ELSEVIER

Computers and Chemical Engineering 27 (2003) 175–197

Computers & Chemical Engineering

www.elsevier.com/locate/compchemeng

Tool support for the management of design processes in chemical engineering☆

Manfred Nagl [a], Bernhard Westfechtel [a,*], Ralph Schneider [b]

[a] RWTH Aachen, Lehrstuhl für Informatik III, D-52056 Aachen, Germany
[b] RWTH Aachen, Lehrstuhl für Prozesstechnik, D-52056 Aachen, Germany

Received 26 November 2001; received in revised form 8 June 2002; accepted 7 August 2002

Abstract

Design processes in chemical engineering are hard to support. In particular, this applies to conceptual design and basic engineering, in which the fundamental decisions concerning the plant design are performed. The design process is highly creative, many design alternatives are explored, and both unexpected and planned feedback occurs frequently. As a consequence, it is inherently difficult to manage design processes, i.e. to coordinate the effort of experts working on tasks such as creation of flow diagrams, steady-state and dynamic simulations, etc. On the other hand, proper management is crucial because of the large economic impact of the performed design decisions. We present a management system which takes the difficulties mentioned above into account by supporting the coordination of dynamic design processes. The management system equally covers products, activities, and resources, and their mutual relationships. With respect to coverage and integration, and with respect to the dynamics of design processes, the functionality of the management system goes considerably beyond commercial project, document, and workflow management systems.

Keywords: Design process; Project management; Document management; Workflow management; Polyamide6

1. Introduction and motivation

Design processes in engineering disciplines, like chemical engineering, often deliver good results, but sometimes perform less effective than they could. Reasons, among others, are that neither the process is explicitly and clearly structured nor its complex result. Especially, the experience of designers is not explicitly gathered and, therefore, cannot be used, the many mutual relationships between parts of the design product are neither explicitly stored nor maintained, designers may interpret the design in different ways, and the management of a project is not given a clear view about the state of the project at a certain time.

Correspondingly, there is a *lack* of *semantic support* by current *tools* for collaborative design processes, carried out by different persons, with different roles, on different sites, eventually in different companies, which altogether build up or maintain the complex product of a design process. Moreover, there are lot of *gaps* with respect to tool support in a collaborative design process. As a consequence, the vision of an integrated design environment providing high-level tools has been realized only to a limited extent.

The *Collaborative Research Center 476 IMPROVE* (Nagl & Westfechtel, 1998; Marquardt & Nagl, 1999), is an integrated project staffed by chemical engineers, plastic engineers, ergonomics researchers, and different groups from computer science (software engineering, information systems, communication systems) with the aim of solving the problems described above. The project concentrates on the early phases of process engineering, namely conceptual process design and basic

☆ This work was carried out in the Collaborative Research Center 476 IMPROVE, which is funded by the Deutsche Forschungsgemeinschaft (DFG).

* Corresponding author

E-mail addresses: nagl@i3.informatik.rwth-aachen.de (M. Nagl), bernhard@i3.informatik.rwth-aachen.de (B. Westfechtel), schneider@lfpt.rwth-aachen.de (R. Schneider).

0098-1354/02/$ - see front matter
PII: S0098-1354(02)00164-3

engineering, the tasks of which are to structure, to simulate, and to evaluate a plant design under different perspectives. This part of the overall design process is especially challenging from a research perspective (many creative decisions, permanent changes, study of variants etc.).

With respect to the development of an integrated design environment, IMPROVE follows a mixed *top–down/bottom–up* approach (Fig. 1). Bottom–up means that existing tools and platforms are re-used as far as possible (grey regions). For example, we make use of existing tools for performing steady-state or dynamic simulations. To further improve the functionality offered to the designers, new tools and services are added. The tools are designed in such a way that they fit into the overall architecture (top–down approach).

Within the IMPROVE project, there are four subprojects which address improved tool support in different functional areas. Within these subprojects, tool services are developed which may be used for the implementation of design tools with innovative functionality (Fig. 1(a)–(d)):

(a) Fine-grained *process support tools* (Pohl et al., 1999) aim at supporting designers in their interaction with design tools by process fragments which (partly) automate sequences of command invocations.

(b) *Incremental integration tools* (Becker, Haase, Westfechtel, & Wilhelms, 2002) assist designers in keeping inter-dependent design data consistent with each other (e.g., flow diagrams and simulation models).

(c) Using *multi-media communication tools* (Schüppen, Trossen, & Wallbaum, 2002), designers may discuss and resolve design problems in joint working sessions (conferences).

(d) *Reactive management tools* Westfechtel, 1999 assist in managing design processes at a more coarse-grained level than the process support tools mentioned above.

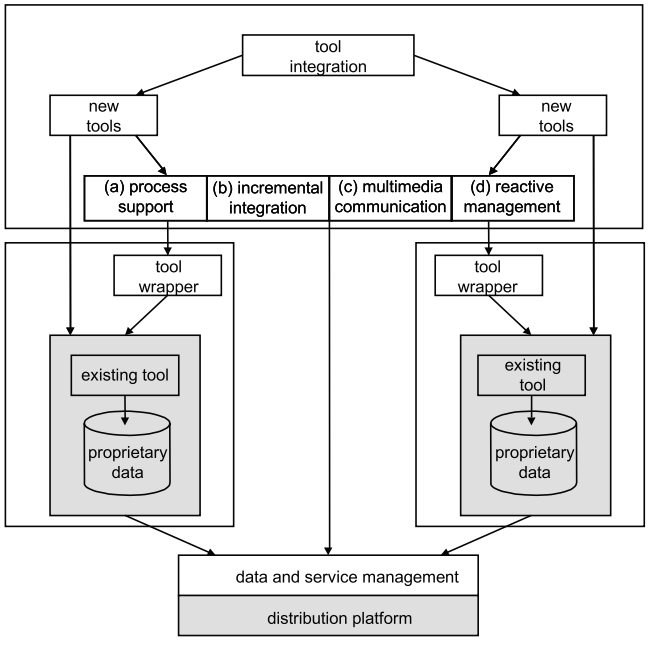

Fig. 1. Mixed tool integration approach (coarse architectural sketch).

These *new functionalities* work *synergetically together*. To take one example, let us regard the change of a flow diagram, for which process support (a) can be used. The dependent tasks are determined by reactive management (d). For changing the dependent documents (e.g., a simulator input description), integration tools (b) are applied. During the change subprocess spontaneous multimedia communication is used as well as a multimedia conference (c) at the end of the subprocess.

In this paper we *concentrate* on tool support for the *management* of *design processes* in chemical engineering (i.e. (d) in Fig. 1). Management, thereby, is restricted to the *coordination* of a design project. So, no strategic or psychological aspects are discussed. However, all aspects of coordination (products, activities, and resources) are seen as being tightly integrated. Since the design process changes permanently, coordination has to react accordingly (dynamics). Management is supported by a collection of tools which constitute a reactive management system. This system is called *A*daptable and *H*uman-Centered *E*nvironment for the M*A*nagement of *D*evelopment Processes (*AHEAD* Jäger, Schleicher, & Westfechtel, 1999a; Westfechtel, 1999).

Management in the sense of this paper is closely *related* to *major design decisions* of a chief designer. So, e.g., choosing one of different variants for a part of the plant induces a specific form of the design result (how many and which documents), design process (which design tasks) and corresponding required resources. Conversely, a limit in the design process' costs influences how intensively design variants can be studied. So, both levels, major design decisions on the technical level and coordination of the design process on the managerial level, are mutually dependent.

The rest of this *paper* is *structured* as follows: In Section 2, we elaborate on the features of design processes in chemical engineering. We also introduce a case study, namely the design of a plant for polyamide6. In Section 3, we discuss the management of design processes and the current state of the art of tool support. We demonstrate that commercial tools for workflow, project, or document management suffer from several shortcomings, e.g., limited support for dynamically evolving design processes. In Section 4, we present our management system (AHEAD) and explain in what respects it goes beyond the functionality of current commercial systems. In Section 5, we demonstrate its use by applying it to the case study introduced in Section 2. While the management system incorporates novel

functionality, it is a research prototype which cannot be immediately applied to industrial practice. Therefore, we indicate different ways of technology transfer in Section 6. Finally, we summarize our contributions and describe current and future work in Section 7.

2. Design processes in chemical engineering

From an economic point of view the early phases of design processes, namely conceptual process design and basic engineering, are worth considering. The decisions made in these phases have a great impact on the later ones. McGuire and Jones (1989) report that up to 80% of the capital costs of a plant are determined in conceptual process design. The importance of design processes in (chemical) engineering and the need for improving and modeling them has been stated in the literature by several authors (e.g., Mostow, 1985; Bañares-Alcántara, 1995; Ponton, 1995; Westerberg, Subrahmanian, Reich, Konda, & n-dim group, 1997).

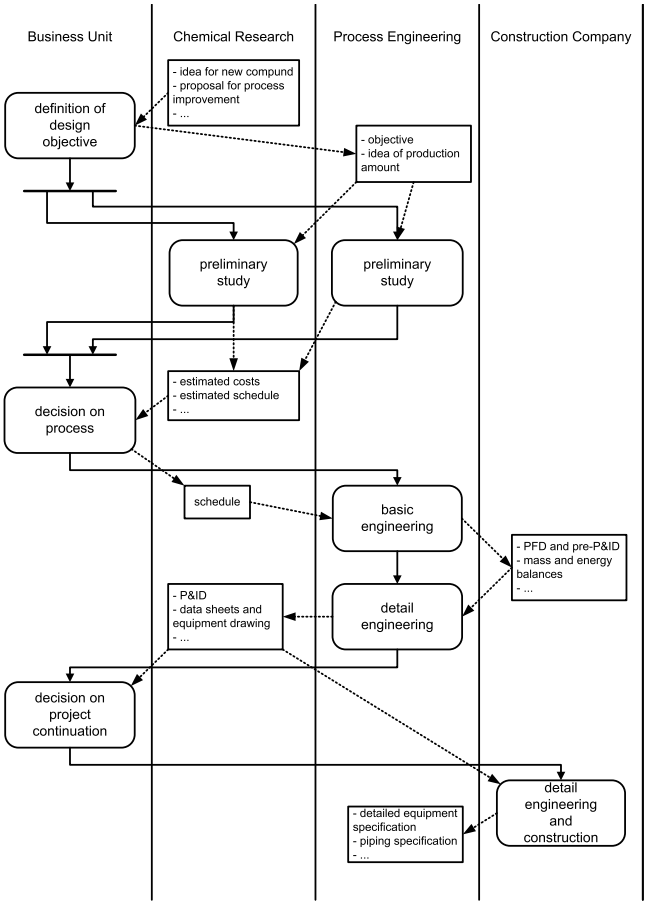

Fig. 2. Life cycle of a chemical design process.

2.1. Characteristics of design processes

The life cycle of a chemical design process ranges from the definition of a design objective to the construction of the plant as shown in Fig. 2 (developed within the Global CAPE-OPEN project Braunschweig & Gani, 2002). Besides the performed *activities* (e.g., basic engineering), which are assigned to roles or departments (*resources*), the needed and produced information (*products*) characterize the design process and are represented in the figure. The temporal order of the activities is given by the control flow (solid lines), the flow of information by the dashed lines.

