# Designing Standards for Open Simulation Environments in the Chemical Industries: A Computer-Supported Use-Case Approach

M. Jarke, A. Becks, J. Köller, C. Tresp, B. Braunschweig*

RWTH Aachen, Informatik V
Ahornstr. 55, 52074 Aachen, Germany
{jarke, becks,koeller}@informatik.rwth-aachen.de

*Div. Informatique Math. Appliquees
Institut Francais du Petrole
1 & 4 avenue de Bois Preau
92500 Rueil Malmaison, France
bertrand.braunschweig@ifp.fr

**Abstract.** Under the double pressures of global competition and increasing environmental awareness, the importance of high-performance simulation tools in the process industries (food, chemicals, oil, ...) is rapidly growing. However, traditional simulation environments are closed monolithic systems which are extensible only by a small group of market-leading vendors. The resulting bottlenecks in interoperability, reuse and innovationled to the CAPE-OPEN project, in which the chemical and oil industries are defining standards for a component-based approach to process simulation, in order to open up the market to smaller vendors and to facilitate rapid industrial uptake of academic research prototypes. For general systems engineering, the CAPE-OPEN standardising process has at least two interesting features: (a) it has experimented on a large scale with a distributed use case approach, following a variant of the UML (Unified Modelling Language) approach for modelling, but also linking down to programming standards based on CORBA (Common Object Request Broker Architecture) and COM (Common Object Model); (b) the management of this world-wide distributed effort has been facilitated by the use of an Internet-based workspace co-operation environment, augmented by advanced structuring and analysis methods from software engineering and computational intelligence.

## INTRODUCTION

Process simulators, often termed flowsheet simulators, are tools designed for creating mathematical models of manufacturing facilities for processing and/or transforming materials. Chemical manufacturing through continuous or batch processing, polymer processing, and oil refining are examples of such processes. Process simulators are central for designing new processes; they are also used extensively to predict behaviour of existing or proposed processes. In addition, they are applied increasingly for process control, process optimisation, and process operator training. Process simulators follow different internal architectures, e.g. block modular systems, equation based systems and simultaneous modular systems. Surveys of current and proposed approaches to simulation-based process modelling can be found in (Marquardt 1994, Jarke& Marquardt 1996).

Process models may represent either the steady-state or dynamic behaviour of a process, and both are used extensively. Dynamic models are used to describe how a process behaves over time when its processing conditions are changed, e.g. following a disturbance to feed flow rates, or altering operating conditions of one or more unit operations. They may also be used to study process start-up and shutdown, or to investigate the effect of operator mistakes or modifying a process control scheme.

Under the double pressures of global competition and increasing environmental awareness, the importance of high-performance simulation tools in the process industries (food, chemicals, oil, ...) is rapidly growing. However, new products from small vendors as well as research results in computer-aided process engineering (CAPE) only slowly find their way into industry due to the software structure of the simulators.

The process simulators that are currently used are closed monolithic applications. They are quite inflexible when it comes to integrating new components. Another drawback of this situation is that it is almost impossible to combine modules from different vendors into one single simulator. In practice such a combination is of high interest due to the limitations of individual products. Furthermore, most operating companies have developed their own simulation components reflecting the company's special needs – a resource-consuming effort. An open simulation environment would enable the integration of these in-house systems (also called legacy code) with other commercial tools in a fruitful manner.

The CAPE-OPEN project was initiated by the companies BASF, Bayer, BP, DuPont, Elf, and ICI, to solve these problems. CAPE-OPEN is co-ordinated by the French process licensing company IFP and supported by software vendors and research institutes.

CAPE-OPEN aims to provide the overall conceptual design and interface specifications for simulators which consist of an assembly of relatively large software components

For example, the entire unit operation model library of a simulator may be replaced by that from another, or just one unit operation model may be substituted or added within the existing library. Unit operation components are individual building blocks used to create models of the sequences and networks of individual chemical and physical operations that determine a manufacturing process.

Similarly, for thermodynamic methods –which are the most frequently appearing calculations in a process simulator– either the entire set may be replaced or one of the constituent methods is substituted or added. This approach creates a much more flexible environment than exists with conventional monolithic simulation systems. CAPE-OPEN also enables simulator components that originate from different types of simulators to be mixed with each other. For example, an equation based unit operation model may be combined with a sequential modular host simulator.

