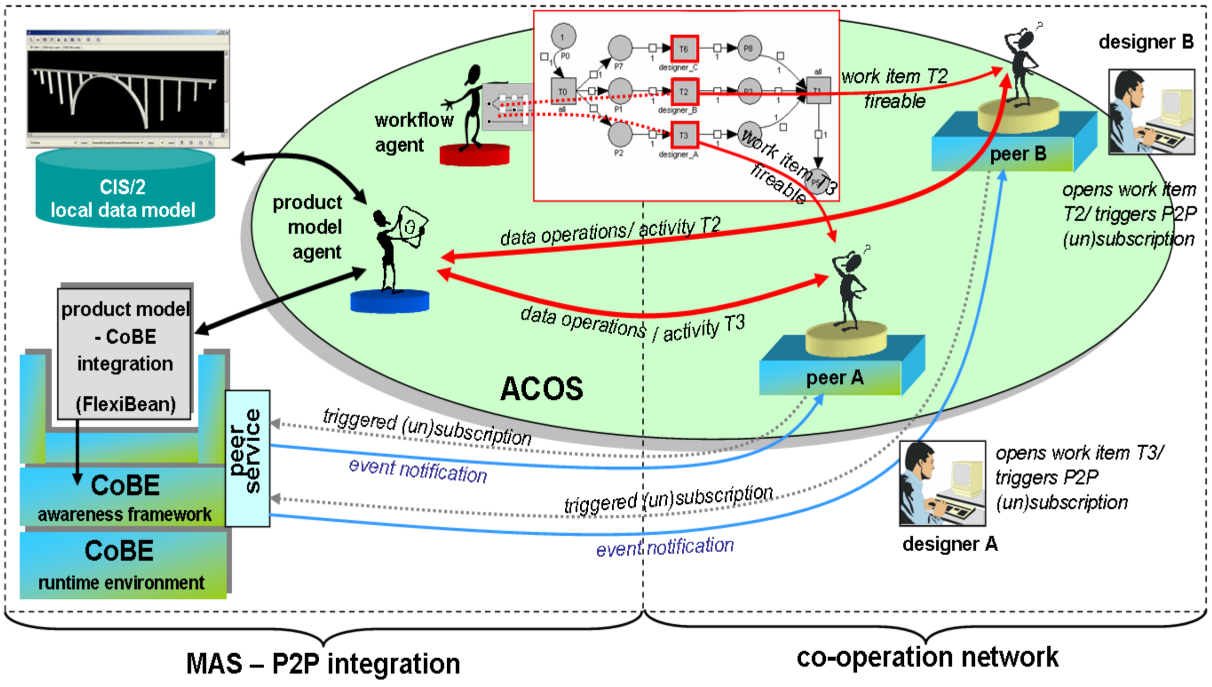

FIG. 6: Overall design of the integrated MAS and P2P architecture with new coordination model

A personal agent constitutes a kind of gateway to other agents and their offered services like the workflow agent. The workflow agent controls initialized workflows and notifies workflow subscribers about changes in the current workflow state e.g. if an activity was completed and tokens are moved. In the structural design domain, a subscriber is primarily an individual personal agent attached to a participating engineer. With that, the workflow agent is able to allocate fireable work items to authorized individual personal agents. The individual structural designers then may decide whether to open the received work item or not. Once one of the authorized designers has opened the eyed work item it is no longer fireable. An already opened work item we call activity. An activity may be cancelled or closed by its associated design expert. If closed the workflow switches to the next state, if cancelled the work item's state is reset to fireable again.

In structural engineering, many activities deal with product model operations. For that reason, we have implemented the PMConnectActivity class, a customized activity that automatically connects a personal agent to the responsible product model agent once a product model based work item is opened. As already indicated it may arise that two or more designers get connected to the same product model agent simultaneously and subsequently are operating on the same common local data model concurrently. In this case it is vital that each designer gets immediately notified when product model objects relevant to its design task have been manipulated by other participating design experts. Additionally, a designer may only be interested in particular notifications e.g. modifications of particular product model objects or parts of the overall product model. In this case a sophisticated, adjustable filter mechanism is required that accepts only relevant notification events.