Our management system aims at supporting the engineering design process (namely the activities from the start of the project to the ones including the detailed process design) with special consideration for the characteristics of a (chemical) engineering design process:

- *Designers:* The designers have different backgrounds and work together in a relatively small, but interdisciplinary team (up to 10 people), lead by one technical project manager.
- *Location:* The design team can be located in different departments and enterprises, distributed across different sites all over the world.
- *Duration:* Chemical engineering design projects differ in their duration, depending on the goal of the design project (e.g., design of a complete new plant, or just a retrofit). They last from several months up to many years.
- *Creativity:* Design processes include highly creative work processes. They are ill-defined and very complex, therefore very difficult to plan in advance.
- *Iterations:* There are a lot of iterations in a design process. Many activities have to be performed several times at different levels of detail (granularity) depending on the available information.
- *Feedback:* The activities performed during a design process are highly interconnected. Therefore changes made in a later phase of the design process can lead to unwanted feedback to activities already performed earlier during process design.
- *Documentation:* The exchange of information between the designers is mainly done during project meetings and by the exchange of email and paper. There is almost no reuse of successful problem solutions of previous design projects due to the difficulties in exchanging and using consistent information as well as missing or bad documentation.
- *Alternatives*: During a design process many different alternatives are created. The detailed investigation of each alternative may lead to totally different design processes.

- *Software tools*: A multiplicity of software tools is used by the designers. These tools often have incompatible data formats which make the exchange of information very difficult or even impossible. Large amounts of data and documents are produced by the software tools which have to be managed.
- *Design product:* The product of a design process in chemical engineering is the plant design. Chemical plants are unique and no bulk products.

In particular, these characteristics imply that design processes cannot be planned in detail in advance because they are dynamically changing due to the highly creative character of the processes and the interdependencies between the different activities and products. Effective support for managing design processes can only be enabled by the combined consideration of all these aspects.

2.2. Polyamide6 case study

The case study given here serves different purposes. First of all, we want to understand the workflow of industrial design processes in order to identify weak points and define requirements for the development of new tool functionalities respectively new tools. This case study can therefore be seen as a guideline for our tool design process. It is a common basis for our work in the IMPROVE project. All tools developed in IMPROVE are tested in the context of the case study. Furthermore, these tools are integrated in a common prototype demonstrating the interactions between the different tools and their support functionalities. Following this working procedure it is possible to evaluate whether the tools really fulfill the defined requirements and contribute significantly to an improvement of design processes in chemical engineering.

In order to demonstrate and evaluate the functionalities of the management system, a realistic *case study* is needed. Since such a case study is not available in the literature, we developed one by ourselves.

First of all we tried to record and structure design processes in chemical engineering at a very coarse level (independently from a concrete example) based on literature (Rudd, Powers, & Siirola, 1973; Douglas, 1988; Biegler, Grossmann, & Westerberg, 1997; Blass, 1997), the PIEBASE activity model PIEBASE Working Group 2, (1998), and own experiences. Furthermore, we developed a process for the production of *polyamide6* (nylon6) fulfilling a specified design task. During this development we observed ourselves and recorded our activities in the form of activity diagrams. In parallel to this we conducted five interviews with project managers of an industrial partner. In these interviews the interviewee had to remember in the sense of a case study approach (Yin, 1984) a past design process. These three elements (literature, self observation, and interviews) form the basis on which our case study is built.

The case study describes the design process of a chemical plant for the production of polyamide6 including the polymer compounding and post-processing. It focuses on the workflow, the people involved and the tools used together with their interactions. The task given in this case study is to design a plant for producing 40 000 tons polyamide6 per year with a given product quality. Polyamide6 is produced by the polymerization of ε-caprolactam. There are two possible reaction mechanisms, the hydrolytic and the anionic polymerization (Kohan, 1995). Since the anionic polymerization is mainly used for special polymers, this case study focuses on the *hydrolytic polymerization*, which is also more often applied industrially. This polymerization consists of three single reaction steps: ring opening of ε-caprolactam, poly-condensation, and poly-addition. The case study covers the design of the reaction and separation system as well as the extrusion. A block flow diagram representing the continuous polymerization of polyamide6 is shown in Fig. 3.

There are different kinds of *reactors* that can be used for the polymerization: sequences of two or more tank reactors or plug flow reactors (Gerrens, 1981) and a special reactor developed for the polyamide6 production, the VK-tube (Deutsches Patentamt, 1969). The polymer melt at the outlet of the reactor contains monomer, oligomers, and water, which must be removed in order to meet the required product quality. Therefore, a *separation* is needed. Two separation mechanisms can be used here: evaporation in a wiped-film-evaporator or extraction with water to remove ε-caprolactam with a successive drying step (Kohan, 1995). Polymer post-processing is done within an extruder: additives, fillers, and fibers are added in order to meet the specified product qualities. Within the extruder, an additional polymerization of the melt and the degassing of volatile components are possible.

For all process units, mathematical models are developed and used for simulation within different simulators. Because the design of each distinct process unit requires very specific knowledge, each block is studied in detail by different experts in our case study.

The different alternatives for reaction, separation, and extrusion lead to several alternative processes for the polyamide6 production. In Fig. 4, a flow diagram of one process alternative is given: the reaction takes place

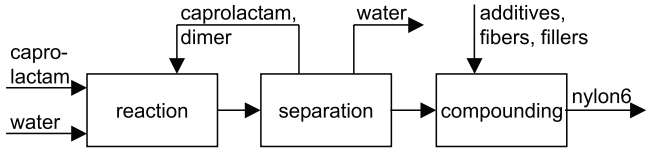

Fig. 3. Block flow diagram of polyamides6 process.

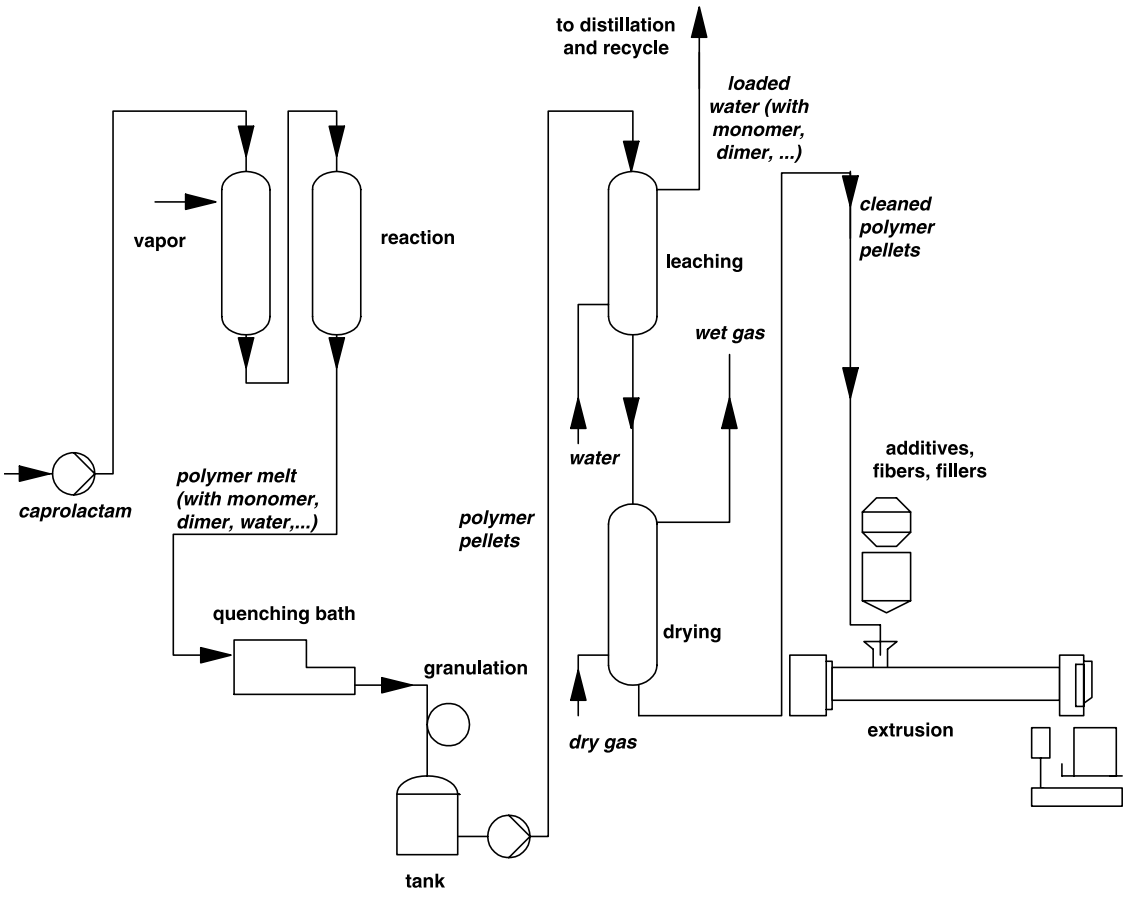

Fig. 4. Flow diagram for polyamide6 production.

within two reactors, separation is done by leaching and drying of polymer pellets. The cleaned pellets are re-melted in the extruder so that additives can be added. More detailed descriptions of the case study can be found in Bayer, Eggersmann, Gani, and Schneider (2002) and Eggersmann, Hackenberg, Marquardt, and Cameron (2002).

In Fig. 5, a simplified overview on the polyamide6 case study is given from a workflow perspective. Regarding the above mentioned applications of our case study it was important to choose a suitable notation, in the sense that engineers as well as computer scientists are able to understand the modeled content and that all necessary information is included in such a workflow model. The notation used is the *C3 formalism* (Foltz, Killich, Wolf, Schmidt, & Luczak, 2001), a modeling language for the notation of work processes, based on the Unified Modeling Language (UML) (Booch, Rumbaugh, & Jacobson, 1999). The abbreviation C3 stands for the three aspects of workflow modeling which are represented in this formalism: cooperation, coordination, and communication. The elements of C3 are roles (e.g., simulation expert), activities (e.g., design reaction alternatives), input/output information (not shown in this figure), control flows (solid lines including forks and joins represented by bars), information flows (also not shown), and synchronous communication (represented by a filled square).

After the start of the project, different alternatives are evaluated on the basis of block flow diagrams. Reaction, separation, and extrusion are investigated in parallel within different simulation tools. The simulation results are compared with experimental results leading to an improvement and refinement of the used mathematical models. After the completion of these activities a plant concept is determined. The case study in its actual form represents one part of an industrial design process, namely the early phases of basic engineering.

On the basis of this case study (with about one hundred activities) the management system has been developed and tested. For reasons of clarity only a small part of the case study is presented and used for demonstration in the following sections.