One example benefit is the ability to develop and test a working component which models behaviour of materials used in some particular process, and then use this validated and well tested component with different types of commercial or proprietary simulator systems. This capability provides a major advance for internal consistency in the industrial use of various types of simulation.

A second benefit is the ability to develop highly specialised, reusable models of unit operations which are not available as part of some standard set. Multi-tube reactors are a classic example. These specialised components will then be reusable in the range of models for different applications, from simulation for design to online optimisation.

**A General Framework for a Process Simulator.** Simulators differ widely in architecture and implementation but all have common functionality imposed by the underlying modelling tasks which they address. This functionality can be summarised in terms of four key 'conceptual' component types:

- **Simulator executive:** This component is the simulator's core as it controls the set-up and execution of a simulation. It is responsible for installing other components, registering them in a repository, managing interactions with users, accessing and storing data, and, finally, for reporting and analysing simulation calculations.
- **Unit operations:** These components represent physical processing unit operations (e.g. a mixer) and possibly perform specialised roles such as performing additional calculations to support process optimisation.
- **Physical properties packages:** A critical simulator functionality is the ability to model properties and behaviour of the materials which are used or created by the process. Properties and behaviour includes both thermodynamic and transport properties.
- **Numerical solvers:** This includes both the specialised mathematical methods used to evaluate the equations that describe a *unit operation* (unit solving) and the methods used to evaluate the overall flowsheet (flowsheet solving).

The CAPE-OPEN standard is defined in terms of this conceptual design, but imposes no requirements on the actual architecture and implementation of a compliant simulator. Instead it is anticipated that different simulator vendors and developers of CAPE-OPEN compliant components will initially to a large extent 'wrap' existing code such that it presents sound software interface. New implementations will be able to incorporate the standards directly into their designs. The wrapping layer and the new designs take extensively use of a component software approach like implementations OMG's *CORBA* architecture or Microsoft's *COM* reference model. This will ensure that the existing functionality of a simulator (including its user interface, unit operation library, physical property systems etc.) is protected while offering immediate access to additional functionality when necessary.

The major problem is the development of sound interfaces for already existing, sometimes quite old software components mixed together with new functionality in an integrated framework as provided by a powerful simulator executive. Without a clear and comprehensible method to define interdependencies between implemented code in different languages like FORTRAN and new methods in more modern languages like Java or C++, the task of software integration becomes futile. This observation is even amplified when different companies and vendors are involved.

From the conceptual point of view, the use of modelling techniques as provided by the *use case* approach within the *Unified Modelling Language* are necessary. Techniques for component software development well interact with those modelling techniques.

## COMPONENT-BASED SOFTWARE IN THE CHEMICAL INDUSTRY

**CORBA vs. COM (or Both?).** Approximately ten years ago, the Object Management Group (OMG) started to create standards for object-based component software (OMG Web Site). The key component is the Common Object Request Broker Architecture (CORBA). In 1994, CORBA 2.0 defined interoperability between objects in heterogeneous systems. Since then, different CORBA implementations – commercial and non-commercial ones like ORBIX and TAO – have come to market and found their impact in many areas, including finance, healthcare and telecommunications.

An alternative but proprietary approach is the one of Microsoft, called COM (Common Object Model) or DCOM (Distributed Common Object Model) for distributed environments (COM Web Site). Recently, Microsoft Corporation announced that it would hand control of its ActiveX object technology over to the a newly-formed public body called the "Active Group", formed in association with the Open Group. Active X –now often called Windows DNA– is based on Microsoft's model COM for component software. Microsoft's stated purpose in forming the Active Group is to create object based standards now even for different platforms in competition to the CORBA approach.

**Component Software in Chemical Engineering.** Since different vendors use alternatively CORBA and COM, the development of a method to combine both approaches becomes relevant. Experimental implementations in CAPE-OPEN have shown that a bridging mechanism between both technologies is possible to build for our application domain but very hard to maintain operational, mostly because both languages are moving targets. Especially COM is often affected by important changes in its internal specification. But also CORBA is evolving and will include features of COM. At the moment, CORBA allows single and multiple inheritance, whereas COM is only equipped with single inheritance but supports multiple interfaces. The latter will be integrated in the new CORBA 3 standard. Nevertheless, the design of sound interfaces for both approaches is important.

