DISTRIBUTED MANAGEMENT OF CO-OPERATIVE DESIGN PROCESSES

Rainer Wasserfuhr [1,2], Raimar J. Scherer [2]
1 MINDBROKER Gesellschaft für interaktives Wissensmanagement, CEO, www.mindbroker.de, Dresden, Germany.
2 TU Dresden, Computeranwendung im Bauwesen, Professor, scherer@cib.bau.tu-dresden.de, Dresden, Germany.

ABSTRACT: Main reasons for inefficient project management and project delays are the lack of information about the progress of work in a team, leading to wrong versions or missing data before starting to execute work units, incorrect receivers of data and incompatible data formats. Our goal is a coherent framework for an integrated management of design data, project management, project communication and controlling, enabled by distributed IT environments. Important improvements should be achieved by enabling a distributed IT system to know about the dependencies between activities of users, permanently making information about these dependencies available to users in a transparent, simple-to-use manner and relating electronic communication in a project (e.g. messaging, file exchange and file sharing) to a shared, workflow driven activity model.
We present a model for a central repository of information about users, their roles, running projects, involved organisations, available application tools and document formats. The relevant participants of a design team, their activities and organisational dependencies are modelled in a detailed object oriented model. Project communication and access to design documents can be done in a task centred manner. The implementation smoothly integrates into existing solutions by embedding traditional messaging and collaboration features into an extensible workflow framework. The model is therefore formally elaborated in EXPRESS-C, an extension of the ISO 10303 modelling language EXPRESS for object behaviour. The model was influenced by existing EXPRESS specifications of the IFC standard and the WFMC standard.

KEYWORDS: Concurrent Engineering, IFC, XML, Project Management, Dynamic Workflow Management

1 INTRODUCTION

Important reasons for inefficient project management and project delays are, that

- involved actors are lacking information about the progress of work in a team of design experts,
- missing or wrong versions of data for starting a project activity,
- incorrect receivers of produced data, or
- incompatible data formats.

Different semi-integrated software solutions like document management systems, systems for messaging and email, workflow systems, project management software or PDM systems are also becoming a barrier to efficient, computer-mediated project management, because first they only offer parts of the required services, secondly they have partially overlapping functionality and thirdly their simultaneous application will result in duplication of data.

Improvements require a distributed IT system to know about the dependencies between activities of users. These dependencies must be made transparent for users in a simple-to-use manner, with a central repository of information about users, access rights, running projects, involved organisations, avaliable applications and document types and relate any kind of electronic communication in a project (e.g. messaging, file exchange and file sharing). These dependencies must be accessible just-in-time.

Knowledge about changes of processes is already available to different IT components: For example, email systems, document management systems or groupware components are involved in many processes, when messages are created or files are modified and exchanged. The log files and history data of these components are valuable source for a better progress monitoring, but they have to be related to each other and integrated, based on shared process models.

This paper describes *PerFlow*, the distributed prototype implementation based on process models. PerFlow is the result of ongoing research in national [10] and European research projects [11,12]. PerFlow is designed to overcome these problems of redundancy.

The basic requirements for process models and their implementation are:

- Simple user interfaces for users with different skills.
- Web based access to a project server which gives a personalized view on a project's progress
- Monitoring and refinement of the process descriptions
- Management of dependencies and schedule constraints
- Interfaces to existing project management software

One important aspect of this environment is a *role based* model for access to shared project data in distributed, cross-organisational teams of engineering experts and other actors involved in the design and building process. In the following, we will describe a role centered workflow model and it's benefits for members of such cross-organisational teams.

A detailed object oriented model describes the relevant participants of a design team, their activities and organizational dependencies. The entire project communication and the access to any type of design documents can be done in a task centred manner. The status of tasks can be determined automatically, because time constraints and dependencies between tasks are represented and maintained. The entire model is formally elaborated in the EXPRESS-C [7] language, an extension of the ISO 10303 modelling language EXPRESS [1].

2 PROJECT AND PROCESS MODELLING

2.1 Team modelling: Roles and Actors

The main conceptual dependencies are shown in figure 1.