To reach this goal, we enhanced the product model agent by the awareness capabilities offered by the CoBE AWARENESS FRAMEWORK. Any kind of data manipulation activities executed within the product agent by third-party agents are made explicit to all (subscribed) members within the P2P co-operation network. Any interested party involved in the co-operation, either personal agent or human expert are then allowed to subscribe to the product agent's awareness channel that emit awareness events whenever changes to a product model are made. In order to detect the awareness channels, all data operations are published as a single peer service into the co-operation network which can be located by third-parties. In order to pre-select awareness events, subscribers are able to define so-called *filter agents* that are able to examine new events before the framework transfers these to the subscriber's environment. With respect to the principle of the awareness model, conflicts can now be recognized and handled accordingly from the viewpoint of the subscribers, that is, designers.

Technically, the product model agent delegates information about the performed product model operations to a customized FLEXIBEAN component ("product model – CoBE integration" FLEXIBEAN), that is plugged into the COBE AWARENESS FRAMEWORK. The FLEXIBEAN component then conveys each received event further to the AWARENESS FRAMEWORK. The framework itself is responsible to interpret these events (e.g. by activating filters, determining subscribers, enriching and converting the event by further data) and to trigger appropriate actions such as synchronous (i.e. on-the-fly) notification, asynchronous notification via Email, or the discard of events when filtered out by filter agents. The following events are supported and can be received: adding, deleting or modifying of product model objects and state changes of product model object like *locked*, *unlocked* or finally *released* as well as events indicating that a member has become online or offline. Filters rules can be expressed by subscribers in terms of attributes from the product model and simple arithmetical operations. For instance, a design expert may only be interested in operations on the product model objects with ID #1220 to #1250. Then, the awareness framework automatically filters out events associated to the denoted objects.

Before the participating designers can receive triggered events they first have to apply for membership to the product model agent's published peer. Obviously, the individual engineers cannot exactly know when to subscribe or unsubscribe to the product model agent's peer in the course of the executing workflow. Instead, we enhanced the PMConnectActivity with an automatic subscription/ unsubscription mechanism. Every time a participating designer opens a PMConnectActivity work item not only a FIPA-ACL based connection to the relevant product model agent is established automatically but also the inherent designer peers (A and B in the example of Fig. 6) are triggered to subscribe to the to the product model agent's peer. On the other hand if the considered designer cancels or closes its PMConnectActivity the designer's peer is triggered to unsubscribe from the P2P network. The subscription of both A and B to the third-party product model agents breaks a little the idea of a decentralized awareness model as motivated in section 3.1. In fact, the designers A and B are not directly acquainted. In practical settings, however, you will often find no distinction between a product model peer and a designer peer as both the product model and the CAD application environment can be installed on the same peer. Given the scenario, that the product model is also deployed on peer A. Peer B naturally becomes a direct subscriber of A. Peer A now acts as provider and the consumer of awareness events at the same time, meeting the original idea of the awareness model of CoBE.

As we can see the P2P based co-operation network consisting of subscribed peer members dynamically changes due to the current state of the Petri net based workflow. Furthermore, the co-operation network is interconnected automatically without explicit user interactions. It has proved that this mechanism significantly helps to avoid or at least reduce intricatenesses evoked by concurrent, product model based design activities. Thereby, the MAS-P2P environment can only identify potential planning conflicts that finally have to be handled by the affected design experts. For anticipatable conflicts, that is, conflicts in which a handling routine can intuitively be determined during design time, a personal agent could be taken as the receiver of awareness events. This way, not the human expert but the personal agent acts as the subscriber to the product agent.

5.3 Example structural design scenario

The above conceived, integrated coordination model is to be demonstrated with an exemplary structural design scenario. This exemplary structure to be designed, analyzed and detailed with our integrated MAS-P2P environment is an arched, pedestrian steel bridge near the city of Dessau, Germany (see Fig. 7). The bridge's steel framework is composed of three main structural elements: (1) the steel arch, spanning 108m with a slope of 17°, (2) 15 tension rods, that are associated with the arch, and (3) the bridge deck composed of several steel panels.

Four concrete abutments (two supporting the bridge deck, two supporting the arch) support the steel structure. The four abutments themselves are founded on concrete piles.