Because of these difficulties, it became important to define the CAPE-OPEN standard not just at the implementation level of COM or CORBA or both, but to have a fallback to a more stable and abstract specification. The Unified Modelling Language (UML) was selected as a candidate, but severe restrictions were agreed upon to ensure reasonable uniformity of application. As in many other projects (Weidenhaupt et al. 1998), it was quickly determined that starting directly with class definitions would not enough focus and structure the standard development process; moreover, the fact that contributors to the standard were distributed across many countries created the need to ensure mutual intuitive understanding without overly frequent face-to-face meetings. Therefore, based on research in the European CREWS project (Jarke et al. 1998), an approach heavily drawing on structured development and management of use cases via an Internet-based repository was designed. In the following section, details of this approach and the experiences gained with it during the first half of the project are reported.

**Component Based Software and Legacy Code.** Most operating companies have developed in-house process simulation software meeting their special needs. These legacy systems suffer from similar drawbacks as the simulators written by external software companies: They are monolithic, inflexible and hard to maintain. Hence, an important objective of CAPE-OPEN is to make these legacy systems compliant to the CAPE-OPEN standard. This aim is aggravated due to the use of FORTRAN for the implementation of these systems. Because FORTRAN renders only very little support for well structured software systems it was decided not to modify the legacy source code. Instead, these simulators are treated as "black-boxes" and are provided with an object oriented interface written in C++. Using these interfaces CAPE-OPEN compliant CORBA or COM objects can be created. These objects can be combined arbitrarily with other CAPE-OPEN compliant components.

This strategy was successfully applied to the IK-CAPE thermodynamics package. IK-CAPE was initially developed in FORTRAN by a consortium of six leading German chemical engineering companies. Now it is possible to use the "wrapped" IK-CAPE package as component of a CAPE-OPEN simulation environment. The interoperability of the IK-CAPE component was verified in a scenario where the package was plugged into an unit operation component using only the CAPE-OPEN standard interfaces.

## COOPERATIVE DESIGN OF COMPONENT STANDARDS VIA USE CASES

A good method to obtain formal interface descriptions as an important step in the direction of developing components for both approaches is the use cases approach. It is specific enough to capture the functionality of a simulator object that has to be mapped into a software component but general enough to be independent from the concrete technology behind a Component Model. Furthermore, it delivers a powerful tool to split up the functionality of existing monolithic simulation software into manageable components that can then be wrapped into or re-implemented as component software objects.

**Use Cases in UML.** In order to achieve clear semi-formal specifications for the different components of chemical engineering, CAPE-OPEN adopted the Unified Modelling Language (UML) (Rumbaugh et al. 1997, UML Web Site) for the set of object models. UML is seen as the way forward for CAPE-OPEN as a language for specifying, visualising, constructing, and documenting large software systems by giving the means to model complex dependencies in software modules/objects. UML presents a collection of well-known engineering practices that have proven successful in the modelling of large and complex systems (see for example (Ramackers 1996)).

In CAPE-OPEN, the application of *use cases* becomes crucial. A use case is a coherent unit of functionality provided by a system or class as manifested by sequences of messages exchanged among the system and outside interactors (called 'actors') together with actions performed by the system. An actor performs temporarily some interaction with the system being described. The use case is attempting to capture the logical interactions rather than the physical appearance of the system. A *use case model* finally consists of a collection of interrelated use cases.

**Use Cases in CAPE-OPEN.** Use cases are used in CAPE-OPEN to present the results of the requirements analysis of the major susbsystems *Physical properties*, *Unit Operation*, *Thermo* and *Numerical*. The result belongs to the phase one of the CAPE-OPEN process model. The other phases to gain running software are component design and component implementation.

The advantages coming along with this approach proved to be twofold. First, it presents in an unambiguous way what has been done in the CAPE-OPEN project. Currently, simulator vendors and third parties have begun to implement new interfaces using COM or CORBA that connect software components to simulators. Use cases describe processes and situations of using object-oriented systems. They are no formal specifications but imply requirements by presenting usage scenarios (Jacobson 1995, Jarke et al. 1998, Larman, 1998, Weidenhaupt et al. 1998). Formally, they are semi-structured narrative texts which introduce the way an external actor uses the system to complete a certain process. Within CAPE-OPEN, use cases are intended to describe the application of simulator components. In particular, they point out the usage of each component and define the order of running physical processing unit operations which represent units of plants. The main groups of use cases are concerned with a GRAPH ANALYSIS TOOL (GAT), the PHYSICAL PROPERTIES UNIT, the SOLVER USE CASES and the UNIT OPERATIONS. An example of a CAPE-OPEN use case is given below:

```
Use Case: Define Unit Report

Principle Actor: Flowsheet Builder

Description: The model builder config-
ures one or more available reports for
the unit based on Information made
public by the unit. The Builder asks the
Unit Manager to get the list of
available report formats from the unit.
If there are some available report
formats the Units Manager displays it,
so that, the user can select one of them
and ask the unit to set it to be its
output format. The unit sets this format
as its output format.
```