3. Management of the design process

3.1. Basic notions

In the previous section, we have introduced our domain of discourse—design processes in chemical engineering—and a case study (design of a plant for

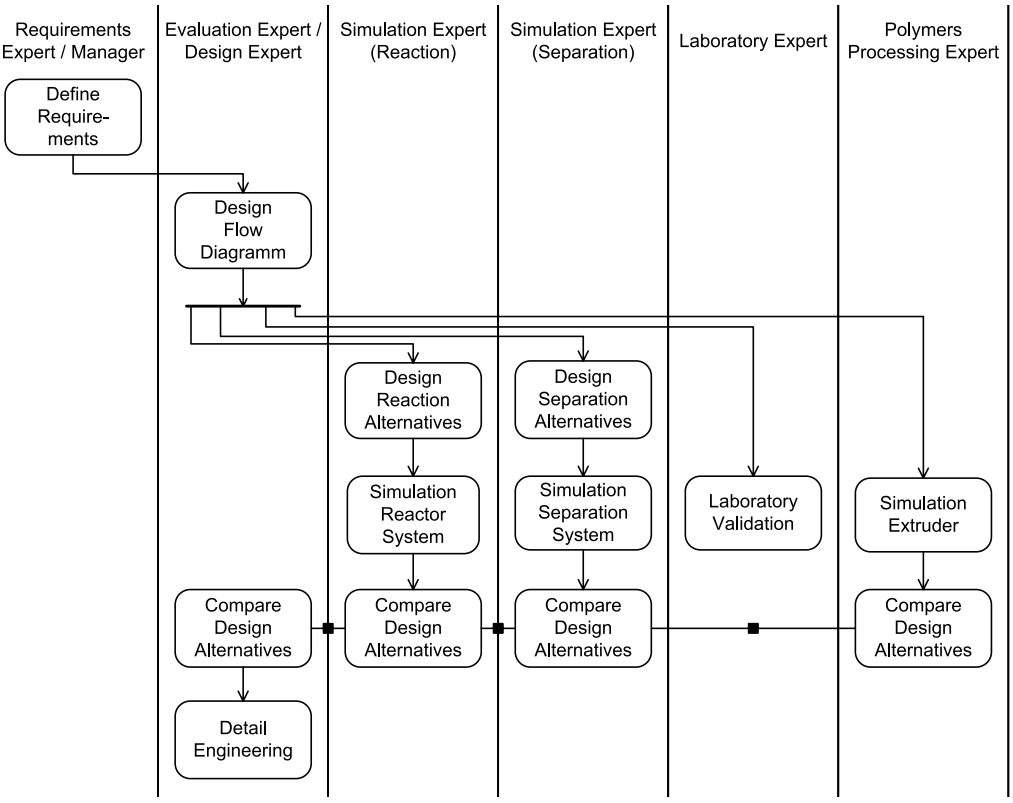

Fig. 5. Simplified overview on case study.

producing polyamide6) which serves as a reference scenario within the IMPROVE project. In this paper, we are specifically concerned with the management of design processes. In Section 2.1, we have described several features of design processes which challenge the capabilities of management. In particular, design processes are highly creative, many iterations are necessary, the tasks to be solved are not known beforehand, design processes last for a long time span, experts from different disciplines have to cooperate smoothly, these experts use heterogeneous tools, and they have to document the results of their work in a traceable way.

Before we discuss solutions to these problems, we have to introduce a set of basic notions which we will use throughout the rest of this paper. In general terms, *management* can be defined as 'all the activities and tasks undertaken by one or more persons for the purpose of planning and controlling the activities of others in order to achieve an objective or complete an activity that could not be achieved by the others acting alone' (Thayer, 1988). This definition stresses coordination as the essential function of management.

More specifically, we focus on the management of design processes by coordinating the technical work of designers. We do not target senior managers who work at a strategic level and are not concerned with the details of enterprise operation. Rather, we intend to support project managers who collaborate closely with the designers performing the technical work. Such managers, who are deeply involved in the operational business, need to have not only managerial but also technical skills ('chief designers').

The distinction between *persons* and *roles* is essential: when referring to a 'manager' or a 'designer', we are denoting a role, i.e. a collection of authorities and responsibilities. However, there need not be a 1:1 mapping between roles and persons playing roles. In particular, each person may play multiple roles. For example, in chemical engineering it is quite common that the same person acts both as a manager coordinating the project and as a (chief) designer who is concerned with technical engineering tasks.

In order to support managers in their coordination tasks, design processes have to be dealt with at an appropriate level of detail. We may roughly distinguish between three levels of *granularity*:

- At a *coarse-grained level*, design processes are divided into phases (or working areas) according to some life cycle model (Fig. 2).
- At a *medium-grained level*, design processes are decomposed further down to the level of documents or tasks, i.e. units of work distribution.
- At a *fine-grained level*, the specific details of design subprocesses are considered. For example, a simulation expert may build up a simulation model from mathematical equations.

Given our understanding of management as explained above, the coarse-grained level does not suffice; rather, decomposition has to be extended to the medium-grained level. On the other hand, management is usually not interested in the technical details of how documents are structured or how the corresponding personal subprocess is performed. Thus, the *managerial level*, which defines how management views design processes, comprises both coarse- and medium-grained representations.

In order to support managers in their coordination tasks, they must be supplied with appropriate views (abstractions) of design processes. Such views must be comprehensive inasmuch as they include products, activities, and resources (and their mutual relationships):

(1) The term *product* denotes the results of design subprocesses (e.g., flow diagrams, simulation models, simulation results, cost estimates, etc.)[1]. These may be organized into *documents*, i.e. logical units which are also used for work distribution or version control.

(2) The term *activity* denotes an action performing a certain function in a design process. At the managerial level, we are concerned with *tasks*, i.e. descriptions of activities assigned to designers by managers.

(3) Finally, the term *resource* denotes any asset needed by an activity to be performed. This comprises both human and computer resources (i.e. the designers and managers participating in the design process as well as the computers and the tools they are using).

Thus, an overall *management configuration* consists of multiple parts representing products, activities, and resources. An example is given in Fig. 6. Here, we refer to the polyamide6 design process introduced earlier. On the left, the figure displays the roles in the design team as well as the designers filling these roles[2]. The top region on the right shows design activities connected by control and data flows. Finally, the (versioned) products of these activities are located in the bottom–right region.

Below, we give a more detailed description of Fig. 6:

(1) *Products.* The results of design processes such as process flow diagrams (PFDs), steady-state and dynamic simulations, etc. are represented by documents (ellipses). Documents are interdependent, e.g., a simulation model depends on the PFD to which it refers (arrows between ellipses). The evolution of documents is captured by version control (each box within an ellipsis represents a version of some document).

(2) *Activities.* The overall design process is decomposed into tasks (rectangular boxes) which have inputs and outputs (white and black circles, respectively). The order of tasks is defined by control flows (thick arrows); e.g., reaction alternatives must have been inserted into the flow diagram before they can be simulated. Finally, data flows (arrows connecting circles) are used to transmit document versions from one task to the next.

(3) *Resources.* Employees (icons on the left) such as Schneider, Bayer, etc. are organized into project teams which are represented by organization charts. Each box represents a position, lines reflect the organizational hierarchy. Employees are assigned to positions (or roles). Within a project, an employee may play multiple roles. e.g., Mrs Bayer acts both as a designer and as a simulation expert in the polyamide6 team.

(4) *Integration.* There are several relationships between products, activities, and resources. In particular, tasks are assigned to positions (and thus indirectly to employees). Furthermore, document versions are created as outputs and used as inputs of tasks.

It is crucial to understand the scope of the term 'management' as it is used in this paper. As already stated briefly above, management requires a certain amount of abstraction. This means that the details of the *technical level* are not represented at the managerial level. This is illustrated in Fig. 7, whose upper part shows a small cutout of the management configuration of Fig. 6. On the managerial level, the design process is decomposed into activities such as creation of reaction alternatives and simulation of these alternatives. Activities generate results which are stored in document versions. At the managerial level, these versions are basically considered black boxes, i.e. they are represented by a set of descriptive attributes (author, creation date, etc.) and by references to the actual contents, e.g., PFDs and simulation models. How a PFD or a simulation model is structured internally (and how their contents are related to each other), goes beyond the scope of the managerial level. Likewise, the managerial level is not concerned with the detail personal process which is executed by some human to create a PFD, a simulation model, etc.

This does not imply that technical details are ignored. Rather, it must be ensured that the managerial level actually constitutes a correct abstraction of the fine-grained information at the technical level—and also controls technical activities. In fact, the management system described in this paper is part of an integrated environment for supporting design processes in chemical engineering (see also Fig. 1). As such, it is integrated with tools providing fine-grained product and process support (Bayer et al., 2002; Becker et al., 2002). The interplay of the tools of the overall environment is sketched only briefly in this paper; see (Nagl, Schneider, & Westfechtel, in press).

[1] Here we do not refer to the product of the chemical process such as e.g. polyamide6. Thus, the reader should be aware of the context in which this notion is used. This remark applies equally to the notion of process, which may refer to both the design process and the chemical process.

[2] For the time being, we ignore computer resources, see Fig. 8.

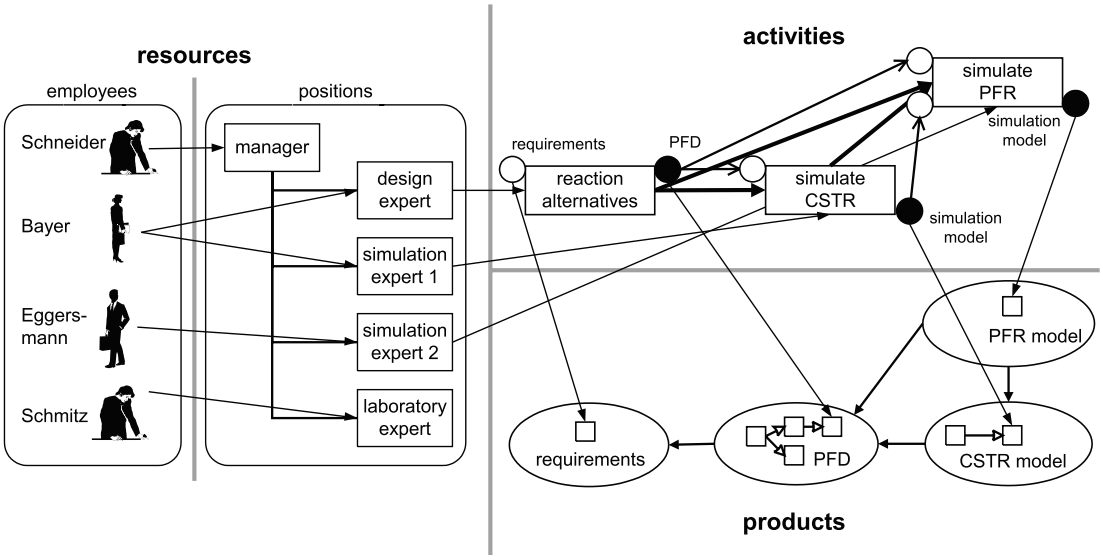

Fig. 6. Management configuration.

Particular attention has to be paid to the *dynamics* of design processes (called evolution in (Smithers & Troxell, 1990)). As we have demonstrated in the previous section, the design process is not known in advance. Rather, it continuously evolves during execution. Often, the term 'dynamics' is interpreted in a rather restricted way, referring only to the execution of a known process with a static definition. In addition, we have to take care of evolving definitions, i.e. the activities to be executed as well as their relationships are defined only at runtime.

As a consequence, all parts of a management configuration evolve continuously:

(1) *Products.* The product structure is determined only during the design process. It depends on the flow diagram which is continuously extended and modified. Other documents such as simulation models and simulation results depend on the flow diagram. Moreover, different variants of the chemical process are elaborated, and selections among them are performed according to feedback gained by simulation and experiments.

(2) *Activities.* The activities to be performed depend on the product structure, feedback may require the re-execution of terminated activities, concurrent/simultaneous engineering calls for sophisticated coordination of related activities, etc.