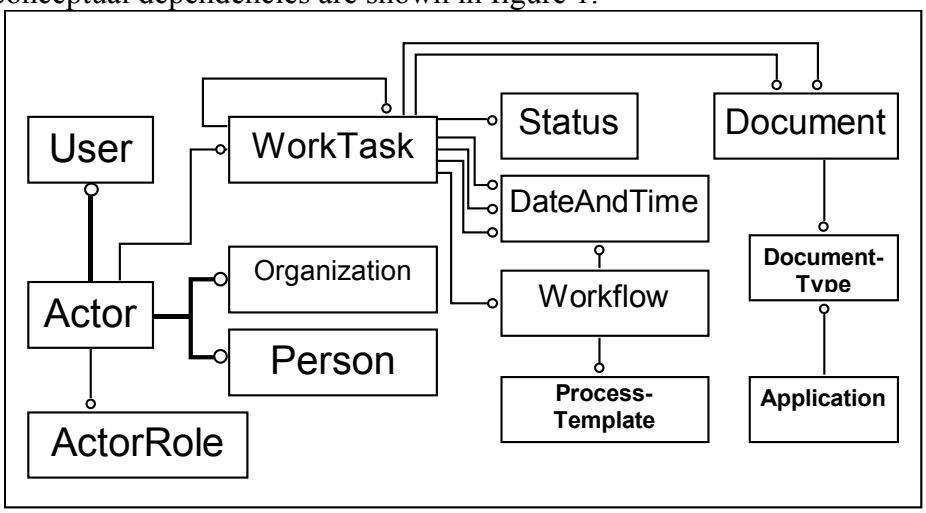

Figure 1: Actors and Roles in PerFlow

The participants of a design team, their activities and organisational dependencies are modelled in a detailed object oriented model. Each user has one or more *roles*, which describes the possible types of activities he or she can perform. Currently the following roles are supported by *CEE (Concurrent Engineering Environment)*:

- Project Manager
- Architect
- Building Owner
- Facility Manager
- Structural Engineer
- HVAC Designer
- Foundation Expert
- Construction Company
- Investor

Figure 2 shows, how the roles may be instantiated in real projects. Other roles can be added at runtime by an authorized project manager. Actors who need to work with PerFlow are represented as users with a unique login and authentication data.

The work of the members of a distributed virtual team is organised in terms of worktasks, (*tasks* for short) which are globally identifiable (like other objects of PerFlow) and linked to:

- actor roles (e.g. architect, structural engineer, etc.)
- required input (documents, product data)
- expected or delivered output (e.g. documents, views of a shared product model or even single objects of a product model)
- the time schedule of a project

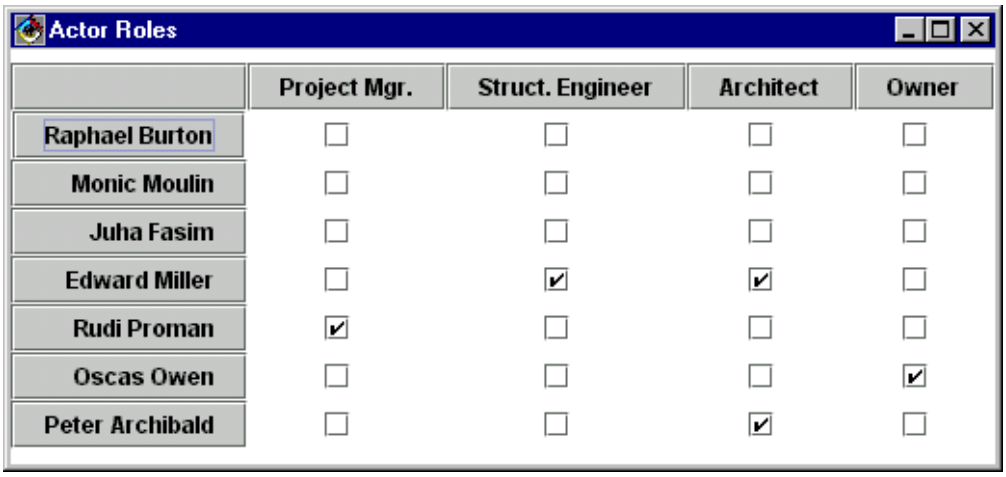

Figure 2: Actors and Roles in the CEE

In real projects, hundreds or thousands of tasks may be defined and executed by the team members. Therefore, tasks can be defined and organised by means of two additional levels: *workflows* and *process templates* (or *templates* for short, Figure 3):

- In order to manage tasks dynamically, they are grouped into workflows. Several workflows of the same type may be performed in one project. The workflows themselves can be derived from generic templates, declared by a project manager, or from predefined templates. Predefined templates can be:
- *Messages*, for internal ad-hoc communication
- *EMails*, which allow external persons to notify the actors inside the environment by standard email gateways. In this way we can embed traditional emails as special "one-step" workflows. This allows a smooth *migration* from existing organisational processes to more advanced workflow driven ways of working.
- *Conflicts*, which appeared during the design process can be registered by users on a Conflict Management Server. The status of conflict (e.g. open, solved, ignored) can be tracked by the project manager.