Close to 30 such use cases were agreed for the UNIT OPERATION interface and a similar number for PHYSICAL PROPERTIES. Up to now, 158 use cases in 4 major groups were obtained by intensive research studies from the different major chemical industry partners in the CAPE-OPEN project.

It is important to note that all of these use cases were also agreed-upon within the consortium, so they are not just informal annotations. Also beyond the individual use case, the production of use cases is a collaborative process. With the high number of partners involved, the danger of inconsistency, redundancy, and less-than-optimal mapping to software components grows.

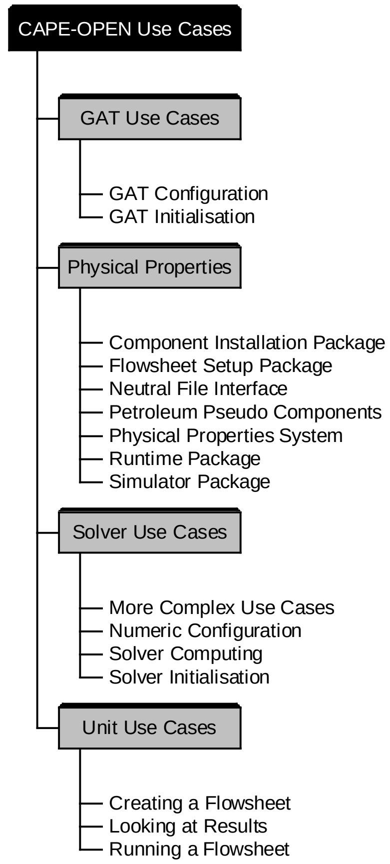


**Figure 1. The main hierarchy of Use-Cases in CAPE-OPEN**

Due to the important function of the use cases in the analysis of requirements a high quality of this step is crucial. In (Becks et al. 1998, Becks et al. 1999) a method of structuring document collections semantically is introduced which yields a map of documents stored in an collaborative environment. This method which allows a combination of knowledge-based structuring and neural network clustering techniques, has been applied to show similarities between use cases which help the people standardising components at the UML, COM and CORBA class definition level identify which use cases are relevant to their component beyond the ones from which it was explicitly derived. Figure 2 shows an example of such an automatically created use case map; it turns out that these maps (being purely created from analysing the text of the use cases) rather faithfully reconstruct the a priori structure shown in figure 1 (a good validation of the approach) but also provide some additional insights about inter-use case relationships.