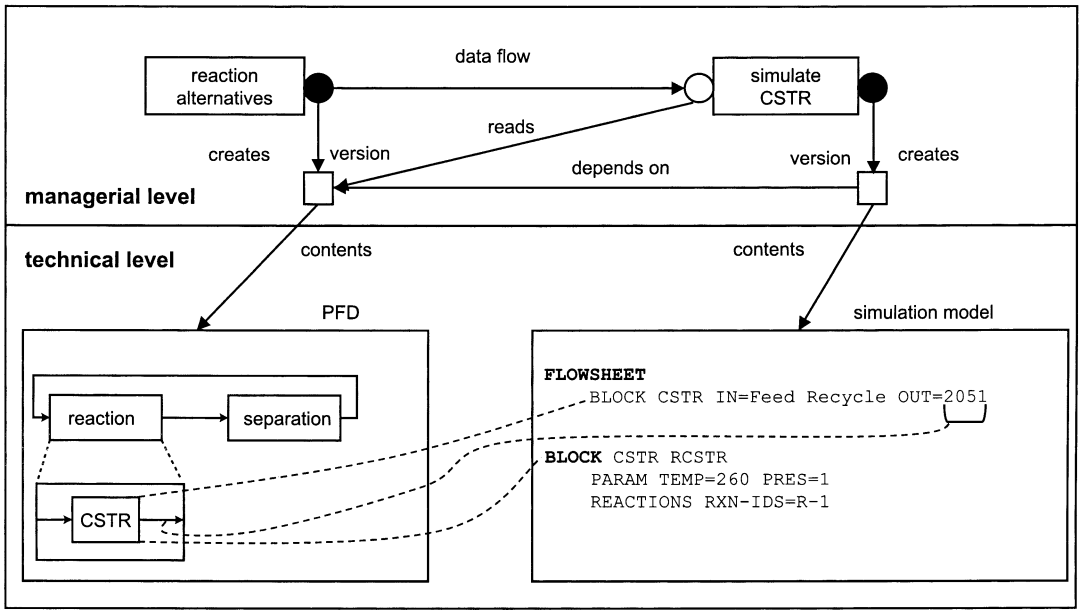

Fig. 7. Managerial and technical level.

Table 1
Comparison of AHEAD with commercial management systems

	AHEAD	Project management systems	Workflow management systems	Product management systems
Granularity of representation	Medium-grained	Coarse-grained	Medium- and fine-grained	Medium-grained
Coverage at the managerial level	Products, activities, resources	Activities, resources	Activities, resources	Products, (activities)
Integration with technical level	Tool integration, document storage	Not supported	Tool integration	Document storage
Support for dynamic design processes	Full support of process evolution	Evolving project plans	Limited (fixed workflows)	Version control for documents
Adaptability	UML models	Not supported	Workflow definitions	Database schema

(3) *Resources.* Resource evolution occurs likewise: new tools arrive, old tool versions are replaced with new ones, the project team may shrink due to budget constraints, or it may be extended to meet a crucial deadline, etc.

However, a management configuration should not evolve in arbitrary ways. There are *domain-specific constraints* which have to be met. In particular, activities can be classified into types such as requirements definition, design, simulation, etc. (likewise for products and resources). Furthermore, the way how activities are connected is constrained as well. For example, a flow diagram can be designed only after the requirements have been defined. Such domain-specific constraints should be taken into account such that they restrict the freedom of evolution.

3.2. Management systems: state of the art

Above, we have discussed design processes and their management on a conceptual level. In the following, we will be concerned with *tool support* for managing design processes. From the previous discussion, we derive a set of crucial requirements for management tools for design processes[3]:

(1) *Medium-grained representation.* The management of design processes has to be supported at an appropriate level of detail. As we have argued before, this requires a medium-grained representation of the design process.

(2) *Coverage and integration at the managerial level.* Management tools have to deal equally with products, activities, and resources. In addition, the relationships among them have to be taken into account.

(3) *Integration between managerial and technical level.* Managerial activities have to be coupled with technical activities. Practically speaking, this implies e.g., that designers have to be supplied with the documents they are going to manipulate, as well as with the tools they are going to use.

(4) *Dynamics of design processes.* Design processes evolve continuously during execution. Thus, a management system must support dynamic changes so that product evolution, feedback, simultaneous and concurrent engineering etc. can be expressed.

(5) *Adaptability.* Management tools have to be adapted to a specific application domain. For example, they must be aware of the types of activities performed in design processes in chemical engineering, and they must provide domain-specific operations to their users.

In industry, a large variety of commercial systems are being used for the management of design processes. These include systems for project management, workflow management, and product management, which are discussed in turn below. All of these systems meet the requirements stated above only partially (Table 1[4]).

Project management systems (Kerzner, 1998) such as e.g., Microsoft Project support management functions such as planning, organizing, monitoring, and controlling. The project plan acts as the central document which may be represented in different ways, e.g., as a PERT or GANTT chart. It defines the milestones to be accomplished and provides the foundation for scheduling of resource utilization as well as for cost estimation and control. Project management systems are widely used in practice, but they still suffer from several limitations: project plans are often too coarse-grained, products (documents) are not considered, project plans are not integrated with the actual work performed by engineers, and there is no way to define domain-specific types of project plans.

Workflow management systems (Jablonski & Bußler, 1996; Lawrence, 1997), e.g., Staffware, FlowMark, or COSA, have been applied in banks, insurance companies, administrations, etc. A workflow management system manages the flow of work between participants, according to a defined procedure consisting of a number

[3] Of course, the list given below is not complete. Rather, we focus on (important) requirements which we have satisfactorily addressed in our work and which are not satisfactorily addressed by the commercial systems to be discussed below.

[4] The AHEAD column refers to our own and will be discussed in the next section.

of tasks (McCarthy & Bluestein, 1991). It coordinates user and system participants to achieve defined objectives by set deadlines. To this end, tasks and documents are passed from participant to participant in a correct order. Moreover, a workflow management system may offer an interface to invoke a tool on a document either interactively or automatically. Their most important restriction is limited support for the dynamics of design processes. Many workflow management systems assume a statically defined workflow that cannot be changed during execution. In this way, dynamic design processes can be supported only to a limited extent (i.e. the statically known fractions can be handled by the workflow management system). Recently, this problem has been addressed in a few university prototypes (see e.g., Derniame, Baba, & Wastell, 1998; Georgakopoulos, Prinz, & Wolf, 1999).

In the context of this paper, we use the term *product management system* to refer to all kinds of systems for storing, manipulating, and retrieving the results of design processes. Depending on the context in which they are employed, they are called engineering data management systems, product data management systems (Harris, 1996), software configuration management systems (Tichy, 1994; Whitgift, 1991), or document management systems. Documentum and Matrix One are examples of such systems which are used in chemical engineering. Documents such as flow diagrams, steady-state and dynamic simulation models, cost estimations, etc. are stored in a database which records the evolution of documents (i.e. their versions) and aggregates them into configurations. In addition, product management systems may offer simple support for the management of activities (e.g., change request processes based on finite state machines), or they may include workflow components, which suffer from the restrictions already discussed above. Their primary focus still lies on the management of products; in particular, management of human resources is hardly considered.

All of the approaches cited above are *domain-independent*. For example, workflow management systems may be applied to arbitrary business processes, and product management systems may be used in different engineering disciplines. We are aware of only a few *domain-specific* approaches which have been developed for chemical engineering. For example, n-dim (Levy et al., 1993; Westerberg et al., 1997) is a distributed and collaborative computer-aided environment for process engineering design; KBDS (Bañares-Alcántara & Lababidi, 1995) deals with the management of design alternatives and design histories. These approaches are better tailored towards design processes in chemical engineering. However, they do not provide comprehensive support for the management of products, activities, and resources. Moreover, they lack the generality of domain-independent systems which can be used in and adapted to different domains.

4. A management system for design processes

Since current management systems suffer from several limitations explained in the previous section, we have

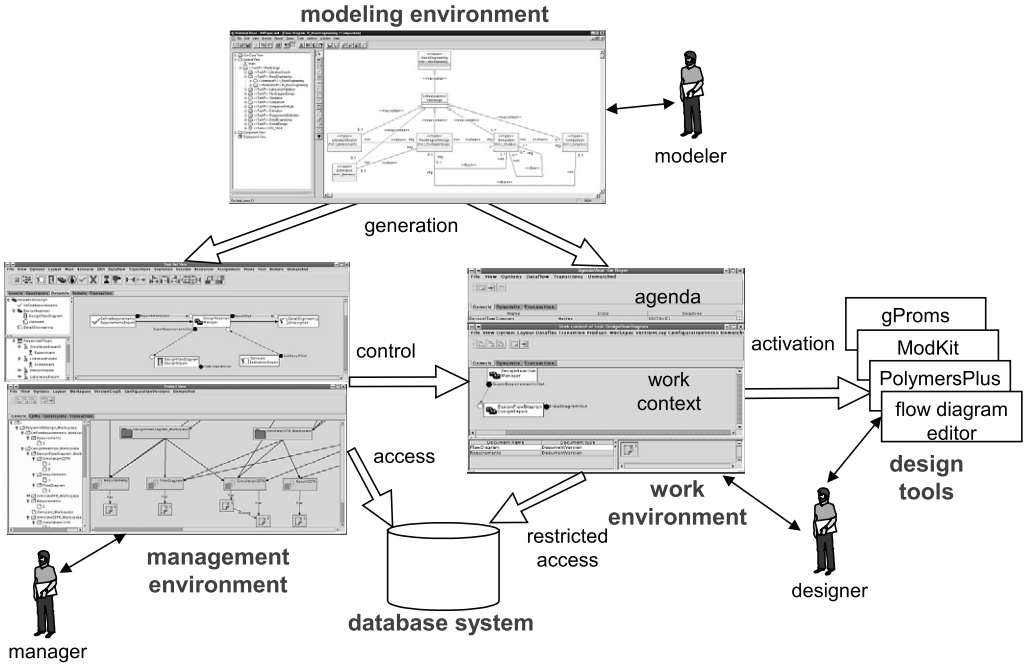

Fig. 8. Architecture of the AHEAD system.

designed and implemented a new management system which addresses these limitations. This system is called *AHEAD* (Jäger, Schleicher, & Westfechtel, 1999b; Westfechtel, 1999). AHEAD is a research prototype that goes beyond commercial systems with respect to the requirements introduced earlier (Table 1):

(1) *Medium-grained representation.* In contrast to project management systems, design processes are represented at a medium-grained level, allowing managers to effectively control the activities of designers. Management is not performed at the level of milestones, rather, it is concerned with individual tasks such as 'simulate the CSTR reactor'.

(2) *Coverage and integration at the managerial level.* AHEAD is based on an integrated management model which equally covers products, activities, and resources. In contrast, project and workflow management systems primarily focus on activities and resources, while product management systems are mainly concerned with the products of design processes.

(3) *Integration between managerial and technical level.* In contrast to project management systems, the AHEAD system also includes support tools for designers that supply them with the documents to work on, and the tools that they may use.

(4) *Support for the dynamics of design processes.* While many workflow management systems are too inflexible to allow for dynamic changes of workflows during execution, AHEAD supports evolving design processes, allowing for seamless integration of planning, execution, analysis, and monitoring.

(5) *Adaptability.* Both the structure of management configurations and the operations to manipulate them can be adapted by means of a domain-specific object-oriented model based on the UML (Booch et al., 1999).

4.1. Functionality and concepts

Fig. 8 gives an overview of the AHEAD system. AHEAD offers environments for different kinds of users, which are called modeler, manager, and designer. In the following, we will focus on the functionality that the AHEAD system provides to its users. Its technical realization will be discussed in the next subsection.