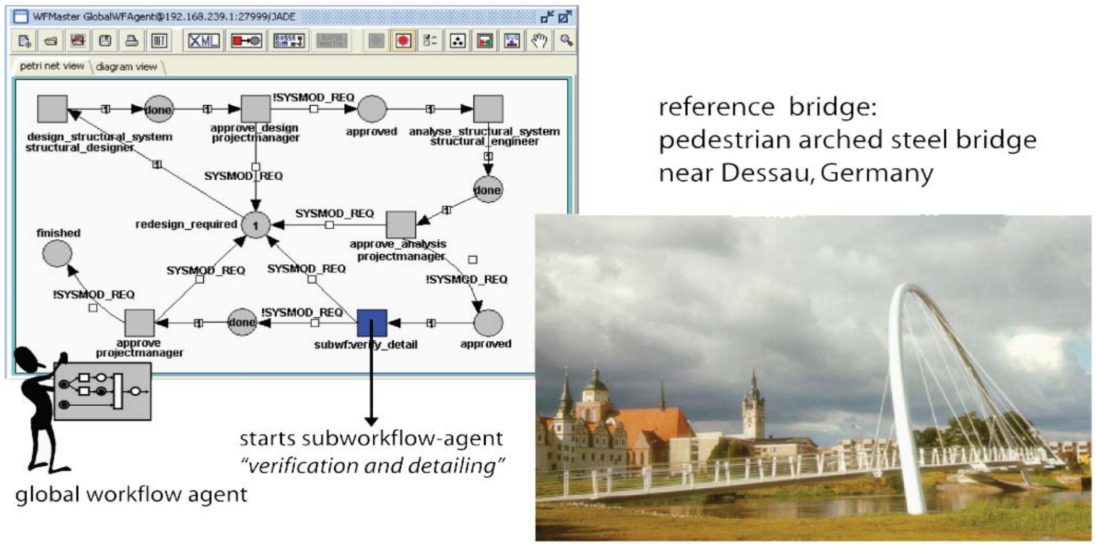
WFMaster GlobalWFAgent@192.168.239.1:27999/JADE
petri net view
diagram view
design_structural_system structural_designer
done
!SYSMOD_REQ
approve_design projectmanager
approved
analyse_structural_system structural_engineer
SYSMOD_REQ
done
redesign_required
SYSMOD_REQ
approve_analysis projectmanager
finished
SYSMOD_REQ
SYSMOD_REQ
!SYSMOD_REQ
!SYSMOD_REQ
!SYSMOD_REQ
approve projectmanager
done
subwf:verify_detail
approved
starts subworkflow-agent "verification and detailing"
global workflow agent
reference bridge: pedestrian arched steel bridge near Dessau, Germany

FIG. 7: Global workflow agent, top-level Petri net and reference structure „pedestrian arched steel bridge".

In our scenario we assume that during the structural engineering process several companies and engineers are involved with varying responsibilities and tasks (varying *roles*). Each engineer is assisted by a P2P-MAS environment comprising a personal co-operation agent and a peer. At the beginning of the structural design the project manager models the expected tasks with the PNML and, then, instantiates the global workflow agent with the delineated global process model. Fig. 8 shows the global workflow agent and the top-level process model.

The global workflow is subdivided into three major tasks (design structure, analyze structure, verify and detail structure) which have to be approved by the responsible project manager afterwards. Each transition is associated with a fixed role definition (e.g. *structural_designer*, *projectmanager*) that corresponds to the declared and managed role definitions of the project agent. Usually, every human engineer exhibits one or more roles due to the project requirements. Role affiliations may dynamically change during the project work. Every time the project manager identifies serious deficiencies of the accomplished task he sets the global workflow variable SYSMOD_REQ (= system modification required) as true such that the design task restarts (iterative design processes). Otherwise, if the task was carried out accurately SYSMOD_REQ is set to false and the next task in the process chain is ready to start. So far, unless the structural analysis is finished the particular tasks are executed sequentially. However, when the analysis has been approved in conclusion the participating structural engineers and technicians are to carry out the verification and detailing of the different structural elements *concurrently* in order to save time and reduce costs. For that, the complex verification and detailing process is transferred to another *sub*workflow agent. If the transition *subwf:verify_detail* is reached and fireable the global workflow agent automatically initiates and starts the "verification and detailing" subworkflow agent with the appropriate verification and detailing Petri net (see Fig. 8).