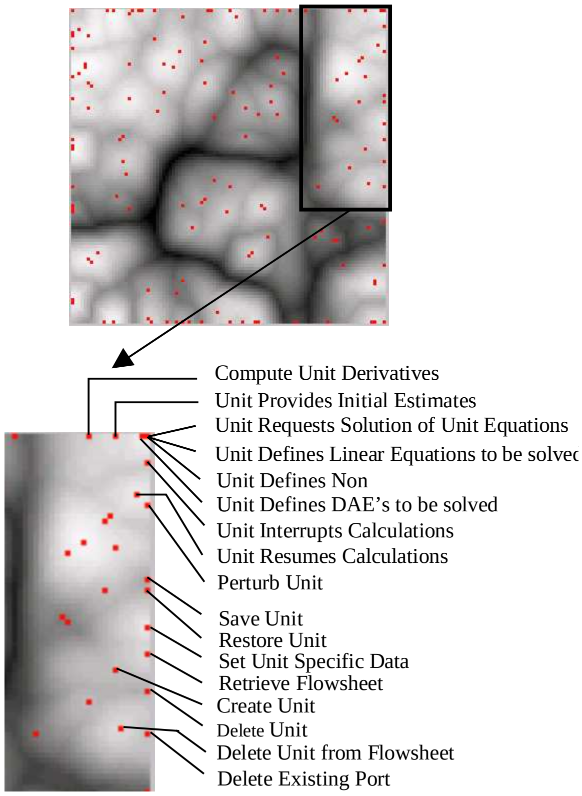


**Figure 2: Example of a use case map generated by a neural network from automated text analysis**

**Storage and Retrieval of Documents in a Collaborative Workbench.** In a large-scale world-wide effort such as CAPE-OPEN, the application of advanced groupware technology was considered imperative to maintain coherence without too many meetings. We have developed an interoperable toolkit which supports efficient group co-operation in an heterogeneous environment via the World Wide Web, in co-operation with the GMD in the European CoopWWW project (Appelt & Jarke 1998). This system, called "Basic Support for Co-operative Work" (BSCW, Bentley et al. 1997) allows the storage and retrieval of documents and enables each user to be aware of the activities of other users. In CAPE-OPEN, the content of the stored documents includes textual descriptions of proposed project aims as well as formal specifications such as use cases and component software interfaces. In addition, management information (such as plans, meeting minutes, deliverables) can be found on this server. The CAPE-OPEN BSCW server is accessible for its roughly 60 project participants through standard password-protected Internet browsers from anywhere in the world, some parts are also publicly accessible.

During the specification and development process, the BSCW system supported each phase of the CAPE-OPEN process model. Since very different specialists – like process engineers, scientist and software developers– were concerned, means for the clarification of the used terminology and an easy-to-use access to the document collection were indispensable. Specially developed structures and features like the support of different repositories – e.g., an interface repository– and glossaries simplified the conceptual phase which is often a bottleneck in the design phase of large-scale software applications. Another important functionality is the support of group management and version control of documents which is crucial since many of the different document types are changing in time.

In CAPE-OPEN, the BSCW environment was able to satisfy the demands from developers from different European countries which were concerned with the use case specification process and the following two phases from the process model. The screenshot in figure 3 offers an overview of the workspace structure. The small icons along each so-called folder or document in the workspace provide the reader with awareness of what changes have been made to this document since it was last read by him or her.

The challenging experience of using the BSCW for project work was worthwhile assessing, given the various personal background and competencies of individuals among the project members. Some of them (i) were proficient in modern software engineering tools and methods and were daily users of the web. Others (ii) were experienced users of simulation packages without any other software development skills than traditional FORTRAN, and had no prior access to the web. Others (iii) were managers of large development teams with little time available for finding out how to upload or download a document.

CAPE-OPEN started to use BSCW with a limited number of individuals chosen among the first category, collectively identified within the project as the « Methods and Tools » group. The structure of the BSCW project database was developed, and the group worked collectively on authoring a key technical document for a short period. This implied to use most of the facilities of the tool, namely, storage of any type of information, HTML tags, versioning, alert mechanisms, contact information, messaging. The result of this test phase was so encouraging that we decided to open the BSCW workspace to all project members after one month.

It took a dozen weeks of encouragement and a number of small focused training sessions before the second and third group of individuals were able to use the BSCW workspace on a regular basis. During this start-up period the question of using the BSCW as *the* means of collaboration within the project was raised many times, as opposed to ftp, exchange of documents by e-mail, and other less advanced methods. We collectively discovered, by trial and error, how to efficiently use the system for ordinary project work. After almost two years of utilisation in CAPE-OPEN, we now realize how the availability of such a tool has been a critical success factor in our development.