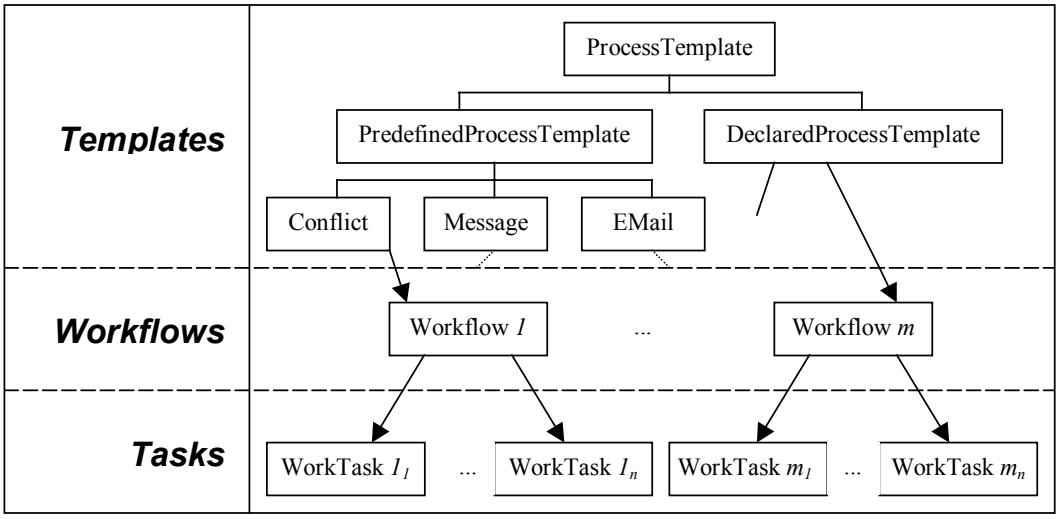

Figure 3: Worktasks, Workflows and Templates

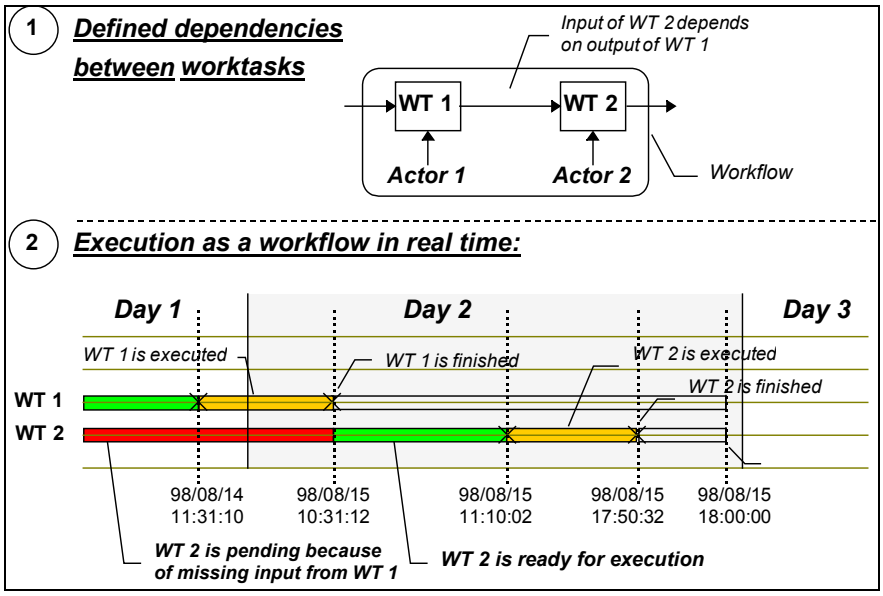

Figure 4. Workflow based scheduling and progress monitoring

By representing time constraints and dependencies between tasks, it is possible to determine the status of tasks automatically. Figure 4 shows a a simple workflow with a dependency between two tasks. A soon as the first task is finished, the second task can be started. Each task a well defined state semantics:

2.2 State transitions of a task

Initially, a task may be `suspended` or ready for execution (Figure 5). A task is suspended, if it requires additional data to be executable, e.g. the task for calculation of loads may be suspended, because data about the building geometry and placement of building elements is missing. As soon as all required input data is available, a task is `readyForExecution`.
If the actor actually starts, the internal operation `start` is performed, ensuring exclusive access to this task and changing the state to `inExecution`. Afterwards the actor may switch the state between `inExecution` and `interrupted`, as often has he wants. The workflow system can be configured to check, if the actor performs two or more tasks simultaneously, and then may present a warning. When the task is finished, the results are linked to the task (performed by the internal operation `unify`) and the state is `finished`. All tasks which are not finished, can be aborted. Abortion can be performed for the workflow which contains the task.

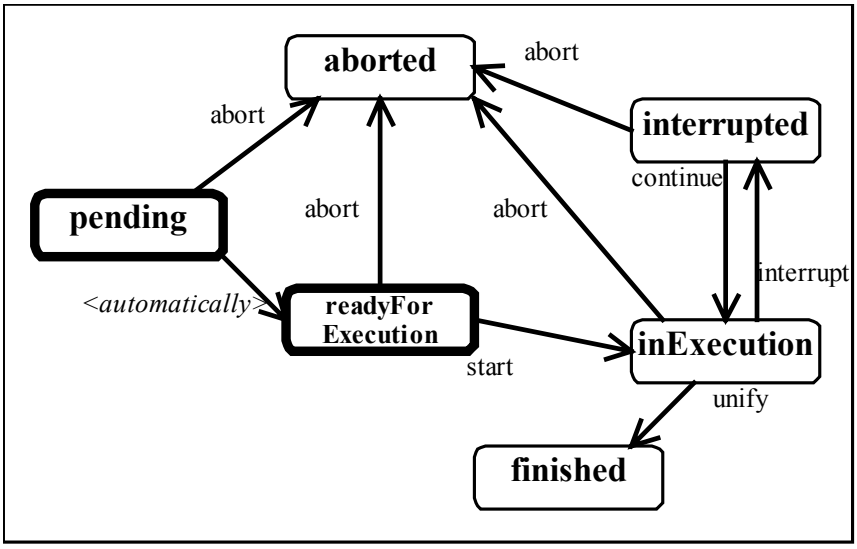

Figure 5: State Diagram of a Worktask

2.3 Workload balancing with a Personalized Worklist

For the end-user perspective, the user basically interacts with workflow clients to perform his tasks. The workflow client operates as a universal inbox, which also allows access to non workflow based messaging services (e.g. Email). All items of the universal inbox can be ordered according to a personalized priority ranking.

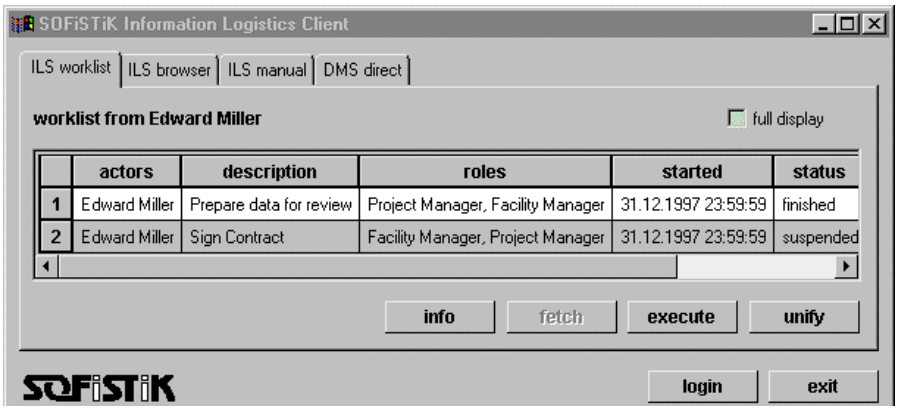

Figure 6: Worklist of the workflow client

During this overall process, the process management tool continuously updates the *worklists* for the different users, which contain exactly those tasks which are relevant for one user. The users indicate that they want to start the execution of a task and then they can select an appropriate tool, e.g. a CAD tool, a structural analysis tool or an office application. If a user finishes a worktask, he assigns his results the process management tool and the tool updates the status of all possible follow-up activities for other users.
The Workflow Client automatically retrieves the work list for the authenticated user (figure 5). The user can choose tasks he is interested to perform and the system provides to him the associated documents. Additionally he can upload any document he believes to be relevant to his task and modify them according to his option. He can store back his finished contriution to the project, signaling termination of a worktask. He can acknowledge his worktasks and define the status of his results.