The *management environment* supports project managers in planning, analyzing, monitoring, and controlling design processes. It provides graphical tools for operating on management configurations. These tools address the management of activities, products, and resources, respectively (Krapp, Krüppel, Schleicher, & Westfechtel, 1998):

(1) For *activity management*, AHEAD offers dynamic task nets which allow for seamless interleaving of planning, analyzing, monitoring, and controlling. A task net consists of tasks that are connected by control flow and data flow relationships. Furthermore, feedback

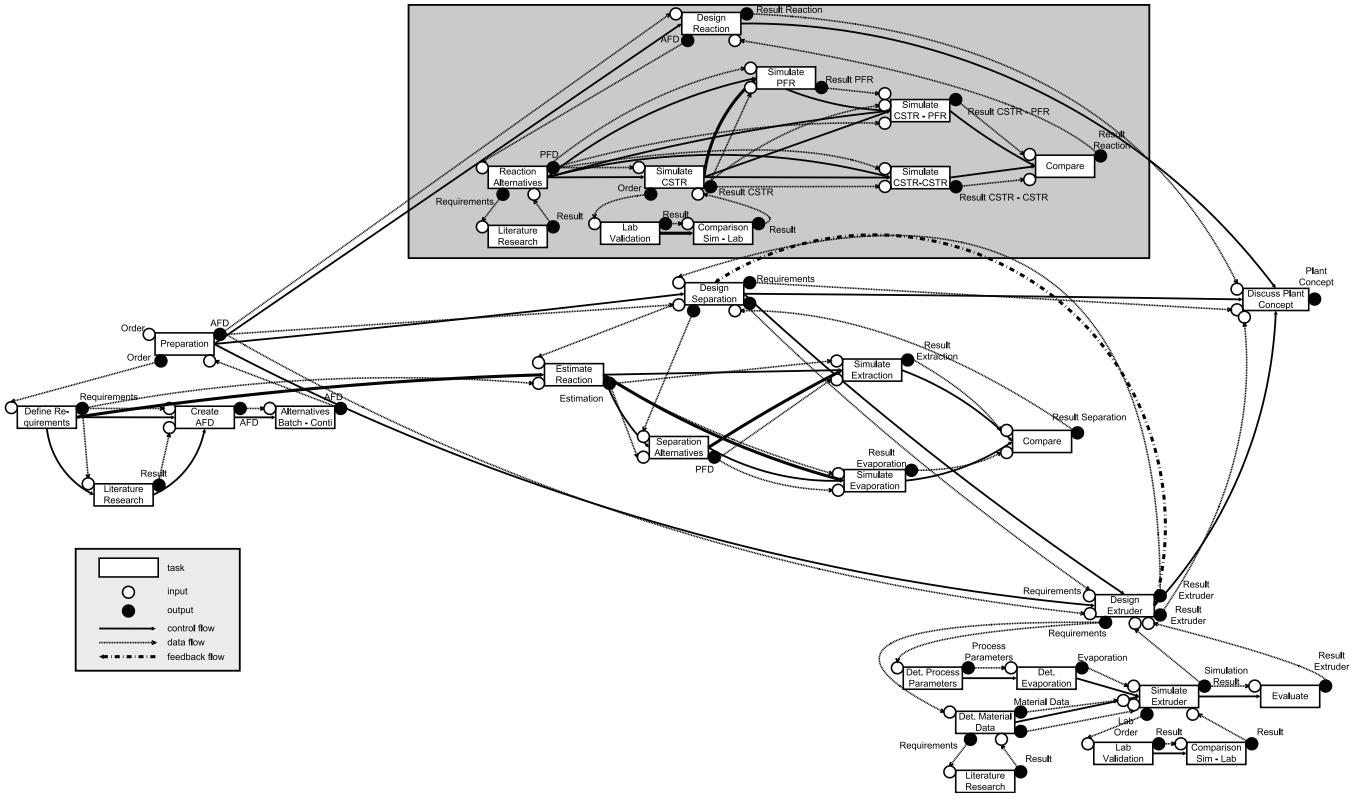

Fig. 9. Task net for the polyamide6 design process.

in the design process is represented by feedback relationships. Tasks may be decomposed into subtasks, resulting in task hierarchies. The manager constructs task nets with the help of a graphical editor. He may modify task nets at any time while a design process is being executed.

(2) *Product management* is concerned with documents such as flow diagrams, simulation models, cost estimations, etc. AHEAD offers version control for these documents with the help of version graphs. Relationships (e.g., dependencies) between documents are maintained as well. Versions of documents may be composed into configurations, thereby defining which versions are consistent with each other. The manager may view the version histories and configurations with the help of a graphical tool. In this way, he may keep track of the work results produced by the designers.

(3) *Resource management* deals with the organizational structure of the enterprise as far as it is relevant to design processes. AHEAD distinguishes between abstract resources (positions or roles) and concrete resources (employees). The manager may define a project team and then assign employees to the project positions.

Management of activities, products, and resources is fully integrated: tasks are assigned to positions, inputs and outputs of tasks refer to document versions. Moreover, AHEAD manages task-specific workspaces of documents and supports invocation of design tools (see below).

AHEAD does not only support managers. In addition, it offers a *work environment* which consists of two major components:

(1) The *agenda tool* displays the tasks assigned to a designer in a table containing information about state, deadline, expected duration, etc. The designer may perform operations such as starting, suspending, finishing, or aborting a task.

(2) The *work context tool* manages the documents and tools required for executing a certain task. The designer is supplied with a workspace of versioned documents. He may work on a document by starting a tool such as e.g., a flow diagram editor, a simulation tool, etc.

Please note that the scope of support provided by the work environment is limited. We do not intend to support design activities in detail at a technical level. Rather, the work environment is used to couple technical activities with management. There are other tools which support design activities at a fine-grained level. For example, a process-integrated flow diagram editor (Bayer, Marquardt, Weidenhaupt, and Jarke, 2001) may be activated from the work environment. 'Process-integrated' means that the designer is supported by process fragments which correspond to frequently occurring command sequences. These process fragments encode the design knowledge which is available at the technical level. This goes beyond the scope of the AHEAD system, but it is covered by the overall environment for supporting design processes to which AHEAD belongs as a central component.

Both the management environment and the work environment access a common *management database*. However, they access it in different ways, i.e., they invoke different kinds of functions. The work environment is restricted to those functions which may be invoked by a designer. The management environment provides more comprehensive access to the database. For example, the manager may modify the structure of a task net, which is not allowed for a designer.

Before the AHEAD system may be used to carry out a certain design processes, it must be adapted to the respective application domain Schleicher, 1999. AHEAD consists of a generic kernel which is domain-independent. Due to the generality of the underlying concepts, AHEAD may be applied in different domains

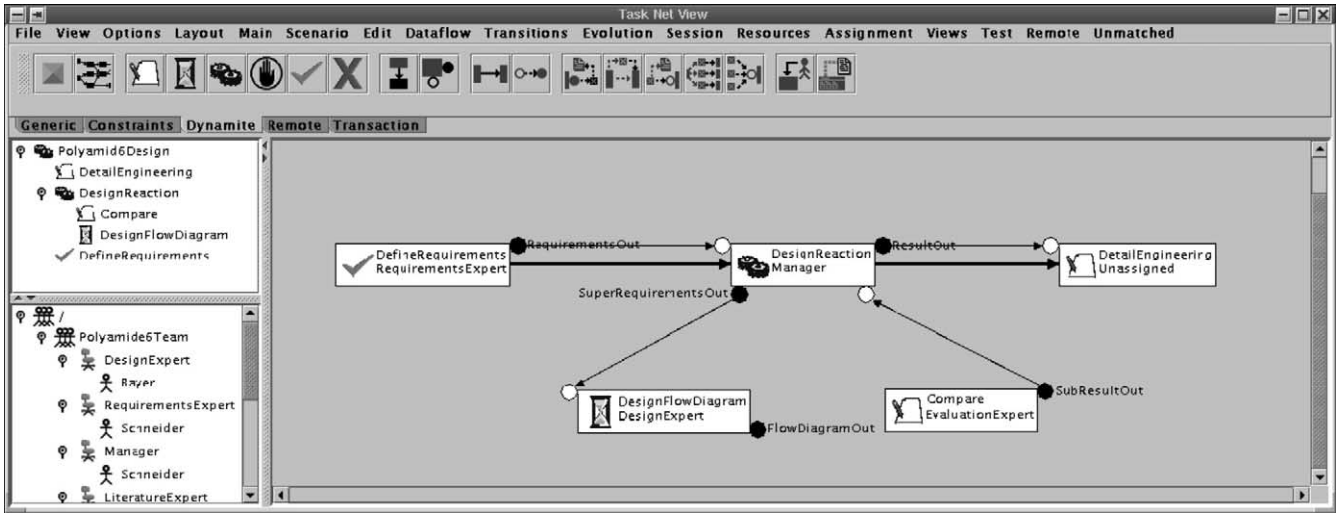

Fig. 10. Initial task net (management environment).

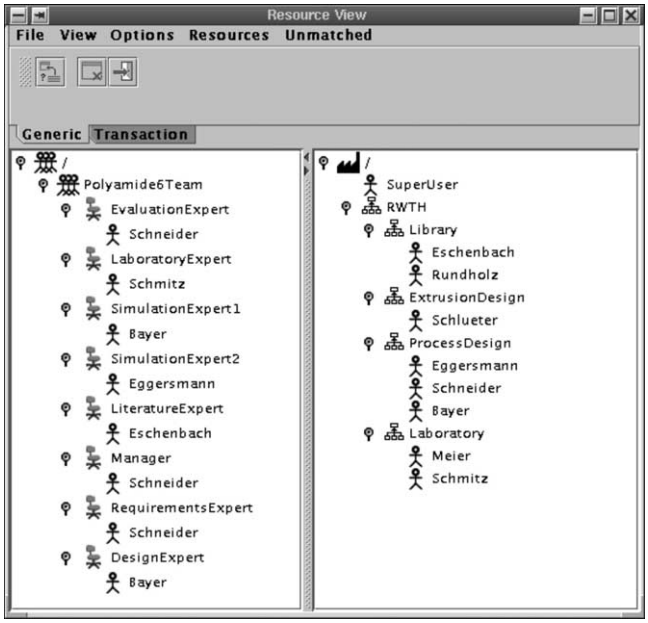

Fig. 11. Resource view (management environment).

such as software, mechanical, or chemical engineering. On the other hand, each domain has its specific constraints on design processes. The *modeling environment* is used to provide AHEAD with domain-specific knowledge, e.g., by defining task types for flow diagram design, steady-state and dynamic simulation, etc. From a domain-specific process model, code is generated for adapting the management and the work environment.

4.2. Realization

AHEAD is a research prototype which has been developed to demonstrate novel functionality. To implement AHEAD, we have used powerful homegrown tools for rapid prototyping (see below). This was the fastest way to obtain a demonstrator, which, however, cannot be immediately employed in industry. In Section 6, we will discuss technology transfer, which can be achieved by reusing commercial tools for project, workflow, and product management.

In the current realization of the AHEAD system, *graph technology* plays an important role. All data for representing management configurations are stored in the graph-based database management system GRAS (Kiesel, Schürr, & Westfechtel, 1995). Graph structures and operations are formally specified in the high-level specification language PROGRES (Schürr, Winter, & Zundorf, 1999), which is based on programmed graph transformations. In AHEAD, the specification written in PROGRES describes how management configurations are built up and what operations for manipulating them are offered to end users. From the specification, code is generated which operates on the management database.