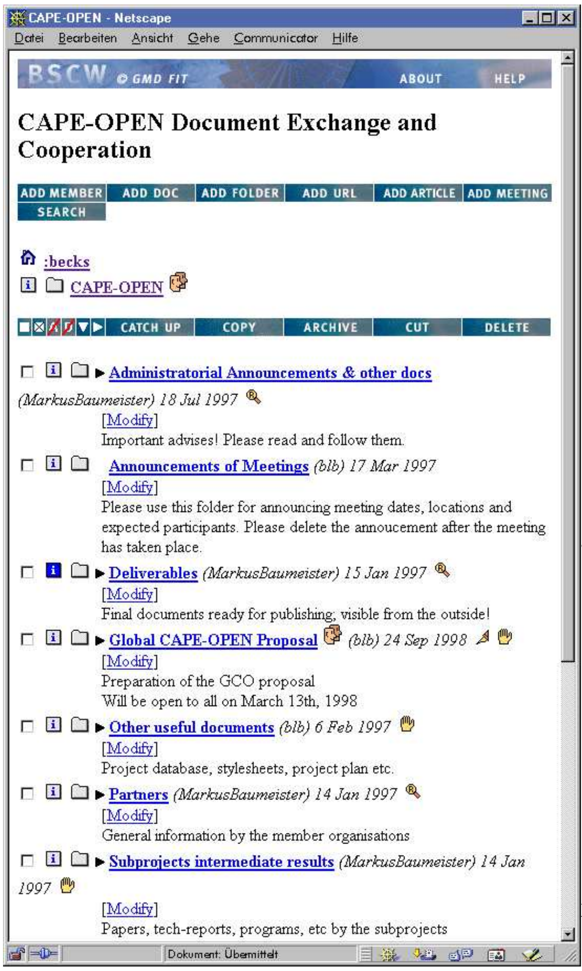


**Figure 3: Screenshot of the CAPE-OPEN BSCW workspace header page**

By early 1999, the workspace has almost 60 users, some of them frequent (e.g. every day), some occasional (e.g. once per quarter), others in between. Among the various facilities available in the current version of BSCW, the most useful ones still are those that were identified in the first place : the repositories for all project documents and work items, including conceptual documents, technical documents, UML models, interface specifications, reports, project management documents.. One of the two BSCWs is the back-end of a web server giving public access to a subset of the technical documents.

But, even with a collaborative tool like this one, we still need conventional ways of collaborations like phone discussions, conferences, and physical meetings, where we can talk about the BSCW among other matters. A tool is a tool, not a replacement for mutual interaction which is the only way to build a team spirit between geographically and mentally distant human beings.

## CONCLUSION AND OUTLOOK

In this paper, we reported experiences with a computer-supported use case approach to the development of interoperation standards for process simulators in the chemical industries. The main lessons learned can be summarised as follows:

1. Given the brittle state of interoperation standards at the implementation level and the recurring inconsistency between the two competitors, a more abstract representation is needed.
2. Given that formal UML representations are hard to understand especially during their design if the team is widely distributed in space and time, use cases proved to be the central medium of communication and agreement more than the formal representations.
3. However, even in a seemingly limited domain such as process simulation, the number of use cases quickly grows, with all the resulting risks of redundancy, inconsistency, and inadequate mapping to formal specifications and implementation standards.
4. Using Internet workspaces not just as a publishing tool but as a collaboration environment, provided the essential infrastructure for keeping all the use case and other standard information together, and make them available to whoever has the need to know, based on the access control of the BSCW server. Additional advanced analysis techniques from Computational Intelligence are proving surprisingly helpful to detect overlooked relationships between use cases that may be important during design.

As of this writing, the described environment has been in use for about 26 months, with about 4 months still to go. The project is now heavily into the process of actual standards development and, in particular, validation, followed by compliance testing of the first standard-compatible components being developed by the software vendors in the process. COM and CORBA based simulator prototypes will be presented on the ESCAPE-9 conference in June 1999.

Based on the success of this work so far, the European Commission has recently approved funding for the European part of a world-wide IMS (Intelligent Manufacturing Systems) project which aims in cooperation with additional Japanese and US partners to establish CAPE-OPEN as a global standard. Part of this will be the establishment of Cape-Open Laboratories (one of them at RWTH Aachen) which will be responsible for further developing and applying the methodology sketched here in a formal certification procedure for CAPE-OPEN compliant components and methods.

**Acknowledgments.** This work was supported in part by the Commission of the European Union under BRITE-EURAM project CAPE-OPEN and under ESPRIT Long Term Research Project CREWS. The contributions of the partners in both projects are gratefully acknowledged, especially those of Wolfgang Marquardt (RWTH Aachen) and of methods and tools coordinator Bill Johns from QuantiSci Ltd. The development of the BSCW toolkit was supported by the EU in the Telematics Application project CoopWWW, the use case analysis tool by the Deutsche Forschungsgemeinschaft in its doctoral programme on Informatics and Engineering at RWTH Aachen and in Aachen's Collaborative Research Centre IMPROVE.

## REFERENCES


Appelt, W., Jarke, M. Final Report of Project TE 2003 – CoopWWW. Aachener Informatik Berichte, March 1998.

Becks, A., Sklorz, S., Tresp, C. Semantic Structuring and Visual Querying of Document Abstracts in Digital Libraries. *Proceedings of the 2nd European Conference on Research and Advanced Technology for Digital Libraries*, Heraklion, Greece, September 1998, pp.443-458.