The data model of PerFlow allows to describe detailed audit trails of the progress of work of a user.

2.4 Services for Advanced Project Management

A tool is provided which allows to represent and define the interactions of the actors of a construction project. The *ProcessWizard* tool was designed to support project managers in the coordination of the actors. An object oriented process definition methodology was developed to achieve a parametric description of worktask pattern, based on *process templates* [7].
Figure 7 shows a screen shot of a session with the tool in a case example. Each activity of a user role is modelled as a task (a node in the process network) and the dependencies are represented as arrows. The main window (1) of the ProcessWizard shows the tasks. By selecting a task, the properties of the task can be modified in a separate window (2) and for each task, the actors and roles can be specified in a third window (3).
With the ProcessWizard, templates may be created, edited, stored and applied. When a template is applied, a new workflow is created. The tasks of this workflow are immediately visible on the worklists of all involved actors. So after activation of a workflow, each of the nodes in the main window of figure 7 will correspond to an item on an actors' worklist as shown in figure 6. The list of all workflows can be browsed with the ProcessWizard and documented for audit trails.

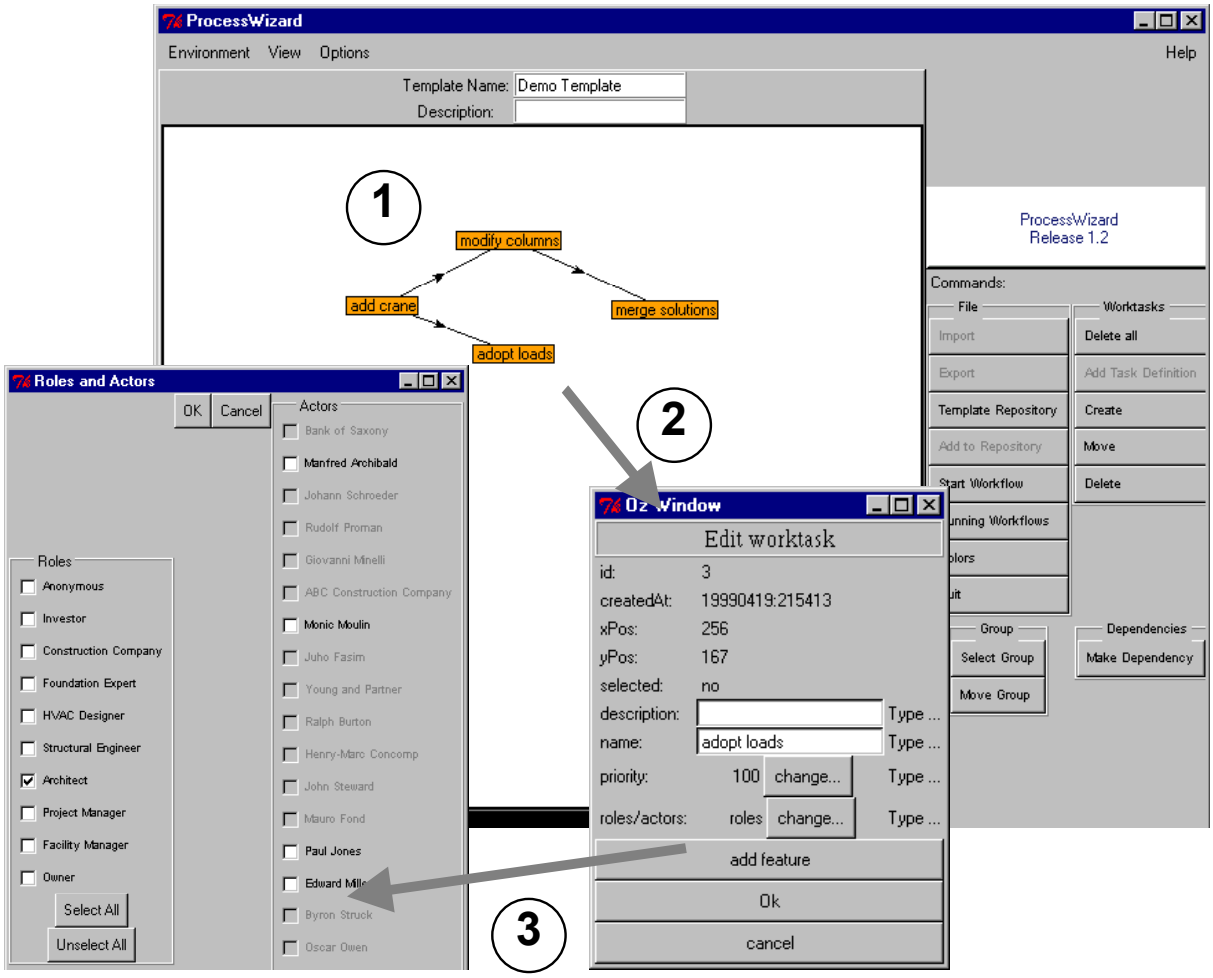

Figure 7: Process Modelling Tool

In traditional paper based project management, the typical work of the project manager has to cover:

1. defining a project schedule,
2. notifying the affected members about the schedule,
3. providing the correct data (versions) to the actors before they start their work,
4. updating the schedule, if results are received
5. writing reminders, if deadlines are ignored.

In PerFlow, many of these steps can be automated by means of the underlying workflow management: When a project manager designs a workflow, it is sufficient to define the roles for each task. The selection of the actual project actors only has to be defined once in the initial project configuration.

The definition of a workflow can be with a few mouse clicks. So in contrast to traditional workflow systems, no time-consuming Business Process Reengineering (BPR) is needed.

The visual appearance of the processes makes it easy for a project manager to rearrange a workflow and to communicate changes to the team members. The work of the project manager can focus on those cases, when rearrangements of schedules or task dependencies are necessary. A shared project calendar allows access to all time related information in a unified form of presentation. The status of tasks is displayed as a barchart, which evolves dynamically in time. Because of the integration of messaging services, the overhead for the electronic management of the process is reduced to a minimum. In PerFlow, all users have instant access to their worklist, which includes email-functionality. Therefore the motivation to use the tool in every day work is very high.