For the user interface, we have developed the *UPGRADE* framework (*U*niversal *P*latform for *GRA*ph-Based Application *DE*velopment (Böhlen, Jäger, Schleicher, & Westfechtel, 2002)). UPGRADE is implemented in Java, based on standard libraries and both public-domain and commercial components (ILOG JViews). It mainly focuses on graphical tools, but it also supports e.g., tabular and tree representations. Graphical tools provide external views on the underlying management graph, hiding all of the technical details of the internal representation.

From the work environment, external tools may be started with the help of *wrappers*. Wrappers constitute tool envelopes which are responsible for supplying tools with data (documents checked out from the product management database), preparing the operating system environment (e.g., by setting environment variables), and calling the tools with appropriate parameters. Wrappers are realized on top of CORBA, an infrastructure for distributed object-oriented computing. The realization of wrappers has been performed by another partner participating in the IMPROVE project (Lipperts & Thißen, 1999).

The *modeling environment* is realized with the help of a commercial CASE tool (Rational Rose), which is based on the *UML* (Booch et al., 1999). UML is a language that serves as a standard notation for object-oriented modeling. For the purpose of process modeling, we have adapted and restricted UML according to our requirements (Jäger et al., 1999b). A UML model is automatically transformed into a (part of a) PROGRES specification (i.e., the end user is not concerned with PROGRES at all). This transformation tool is realized with the help of the OLE interface providing access to Rose's model database.

5. Applying the management system to chemical engineering

Within the IMPROVE project, the polyamide6 design process played a key role not only on a conceptual level. In addition, it was used for the development of a demonstrator integrating all software components contributed by the various project partners. The AHEAD system served as the central, coordinating component of this demonstrator (see also Section 2.2). It was integrated with the following tools: the PRIME flow diagram editor (Bayer et al., 2001), ModKit (for creating dynamic simulation models, see (Bogusch, Loehmann, & Marquardt, 2001)), KomPakt (for synchronous multimedia communication (Schüppen et al., 2002)), and Morex (for extruder design) were contributed by partners involved in the IMPROVE project; EXCEL (from Microsoft) and PolymersPlus (from Aspen Tech) are commercial tools which are used for cost calculations

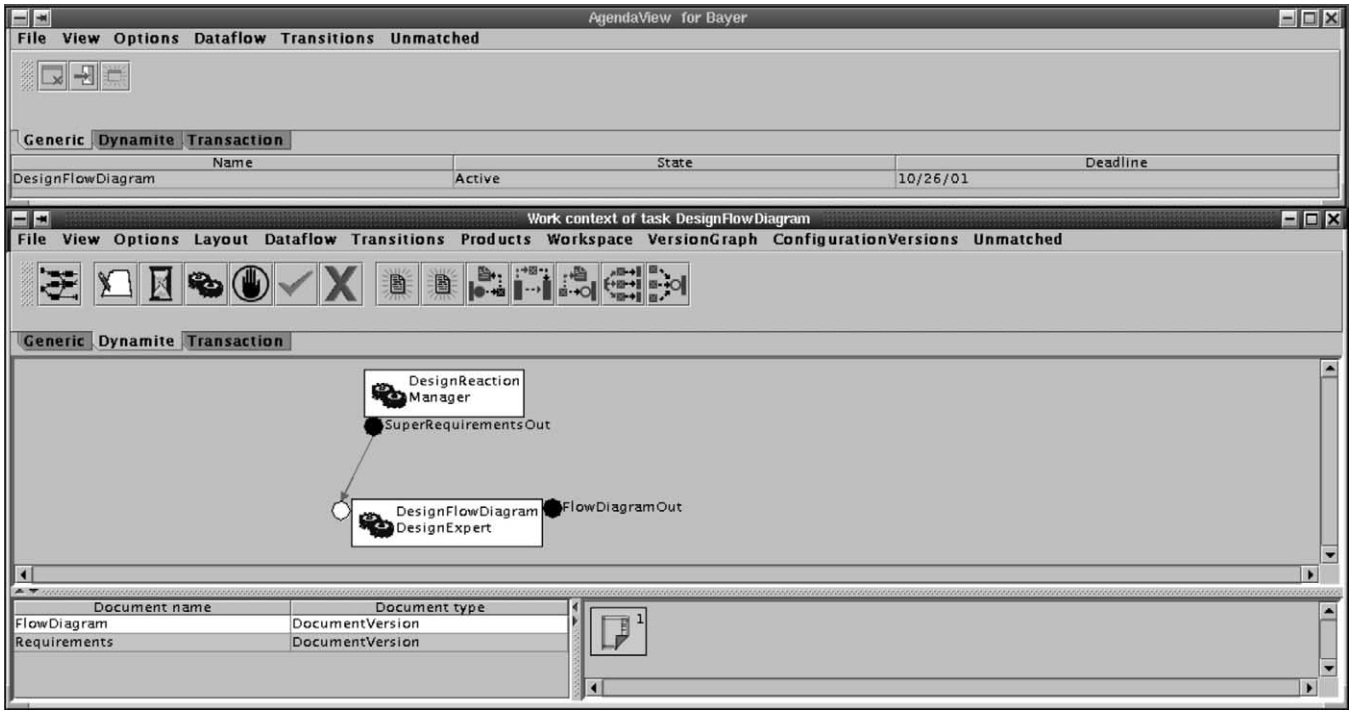

Fig. 12. Work environment.

and steady-state simulations, respectively. For the demonstrator, a comprehensive demo session was prepared which was successfully presented at the IMPROVE project review in May 2000 and at a national workshop with participants from chemical industry and tool vendors that was held in November 2000.

5.1. The polyamide6 design process

Based on the case study introduced in Section 2.2, the polyamide6 design process is represented as a dynamic task net in Fig. 9. This task net was created from a representation in the C3 modeling language which we used in Fig. 2 and Fig. 5. Please note that Fig. 5 shows a very simplified view of the design process. To construct the task net of Fig. 9, we used a more detailed C3 model.

As we will demonstrate in the demo session of Section 5.2, the task net of Fig. 9 is not available at the start of the design process. Rather, it is built up as the design proceeds. In an early phase, the task net will just contain a top-level decomposition into design tasks for reaction, separation, and compounding, respectively. However, in the IMPROVE project we were dealing with a case study which was designed beforehand (and was used to plan the development of the demonstrator). Therefore, a C3 model of the overall design process was already available in this artifical setting; it would not be available in real-world use.

The mapping from C3 to task nets was performed manually, albeit in a systematic manner. In general, human expertise is required to perform the mapping. On the other hand, there is a straightforward initial mapping which may be obtained by applying the following simple rules:

(1) Each activity is mapped onto a task. This also applies to cooperative activities, which occur only once in a task net.[5]

(2) Ordering relationships between activities are mapped onto control flows. While forks and joins are represented in C3 by bars, they are not shown explicitly in task nets. Rather, they appear as bundles of outgoing and incoming control flows, respectively.

(3) An arrow from an activity to a document is mapped onto an output parameter of the respective task. For each arrow from a document to an activity, an input parameter is generated, and output and input are connected by a data flow.

Human expertise is required e.g., for introducing task hierarchies. The C3 model which we used as input was flat. Based on domain knowledge, reasonable subprocesses may be identified. In our case study, we have introduced subprocesses for designing the reaction, the separation, and the compounding of the chemical process. In addition, there is a preparation phase in which the overall design problem is decomposed, and a final integration phase, where the overall plant design is

[5] In C3, multiple nodes are required because each activity node occurs in one swim-lane corresponding to a position or role.

synthesized and discussed among the involved design experts.

Please note that reaction, separation, and compounding are designed in an integrated way:

(1) To design the separation, input is required with respect to incoming streams. In the sample process, this input is generated by an initial estimation which is refined later on. In this way, reaction and separation may be studied in parallel. This speeds up the overall design process.

(2) With respect to separation and compounding, there exist some degrees of freedom with respect to the decomposition of the overall chemical process. In particular, if separation is performed with the help of evaporation, evaporation can be performed partly in the extruder. Therefore, the respective design processes are arranged in a feedback loop for optimizing the chemical process.

In the demo session to be presented below, we focus on the design of the reaction (shaded region in Fig. 9). After an initial PFD has been created which contains multiple design variants, each of these variants is explored by means of simulations and (if required) laboratory experiments. In a final step, these alternatives are compared against each other, and the most appropriate one is selected. This simplified part of the design process suffices to demonstrate many of the essential features provided by the AHEAD system; furthermore, it is sufficiently small to be presented in this paper.

5.2. Demo session

In this subsection, we illustrate the functionality of the AHEAD system with the help of some snapshots. We will primarily focus on the management environment; the work environment will be discussed rather briefly. The modeling environment goes beyond the scope of this paper; see Jäger et al. (1999a).

Fig. 10 presents a snapshot from the management environment taken in an early stage of the polyamide6 design process. The upper region on the left displays a tree view of the task hierarchy. The lower left region offers a view onto the resources available for task assignments (see also Fig. 11). A part of the overall task net is shown in the graph view on the right-hand side. Each task is represented by a rectangle containing its name, the position to which the task has been assigned, and an icon representing its state (e.g., the gear-wheels represent the state Active, and the hourglass stands for the state Waiting). Black and white circles represent outputs and inputs, respectively. These are connected by data flows (thin arrows). Furthermore, the ordering of task execution is constrained by control flows (thick arrows). Hierarchical task relations (decompositions) are represented by the graphical placement of the task boxes (from top to bottom) rather than by drawing arrows (which would clutter the diagram).

Please recall that the demo session deals only with the reaction part, i.e., we do not consider separation,

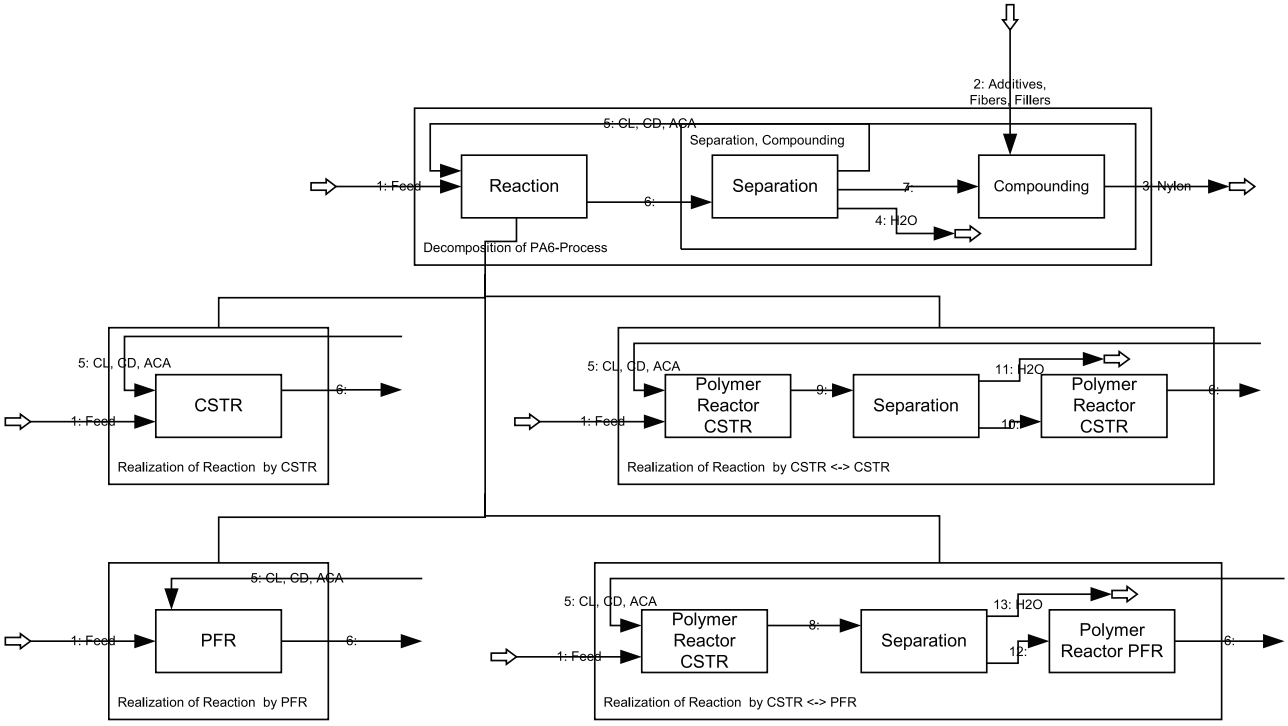

Fig. 13. Reaction alternatives in the PFD.