The subworkflow "verification and detailing" is structured into two similar process chains. One is the verification and detailing of the steel structure and the other one handles the verification and detailing of the concrete structural elements (abutments and piles). If either the verification or the detailing of the steel or concrete structural elements is deficient, a modification of the structural system design may be required. In this case, the subworkflow will stop and return to the top-level workflow where another structural redesign process cycle starts.

The modeled engineering tasks are mostly affiliated to the structural system/elements and, hence, require the application of product model agents. Taken into account the first main tasks of the global workflow, design and analysis, it is sufficient to deploy only one single CIS/2 product model agent for handling and providing the steel structure. Besides, the CIS/2 product model agent another product model agent for managing the specific, concrete structural elements (here: IFC product model agent) is required when preparing the complicated structural detailing and verification tasks. Although a product model agent is capable of notifying registered users about product model modifications immediately, notification becomes quite more complicated when applying two or more completely different product model agents concurrently. There is no common, shared notification service for several, completely different product model agents conceptualized at the moment.

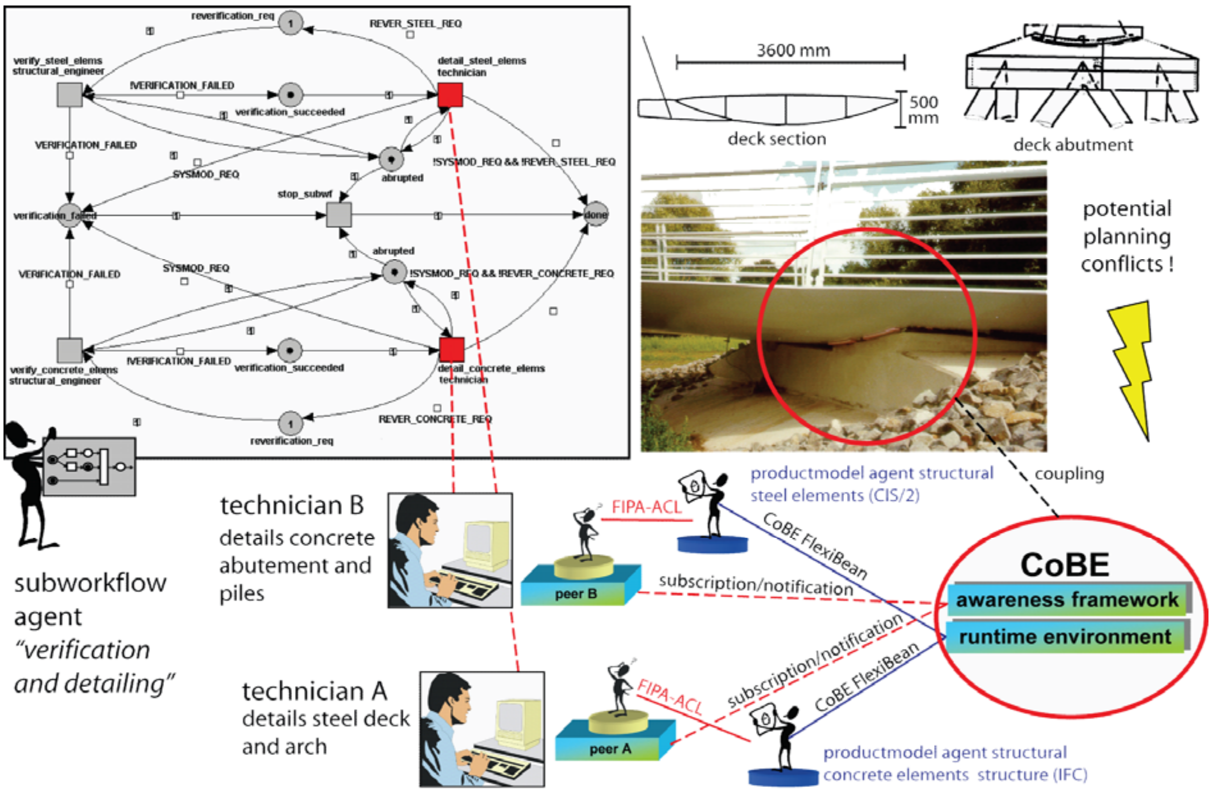
reverification_req
REVER_STEEL_REQ
verify_steel_elems structural_engineer
!VERIFICATION_FAILED
verification_succeeded
detail_steel_elems technician
VERIFICATION_FAILED
SYSMOD_REQ
abrupted
!SYSMOD_REQ && !REVER_STEEL_REQ
stop_subwf
verification_failed
done
abrupted
VERIFICATION_FAILED
SYSMOD_REQ
!SYSMOD_REQ && !REVER_CONCRETE_REQ
verify_concrete_elems structural_engineer
!VERIFICATION_FAILED
verification_succeeded
detail_concrete_elems technician
REVER_CONCRETE_REQ
reverification_req
3600 mm
500 mm
deck section
deck abutment
potential planning conflicts !
subworkflow agent "verification and detailing"
technician B
details concrete abutement and piles
technician A
details steel deck and arch
FIPA-ACL
productmodel agent structural steel elements (CIS/2)
coupling
CoBE FlexiBean
peer B
subscription/notification
CoBE
awareness framework
runtime environment
FIPA-ACL
subscription/notification
CoBE FlexiBean
peer A
productmodel agent structural concrete elements structure (IFC)