Becks, Andreas; Sklorz, Stefan, Jarke, Matthias. Document Maps: Semantic Structuring of Technical Document Collections. Crews Report 99-05, RWTH Aachen, 1999, http://SunSITE.Informatik.RWTH-Aachen.DE/CREWS/reports.htm

Bentley, R., Appelt, W., Busbach. U., Hinrichs, E., Kerr, D., Sikkel, S., Trevor, J. and Woetzel, G., Basic Support for Cooperative Work on the World Wide Web, *International Journal of Human-Computer Studies* 46(6): Special issue on Innovative Applications of the World Wide Web, 1997

COM Web Site: http://www.microsoft.com/com/default.asp

Jacobsen, I. The use case construct in object-oriented software engineering. In J.M. Carroll (ed.): *Scenario-Based Design: Envisioning Work and Technology in Systems Development*, Wiley & Sons 1995, 309-336.

Jarke, M., Marquardt, W. Design and evaluation of computer-aided process modeling tools. In Davis/Stephanopoulos/ Venkatsubramanian (eds.): *International Conference on Intelligent Systems in Process Engineering* (Snowmass, Co, July 1995), AIChE Symposium Series, vol. 92, 1996, 97-109.

Jarke, M., Bui, X.T., Carroll, J.M. Scenario management: an interdisciplinary approach. *Requirements Engineering Journal 3*, 3 (1998).

Larman, C., Applying UML and Patterns: An Introduction to Object-Oriented Analysis and Design,Prentice Hall, 1998

Marquardt, W. Trends in Computer-Aided Process *Modeling. Computers and Chemical Engineering*, 1995.

OMG Web Site: http://www.omg.org/

Ramackers, G. and Clegg, D., "Extended Use Cases and Business Objects for BPR," *ObjectWorld* UK '96, 1996.

Jim Rumbaugh, Ivar Jacobsen, and Grady Booch, *Unified Modeling Language Reference Manual*, Addison Wesley, 1997.

UML Web Site: http://www.rational.com/uml

Weidenhaupt, K., Pohl, K., Jarke, M., Haumer, P. Scenario usage in software development: current practice. Special Section on Best Papers from ICRE 98, *IEEE Software*, March 1998, 34-45.


## BIOGRAPHY

Matthias Jarke is professor of Information Systems and Chairman of the Computer Science Department at Aachen University of Technology (RWTH Aachen), Germany. He is coordinator of ESPRIT Long Term Research Project CREWS (Cooperative Requirements Engineering With Scenarios) and Chief-Editor of the journal, Information Systems. In 1998, he has been guest editor of three special issues related to the topics of INCOSE, on Requirements Tracing (Communications of the ACM) and Scenario Management (IEEE Transactions on Software Engineering, Requirements Engineering Journal). Jarke got his degrees at Hamburg University, Germany, and served on the faculties of New York University and Passau University prior to joining Aachen in 1991. His main research interest is information systems support for complex engineering processes.

Andreas Becks received his diploma degree in Computer Science from University of Dortmund in 1997. Currently, he is a PhD-student in the graduate school 'Informatics and Engineering' at RWTH Aachen. His research interests include information retrieval, intelligent human-computer interfaces, knowledge management, and CASE tools.

Jörg Köller received his diploma degree in Computer Science from RWTH Aachen in 1997. He is now a PhD student at the Information Systems Chair in Aachen's Department of Computer Science. He works in the CAPE-OPEN project and his research focuses on component-based software and formal semantics specification and analysis.

Christopher Tresp obtained his diploma degree in Computer Science from University of Dortmund in 1994. After that, he was a PhD-student in the graduate school 'Informatics and Engineering' at RWTH Aachen. Recently, he started to work for the Bayer AG, one of the leading chemical-pharmaceutical companies. His research interests encompass fuzzy knowledge representation, pattern recognition and different middleware approaches.

Bertrand Braunschweig is a principal research engineer within the Computer Science and Applied Mathematics Department of Institut Français du Pétrole, Rueil Malmaison, France. He holds a "diplôme d'ingénieur" in information technology from IIE-CNAM, 1977, and a PhD in Computer Science from Université Paris-Dauphine, 1998. Before joining IFP, Dr. Braunschweig worked 12 years for Elf Aquitaine where he managed and developed dynamic simulation and knowledge-based systems. Within IFP, he has been AI and Statistics group leader since 1989, and is currently coordinator for the CAPE-OPEN and Global CAPE-OPEN projects. B. Braunschweig is also president of the French AI Society since 1998.