A crucial aspect for inter-organisational workflows is the confidentiallity of internal processes. In PerFlow, workflows provide a good basis for this: On the level of the inter-organizational project team, a workflow may appear as a single task of an organisation, but internally it may be refined into a workflow which affects different actors inside the organisation.

3 SYSTEM ARCHITECTURE

3.1 Distributed Architecture

The co-ordination of concurrent access of client applications to server-side data and services is achieved by a middleware layer, which is implemented as

- a *common request broker*, which is a uniform gateway to access data and services for process and workflow management and other servers, so called *server plug-ins*.
- *middleware adapters* for client applications, which can support different existing middleware standards like HTTP, Java RMI or CORBA.

The integration of PerFlow as a distributed client server architecture is ensured by a formal specification of the functionality and semantical integration of the servers. This specification consists of a set of *EXPRESS-C* specifications. The EXPRESS-C language (Staub 95) is an object oriented formal language for the definitions of classes and data structures. It is an extension of the EXPRESS language, which is part of the ISO Standard 10303, STEP (STandard for the Exchange of Product Data). Initially it was intended for use for the description of product data, but it is flexible enough to be applied in other domains.

3.2 Component Interoperability

Interoperability between different PerFlow components is achieved by a generic middleware approach. The *middleware* is a software layer which enables applications to access and modify data of distributed PerFlow components independently of the physical distribution of

the data. The system architecture is based on a generic middleware approach, defining an abstract conceptual middleware framework which can be applied to different existing technologies. The basic concepts are:
The basic mechanism for communication between different distributed *components* of PerFlow are *requests* and their *responses*. A component which sends the request is called the *client*. Each component which can accept requests is called a *server*. It is responsible for creating the response. The terms of clients and servers are only defined for a single communication event. So a server in a communication event can be client in another communication event and vice versa.
Communication events can be executed by adopting different existing middleware standards. Therefore, clients and servers have to use common so called middleware *adapters* . The current release of PerFlow supports adapters for

- the WWW centered middleware, based on the HTTP protocol extensions [W3C99],
- the Java RMI middleware [Sun98],
- CORBA ORB [OMG98].