FIG. 8: Subworkflow agent „verification and detailing“ with example detailing scenario.

In our scenario for instance, we assumed that two technicians are concurrently preparing the structural detailing of the northbound, supporting node where the steel deck is attached to the concrete abutment. One of them, technician A, thereby details the structural steel elements in this section. The other one, technician B, however designs the concrete abutment below the deck in this section. While performing their modeling actions technician A interacts with the CIS/2 product model agent and technician B cooperates with the IFC product model agent. If technician A modifies for instance the anchor plate of the supporting node this may effect the reinforcement detailing of the underlying concrete abutment. For that reason, technician B has to be informed about those relevant modifications in time.

At this point, the integrated peer environment is deployed to avoid such conflicts. Every deployed product model agent publishes its capabilities as CoBE FLEXIBEAN within the CoBE runtime environment. With that, every design participant may subscribe to the COBE AWARENESS FRAMEWORK for receiving relevant product model events of *both* product model agents. To reduce the amount of listened events the two technicians A and B use the filter capabilities of the COBE AWARENESS FRAMEWORK and, thus, only receive events referring to the supporting node section. Moreover, the peers of A and B subscribe automatically to THE COBE AWARENESS FRAMEWORK when they begin preparing one of the tasks modeled within the subworkflow "verification and detailing". This is because the transitions modeled are instances of the already introduced *PMConnectActivity* which provides an automatic CoBE subscription/ unsubscription mechanism. Every time a participating designer opens the work item "verify steel elements" of "detail steel elements" not only a FIPA-ACL based connection to the CIS/2 agent is established automatically but even the inherent designer's peer is triggered to subscribe to the to the product model agent's peer. Similarly, this works with the "verify concrete elements" and "detail concrete elements" work items. If the considered technician A or B cancels or closes its PMConnectActivity the designer's peer is triggered to unsubscribe self-controlled from the P2P awareness network automatically.

6. CONCLUSIONS AND FUTURE WORK

6.1 Impacts of the work

The related works show that much effort was carried out to improve co-operation and collaboration in particular fields of structural engineering. The hereby developed software systems and subjective solution concepts mainly concentrate on a qualified perspective which unfortunately is not completely sufficient and adequate for the *holistic* and *overall* support of collaborative structural design in general. With our proposed MAS-P2P architecture, we merge multiagent and awareness based concepts and technologies to a novel, adaptable platform comprising sophisticated networked process and shared product model support. The detection of product model modifications and their potential impact on project dependent workflows is hence realized with the presented, MAS-P2P inherent coordination model. Furthermore, due to the agent and awareness-based concepts utilized our proposed MAS-P2P platform is strongly adaptable, expandable and customizable. For that reason, it is generally applicable to support any arbitrary structural design project work independently from its particular composition and the structure to be designed, computed and detailed co-operatively.

6.2 Limitations of the contribution

For demonstration purposes, we turned some of the proposed integrated architecture and coordination models into an executable prototype. This prototype is based on state-of-the-art technologies (such as the Java platform, or XML as the dominant document exchange format) and frameworks (JXTA for the peer-to-peer platform, JADE for the agent-oriented platform). Further self-made or off-the-shelf libraries were used for implementing the workflow control and the awareness framework. The integration of both approaches (peer-to-peer vs. agent-platform) resulted in an extensive prototype as both platforms are operating in parallel on a single peer. Owing to the tremendous demand on resources (memory, CPU, hard disk etc.) for our integrated approach, the overall environment of a peer becomes relatively slow which reduces the usability of other applications such as an CAD application. What is actually needed is a prudential integration that allows for a thin prototype interweaving the most essential aspects of both agent and peer-to-peer platform. So far, we have not tackled such integration.