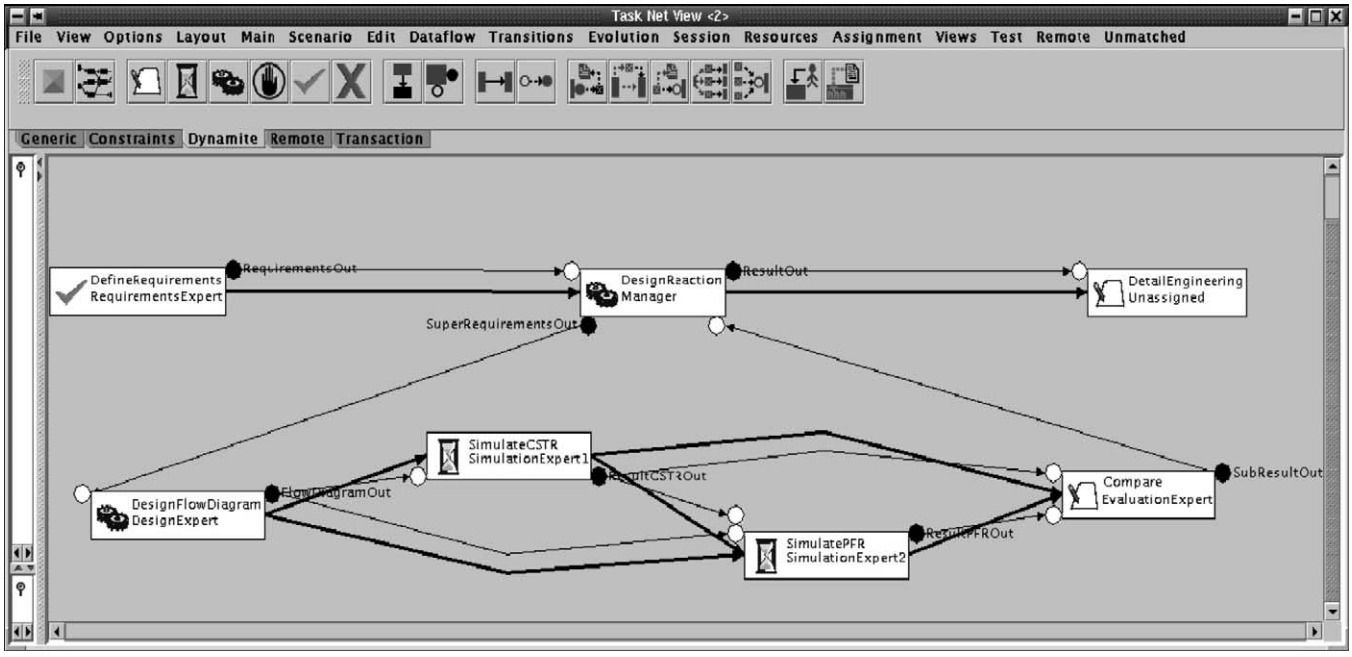

Fig. 14. Extended tesk net (management environment).

extrusion, etc. On the top level of the task net of Fig. 10, the process is decomposed into three tasks: Define Requirements, Design Reaction, and Detail Engineering. Only the design of the reaction is elaborated in the sequel. In this early stage, it is only known that initially some reaction alternatives have to be designed. The result of this design task is documented in a flow diagram. Furthermore, at the end these alternatives have to be compared, and a decision has to be performed. All other tasks—e.g., for performing simulations and laboratory experiments—have to be filled in later. Thus, the initial task net is incomplete.

In addition to the initial task net, the manager has also used the resource management tool for building up his project team (Fig. 11). The region on the left displays the structure of the polyamide6 design team. Each position (represented by a chair icon) is assigned to a team member. Analogously, the region on the right shows the departments of the company. From these departments, the team members for a specific project are taken for a limited time span. The management environment offers commands for defining both the team and the department structure, and for assigning persons to positions in design teams. Please note that tasks are assigned to positions rather than to actual employees (see lower left view in Fig. 10). In this way, assignment is decomposed into two steps. The manager may assign a task to a certain position even if this position has not been filled yet. Moreover, if a different employee is assigned to a position, the task assignments need not be changed: The tasks will be redirected to the new employee automatically.

The work environment is illustrated in Fig. 12. As a first step, the user logs into the system (not shown in the figure). After that, AHEAD displays an agenda of tasks assigned to this user (more precisely: assigned to the roles played by this user). In Fig. 12, the agenda is shown in the top window. Since the user Bayer plays the role of the design expert, the agenda contains the task Design Flow Diagram. After the user has selected a task from the agenda, the work context for this task is opened (bottom window). The work context graphically represents the task, its inputs and outputs, as well as its context in the task net (here, the context includes the parent task which defines the requirements to the flow diagram to be designed). Furthermore, it displays a list of all documents needed for executing this task. For some selected document, the version history is shown on the right (so far, there is only one version of the requirements definition which acts as input for the current task).

From the work context window, the user may activate design tools for operating on the documents contained in the workspace. Here, the user invokes a flow diagram editor (Bayer et al., 2001) in order to insert reaction alternatives into the flow diagram for the polyamide6 process. The flow diagram editor, which was also developed in the IMPROVE project, is based on MS Visio, a commercial drawing tool, which was integrated with the PRIME process engine prime (Pohl et al., 1999). The flow diagram editor supports hierarchical flow diagrams (abstract or process flow diagrams); furthermore, it may represent alternative refinements (*variants*) for blocks occurring in the flow diagram. The flow diagram editor offers fine-grained process frag-

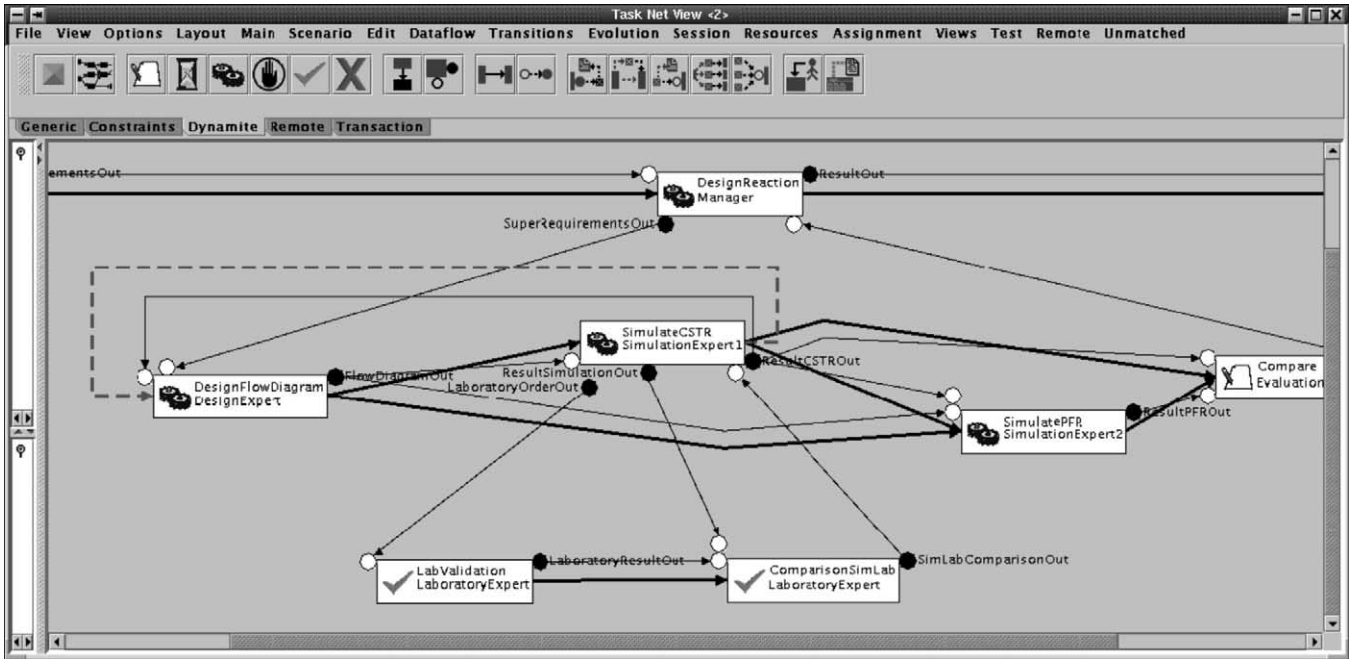

Fig. 15. Feedback and simultaneous engineering (management environment).

ments which are used for guidance and automation. These fragments incorporate domain-specific knowledge. For example, there is a process fragment which assists the designer in introducing reaction alternatives into the flow diagram (based on experience from previous design processes).

The resulting flow diagram is displayed in Fig. 13. The chemical process is decomposed into reaction, separation, and compounding. The reaction is refined into four variants. For our demo session, we assume that initially only two variants are investigated (namely a single CSTR and PFR, respectively). That is, at the current state of design the alternatives on the right-hand side have not yet been introduced into the flow diagram; they will be considered later on.

After the generation of the four variants, the manager extends the task net with tasks for investigating the alternatives that have been introduced so far (*product-dependent task net*, Fig. 14). Please note the control flow relation between the new tasks: The manager has decided that the CSTR should be investigated first so that experience from this alternative may be re-used when investigating the PFR. Furthermore, we would like to emphasize that the design task is not terminated yet. As to be demonstrated below, the designer waits for feedback from simulations in order to enrich the flow diagram with simulation data. Depending on these data, it may be necessary to investigate further alternatives.

Subsequently, the simulation expert creates a simulation model (using PolymersPlus) for the CSTR reactor and runs the corresponding simulations. The simulation results are validated with the help of laboratory experiments. After these investigations have been completed, the flow diagram can be enriched with simulation data such as flow rates, pressures, temperatures, etc. To this end, a *feedback flow*—represented by a dashed arrow—is inserted into the task net (Fig. 15). The feedback flow is refined by a data flow, along which the simulation data are propagated. Then, the simulation data are introduced into the flow diagram.