Based in this middleware approach, it is also possible to include external legacy software into PerFlow. In the given scenario, interfaces to a document management system and to product data management system where developed and tested.

Below is a simple example of an EXPRESS-C specification of the operation `notify` for the class `IfcPerson`.

```
ENTITY IfcPerson
 SUBTYPE OF (IfcActor);
 ...
 OPERATIONS
    notify(
          message:STRING;
     );
 ...
END_ENTITY;
```

4 CONCLUSIONS

The models are targeted to be included in standardisation efforts and as a reference model for commercial applications of document management systems and workflow systems in the A/E/C sector. Currently, the adapters of the system components are based on a *late binding* approach, so that schema definitions can be changed at runtime, but strong type checking for the source code of the client application is not possible. First steps have been made to create an *early binding*, by directly compiling an EXPRESS-C specification into implementation skeletons of the target implementation language or an intermediate language independent specification layer (e.g. CORBA IDL). Further investigations will be made in accordance to ongoing standardization activities like the Java language binding in CORBA 2.2, the Java to IDL Revision Task Force of the OMG or the IDL to SDAI binding proposal of ISO STEP, Part 26.
Future research will be undertaken for a-posteriori classification of user interactions according to predefined workflow patterns and ad-hoc changes. Extensions of the implementation are possible towards Unified Messaging services (FAX, VoiceNotes or WAP) or services for procurement and digital market places.

5 ACKNOWLEDGEMENTS

This work is based on a co-operation between Dresden University of Technology and Obermeyer Planen&Beraten, the largest German company for Integrated Design in A/E/C. Obermeyer is one of the first German companies which successfully applied document management systems in large scale A/E/C projects.
The first phase of the implementation was done under the framework of the ESPRIT project ToCEE (1996-1999). The development is currently supported by a german research project and the ISTFORCE project (2000-2002, IST-Programme).

REFERENCES

1. International Organisation for Standardisation (ISO). (1998). TC184/SC4. *STEP- Standard for the Exchange of Product Data.* http://www.nist.gov/sc4.
2. Sun Technologies. (2000). *RMI - Remote Method Invocation*. http://java.sun.com/products/jdk/1.1/docs/guide/rmi/.
3. W3 Consortium. (2000). *HTTP- Hypertext Transfer Protocol*. http://www.w3.org/Protocols/.
4. International Alliance for Interoperability (IAI). (1997) *Industry Foundation Classes, Release 1.0 - End-User Guide and Specifications*, IAI Publ.,. http://www.interoperability.com
5. Object Management Group (OMG). (1995). *CORBA: The Common Object Request Broker: Architecture and Specification*, Revision 2.0. OMG Document, July 1995. http://www.omg.org/corba/corbaiiop.html .
6. Scherer R.J., Wasserfuhr R., a.o. (1997). *A Concurrent Engineering IT Environment for the Building Construction Industry.* ESPRIT Conference "Integration in Manufacturing", Dresden, Sept. 1997.
7. Staub G.. (1995). *The EXPRESS-C language.* ISO TC 184/SC4/WG5 N202 document. Available at http://www-rpk.mach.uni-karlsruhe.de/kompetenzen/Ecco/EXPRESS-C
8. Wasserfuhr R., Scherer R. J. (1997). *Process Models supporting Information Logistics in Concurrent Engineering for the Building Life Cycle.* International Conference on Concurrent Enterprising, Nottingham, Oct.1997.
9. Workflow Management Coalition. (1998). *Glossary - A Workflow Management Coalition Specification*. WfMC Technical Document.
10. Scherer R.J. (2000). iCSS – Integrated Client Server System for the Virtual Building Construction Team, Fact Sheet, German Government Research and Development Project, 2000-2002, http://cib.bau.tu-dresden.de/projekte.html
11. Scherer R.J. (1999). ToCEE - Client Server System for Concurrent Engineering, Report, Dresden, University of Technology, EU Research and Development Project, 1996-1999, http://cib.bau.tu-dresden.de/tocee
12. Scherer R.J. (2000). ISTforCE – Intelligent Services and Tools for Concurrent Engineering, Fact Sheet, EU Research and Development Project, 2000-2002, http://www.istforce.com