As already pointed out in section 2 (related work), our research endeavours have mainly concentrated on presenting conceptual and technical contributions. We have almost disregarded social and human aspects (e.g. social aspects when introducing a new coordination model into a project, learning models for coping with new knowledge standards). Incorporating such issues would clearly benefit our work.

6.3 Outlook on future work

As pointed out in the previous section, we see the completion of the prototype as the major milestone. In the present paper we could already demonstrate the benefit of our approach by means of a scenario resulting from the area of structural design. We believe, however, that our holistic approach for the coordination of processes can be adopted to other fields in the area of civil and building engineering, such as facility management or risk management. Moreover, further well-known architectural styles such as grid architectures might be taken to enhance the capability of the proposed integrated architecture. Grid architectures are useful, if complex computations or simulations need to be distributed and scheduled within a networked co-operation. The success and appreciation of an integrated architecture can also be increased, if the architecture falls back on standards for service interaction, description, and discovery. Nowadays, the most prominent standards are described by the Web Services standards. New technologies such as ontologies (e.g. OWL) might be applied for establishing a common vocabulary and thus a better coordination among all partners, agents and services involved in a co-operation.

7. ACKNOWLEDGEMENTS

The research work, reported here, is currently supported by funds of the German Research Foundation (DFG), within the scope of the priority program 1103 “Networked-based co-operative planning processes in structural engineering” (PP1103). All authors gratefully acknowledge the valuable funding provided by the DFG. Moreover, the authors thank all team members of the agent systems working group for their close collaboration and beneficial teamwork over the last view years.

8. REFERENCES

Alda S. & Cremers A. B. (2004). Strategies for Component-based Self-Adaptability Model in Peer-to-Peer Architectures, *Proceedings of the 4th International Symposium on Component-based Software Engineering (CBSE7)*, Springer, Edinburgh, UK.

Alda S., Bilek J., Hartmann D. & Cremers A.B. (2004). Support of Collaborative Structural Design Processes through the Integration of Peer-to-Peer and Multiagent Architectures, *Proceedings of the 10th International Conference on Computing in Civil and Building Engineering (ICCCBE-X)*, June, Weimar, Germany.

Alda, S., Won, M. & Cremers, A.B. (2005). Group-Oriented Learning of Object-Oriented Software Methods in Construction Engineering: A Case for GeoCafe, *Proceedings of 4th Workshop Construction Information Technology in Education*. Dresden, Germany.

Bilek J. & Hartmann D. (2003). Development of an Agent-based Workbench supporting Collaborative Structural Design, *Proc. of the 20th CIB W78 Conference on Information Technology in Construction* (Amor R., editor), April, Waiheke Island, New Zealand, 39-46.

Bilek J., Alda S., Hartmann D. & Cremers A.B. (2005). Integrated Multiagent and Peer-to-Peer based Workflow-Control of Dynamic Networked Co-operations in Structural Design, *Proceedings of the 22th CIB W78 Conference on Information Technology in Construction*, July, Dresden, Germany.

Bretschneider D. & Hartmann, D. (1999). Collaborative and Concurrent Engineering in the Construction Industry, *Special Issue of Artifical Intelligence in Engineering*.

Brookshier, D., Govoni, D. & Krishnam, N. (2002). *JXTA: Java P2P Programming,* SAMS, Indianapolis, USA.

Brown A., Rezgui Y., Cooper G., Yip J. & Brandon P. (1996). Promoting Computer Integrated Construction Through the Use of Distribution Technology, *Electronic Journal of Information Technology in Construction (ITCon)*, Vol. 1, 51-66.

Cervantes H. & Hall R. S. (2005). Technical Concepts of Service Orientation, in: Stojanovic, Z. & Dahanayake, A. (Eds.) *Service-Oriented Software System Engineering: Challenges and Practices*. IDEA Group Publishing.

DFG-PP1103 (1999). *Network-based Co-operative Planning Processes in Structural Engineering, Priority Program 1103 of the German Research Foundation*, http://www.dfg-spp1103.de.

Dourish P. & Bellotti V. (1992). Awareness and Coordination in Shared Workspaces, *Proc. 4th ACM Conference on CSCW*, Toronto. Canada.

Fenves S., Flemming U., Hendrickson C., Maher M. L., Quadrel R., Terk M. & Woodbury R. (1994). *Concurrent Computer-Integrated Building Design*, PTR Prentice-Hall Inc., Eaglewood Cliffs, New Jersey, USA.

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

FIPA (2002). *Foundation for Intelligent Physical Agents (FIPA) 2002 standards*, http://www.fipa.org.

Forgber U. (1996). A Design Process Model Approach for Distributed Cooperative Work, *Proceedings 9th International Conference on Systems Research, Informatics and Cybernetics*, Baden-Baden, Germany August 17 - 20.

Fraunhofer (2005). *BSCW – Homepage*, http://bscw.fit.fraunhofer.de/.

Greb S., Meissner U. & Rüppel U. (2004). A Petri Net based Method for Distributed Process Modeling in Structural Engineering, *Proceedings of the 10th International Conference on Computing in Civil and Building Engineering (ICCCBE-X),* June, Weimar, Germany.

Grudin J. (1994). Groupware and Social Dynamics: Eight Challenges for Developers. *Communications of the ACM,* Vol. 37, No. 1, pp. 92-105.

Hawryszkiewycz I.T. & Debenham J. (1998), A Workflow System based on Agents, *Proceedings of the 9th International Conference and Workshop on Database and Expert Systems Applications (DEXA),* Vienna, Austria, 135-145.

Hughes J., King V., Rodden T., & Andersen H. (1994). Moving Out From the Control Room: Ethnography in System Design, *Proc. of the Conference on Computer Supported Cooperative Work (CSCW),* 429-439.

ICQ (2005), *ICQ Homepage*, http://www.icq.com/.

Jensen K. (1998). A brief Introduction to Colored Petri Nets, *Proceedings of the Workshop on the Applicability of Formal Models*, Aarhus, Denmark, 55-58.

Juli R. & Scherer R.J. (2002). iCSS – Ein integriertes Client-Server System für das virtuelle Planungsteam, *VDI-Berichte* 1668, VDI-Verlag, Düsseldorf, Germany.

Kalay Y.E. (1999). P3: An Integrated Environment to Support Design Collaboration, *Journal Automation in Construction*, vol. 8, no. 1, Elsevier Ltd., 37-48.

Kopp B., Zabel M. & Mandl H. (2001). *Dozentenleitfaden für die mediendidaktische Gestaltung problemorientierter virtueller Lernumgebungen an Hochschulen.* Ludwig-Maximilians-Universität, Department Psychologie, Institut für Pädagogische Psychologie. München.

Lai Y.C., Christiansson P. & Svidt K. (2002). IT in Collaborative Building Design (IT-CODE), *Proceedings of the European Conference on Information and Communication Technology Advances and Innovation in the Knowledge Society* (Rezgui Y., Ingirige B. & Aouad G., editors), Salford, UK, 323-331.

Maher M. L., Liew P. & Gero, J.S. (2003). An agent approach to data sharing in virtual worlds and CAD, in: Amor R (ed.), *Construction IT Bridging the Distance*, CIB Report 284, pp 204-213.

Maxfield J., Fernando T. & Dew P. (1995). A Distributed Virtual Environment for Concurrent Engineering, *Proc. of the Virtual Reality Annual International Symposium (VRAIS'95)*, IEEE Computer Society.

Milner R. (1991). *The polyadic pi-calculus: a Tutorial*. Technical Report ECS_LFCS_91_180, University of Edinburgh, Edinburgh, UK.

Mittrup I., Smarsly K., Hartmann D. & Bettzieche V. (2003). An Agent-based Approach to Dam Monitoring, *Proceedings of the 20th CIB/W78 Conference on Information Technology in Construction* (Amor R., editor), Waiheke Island, New Zealand, 239-246.

Mueller M.. Ruben J. & Meissner U. (2005). Dynamically Distributed and p-Adaptive FE-Simulation of Soil-Structure-Interaction based on Multi-Agent-Systems, *Proceedings of the 3rd MIT Conference on Computational Fluid and Solid Mechanics* (Bathe K.J., editor), Cambridge, USA, 993-997.

Nwana N., Ndumu D., Lee L. & Collis J. (1999). ZEUS: A Tool-Kit for Building Distributed Multi-Agent Systems, *Applied Artifical Intelligence Journal*, Vol. 13, No. 1, 129-186.

Petri C. A. (1973). Concepts of Net Theory, *Proceedings of the Symposium and Summer School Mathematical Foundations of Computer Science*. Math. Inst. of the Slovak Acad. of Sciences, High Tatras, Slovakia.

Purvis M., Savarimuthu B.T.R. & Purvis M.K. (2004). Evaluation of a Multi-agent Based Workflow Management System Modeled Using Coloured Petri Nets, *Proc. 7th Pacific Rim International Workshop on Multi-Agents (PRIMA)* (Barley M.W. & Kasabov N., editors), Auckland, New Zealand, 206-216.

Qian Z., Xue C. & Pan S. (2001). FEA agent for multidisciplinary optimization, *Structural and Multidisciplinary Optimization*, Vol. 22, No. 5, 373-383.

Reed K. A. (2002). Role of the CIMsteel Integration Standards in Automating the Erection and Surveying of Constructional Steelwork, *Proc. 19th International Symposium on Automation and Robotics in Construction (ISARC),* National Institute of Standards and Technology, Gaithersburg, Maryland, USA, 15-20.

Rosenman M.A. & Wang F.J. (2001). A Component Agent Based Open CAD System for Collaborative Design, *Journal of Automation in Construction,* Vol. 10, No. 4, 383-397.

Sauer J., Freese T. & Teschke T. (2000). Towards agent-based multi-site scheduling, *Proc. of the ECAI 2000 Workshop on the New Results in Planning, Scheduling, and Design*, Berlin, 123-130.

Scherer R.J. & Sparacello H.M. (1996). COMBI (Computer Integrated Object-oriented Model for the Building Industry), Final report of the EU-project EU/CEC ESPRIT III, project no. 6609, Technical University of Dresden, Germany.

Scherer R.J., Juli R., Reinecke W. & Wasserfuhr R. (1998). Towards a Concurrent Engineering Environment in the Building Contstruction Industry (ToCEE), *Proceedings of the XIIIth FIP Congress on Challenges for Concrete in the next Millennium*, Amsterdam, The Netherlands.

Schoen D. A. (1983). *The Reflective Practitioner: How Professionals Think in Action*, Basic Books, NY, USA.

Schwabe G., Streitz N. A. & Unland R. (2001). *CSCW-Kompendium.* Springer Publishing, Berlin.

Shirky C. (2001). Listening to Napster, in: Oram A.(ed.): *Peer-to-Peer: Harnessing the Power of Disruptive Technologies*, O'Reilly and Associates, Inc. London.

Stormer H. & Knorr K. (1998). PDA- and Agent-based Execution of Workflow Tasks, *Proc. Informatik 2001 Conference*, Vieanna, Austria.

Sun (2004). *JXTA v2.0 Protocols Specification*, Sun Microsystems Inc., http://spec.jxta.org/v2.0/.

Theiss M., Meissner U. & Rüppel U. (2004). Network-Based Fire Engineering Supported by Agents, *Proceedings of the 10th International Conference on Computing in Civil and Building Engineering (ICCCBE-X)*, June, Weimar, Germany.

Van der Aalst W. M. P. (1998). The Application of Petri-Nets to Workflow Management, *Journal of Circuits, Systems and Computers*, Vol. 8, No. 1, 21–66.

Van der Aalst, W. M. P. (1999). Formalization and Verification of Event-driven Process Chains, *Information and Software Technology*, Vol. 41, No.10, pp. 639-650.

Weiss G. (2000). *Multiagent Systems: A Modern Approach to Distributed Artificial Intelligence*, MIT Press, 2nd ed., Cambridge, Massachusetts.

Wooldridge M. J. (2002). *Introduction to Multi-Agent Systems*, John Wiley & Sons, Chichester, England.

Zarli A. & Poyet P. (1999). Distributed Architectures and Components for the Virtual Enterprises, *Proceedings of the 5th International Conference on Concurrent Enterprising (ICE'99)*, Den Haag, The Netherlands.