Please note that the tasks for designing the flow diagram and for investigating the CSTR are active at the same time. The semantics of control flows is defined such that tasks connected by control flows can be active simultaneously. In this way, we support *simultaneous engineering* (Bullinger & Warschat, 1996). As a consequence, we cannot assume that the work context of a task is stable with respect to its inputs. Rather, a predecessor task may deliver a new version that is relevant for its successors. This is taken care of by a sophisticated release policy built into the model underlying dynamic task nets (Westfechtel, 1999).

After the alternatives CSTR and PFR have been elaborated, the evaluation expert compares all explored design alternatives. Since none of them performs satisfactorily, feedback is raised to the design task. Note this is an example of far-reaching feedback (from the end to the start of the subprocess for designing the reaction part). Here, we assume that the designer has already terminated the design task. As a consequence, the design task has to be reactivated. Reactivation is handled by creating a new *task version*, which may or may not be

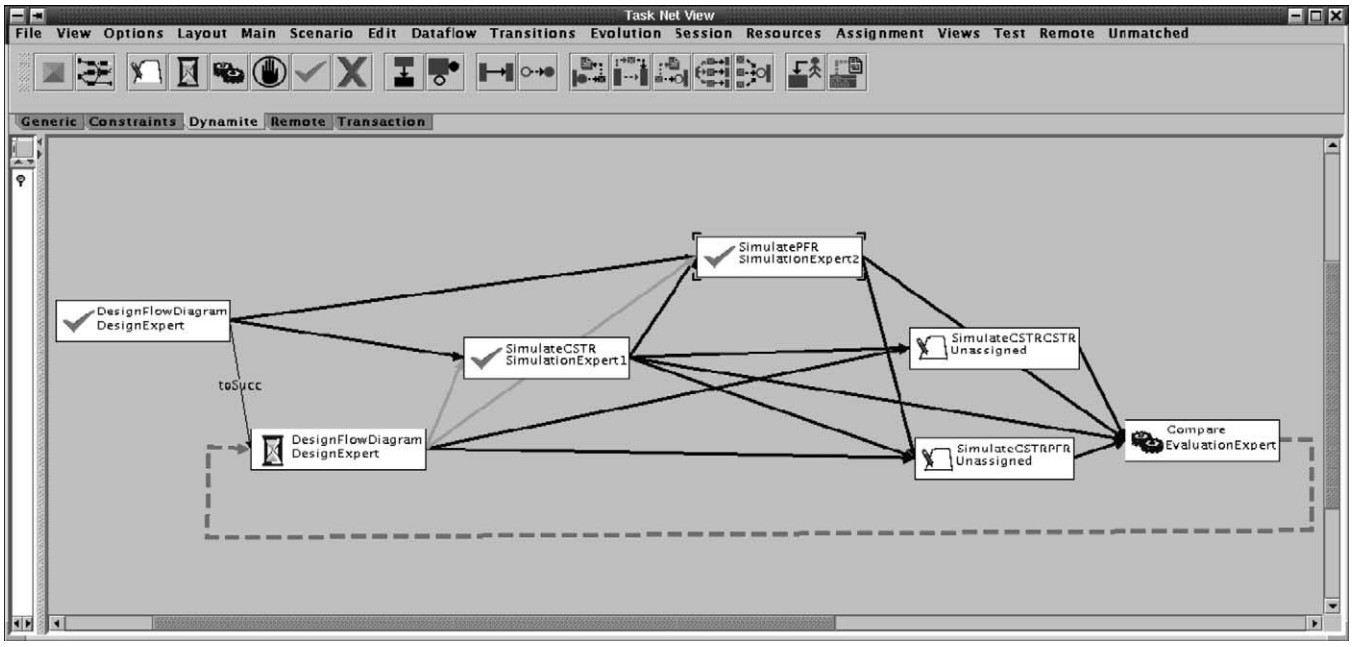

Fig. 16. Far-reaching feedback (management environment).

assigned to the same designer as before. New design alternatives are created, namely a CSTR–CSTR and a CSTR–PFR cascade, respectively (see again Fig. 13). Furthermore, the task net is augmented with corresponding simulation tasks (Fig. 16[6]). After that, the new simulation tasks are delegated to simulation experts, and simulations are carried out accordingly. Eventually, the most suitable reactor alternative is selected.

So far, we have primarily considered the management of activities. Management of products, however, is covered as well. This is illustrated by the snapshot in Fig. 17, which is again taken from the management environment. It shows a tree view on the products of design processes on the left and a graph view on the right. Products are arranged into *workspaces* that are organized according to the task hierarchy. Workspaces contain sets of *versioned documents*.

Generally speaking, a *version* represents some state (or snapshot) of an evolving document. We distinguish between *revisions*, which denote temporal versions, and *variants*, which exist concurrently as alternative solutions to some design problem. Revisions are organized into sequences, variants result in branches. Versions are connected by *history relationships*. In Fig. 17, there is a history relationship between revisions 1 and 2 of Simulation CSTR, the simulation model for the CSTR reactor. In general, the version history of a document (flow diagram, simulation model, etc.) may evolve into an acyclic graph (not shown in the snapshot).

[6] Task parameters and data flows have been filtered out to avoid cluttered diagram.

Please note that in Fig. 17 there is only one version of the flow diagram. Here, we rely on the capabilities of the flow diagram editor to represent multiple variants. Still, the flow diagram could evolve into multiple versions at the managerial level (e.g., to record snapshots at different times). Moreover, in the case of a flow diagram editor with more limited capabilities (no variants), variants would be represented at the managerial level as parallel branches in the version graph.

6. Technology transfer

In this section we discuss how the obtained *results* can be *transferred* to industry in order to improve the state of the art of coordinating industrial design processes.

6.1. Ways of technology transfer

Let us start with discussing the different *ways* of *technology transfer* of the obtained results. Transfer ranges from evaluating the new concepts introduced above to implementing the functionality of AHEAD in an industrial context by using existing systems:

(1) *Conceptual transfer* (transfer 1): Conceptual transfer includes to see whether the concepts can be explained and are accepted, whether they solve real-world problems, and whether the extension of industrial practice can be managed. This can be achieved by demonstrating AHEAD using a prepared demo session as described in the previous section, by discussing the underlying concepts with industrial partners, etc.

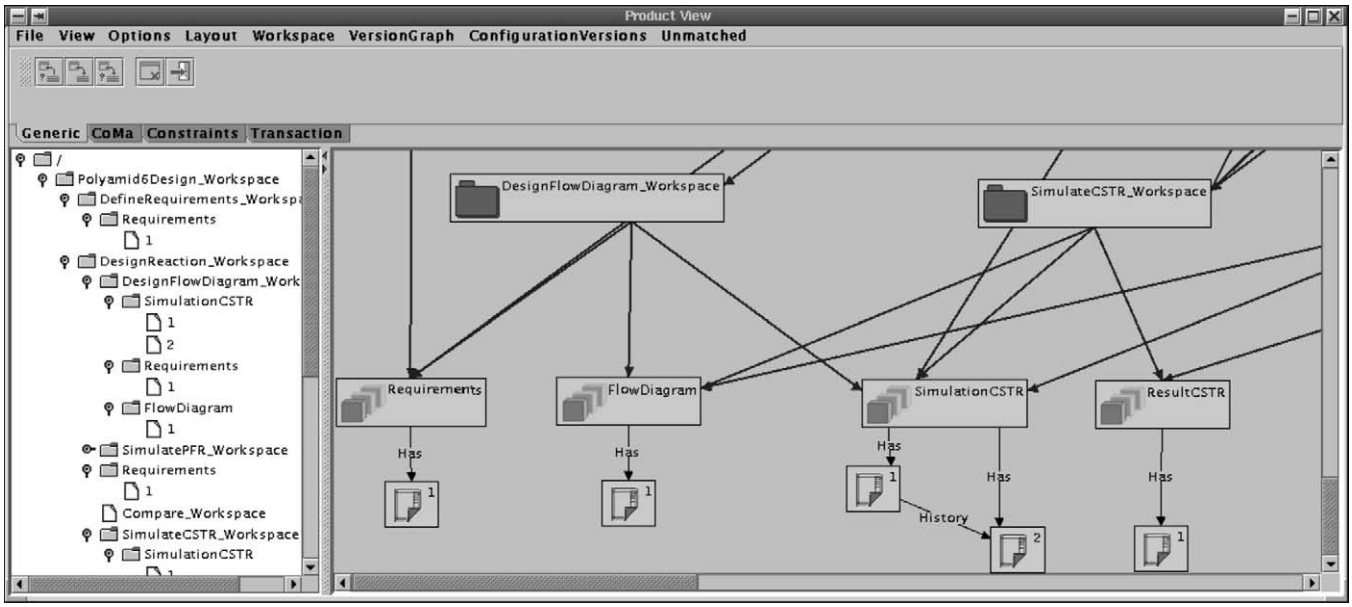

Fig. 17. Product view (management environment).

(2) *Evaluation of the demonstrator prototype* (transfer 2): Here, we plan the use of the AHEAD demonstrator for a small industrial project, again in a project together with an industrial partner. This requires that control on a medium-grained level is available in the company and there is the intention to manage that level. Discussions with different industrial partners, where the AHEAD system was demonstrated, have shown that the necessity of control on medium-grained level is seen. An exemplary use of the system also includes the evaluation of the user interfaces for the work environment, the management environment, and the modeling environment.

(3) *Realization of an industrial system with comparable functionality* (transfer 3): Comparable functionality here means that the requirements stated in Section 3 and Section 4 are met, or are at least partially met. So, we aim at getting as much functionality of the AHEAD system as possible or, at least, we do not loose essential functionality. Thereby, we take existing systems and integrate them on a bottom–up basis.

6.2. Different strategies and common problems

In the rest of this section we *concentrate* on *realization transfer* (transfer 3). There are *two ways* how an industrial system (with the functionalities of AHEAD) can look like.

The AHEAD system itself is acting as a *management integration instance*. Existing industrial systems are wrapped and incorporated into the integrated industrial system (Fig. 18).

We take again industrial systems for different purposes. On top of them a *new system* is realized which has essential AHEAD functionality. However, this system realization is using *industrial components* as well as a *conventional* industrial software development *process*, i.e., no rapid prototyping (Fig. 19).

For the rest of this subsection, we give some remarks and state some implementation problems which hold for both solutions (a) and (b). The reader should note that both ways *follow* the overall *bottom–up approach* of IMPROVE as explained in Section 1 (Fig. 1) inasmuch as existing tools are re-used.

Industrial *systems* to be used in a bottom–up approach have to be *stripped* in order to concentrate on their main functionality (separation of concerns). For example, a document management system is only used for the management of documents and not for managing activities and resources, although such systems may include some restricted functionality for that purpose. Otherwise, we would have to handle activity management in different industrial systems and in the integrated system on top as well. It is difficult enough to handle it both on the level of existing systems and on the integration instance level, where different existing systems are unified and integrated.

Both variants (a) and (b) introduce *wrappers* for existing systems. Wrappers have two aspects. First, we need a *functional interface* which is coarse- (start/end) or fine-grained (invoking all commands) depending on the openness of existing systems. Second, there is a *data interface* in the form of a view. So, we do not access the mostly cryptic internal data of an industrial application. Instead, we define a data abstraction layer by which we get rid of data structuring details. Of course, the data views of industrial systems have to be homogeneous with respect to granularity and structure.

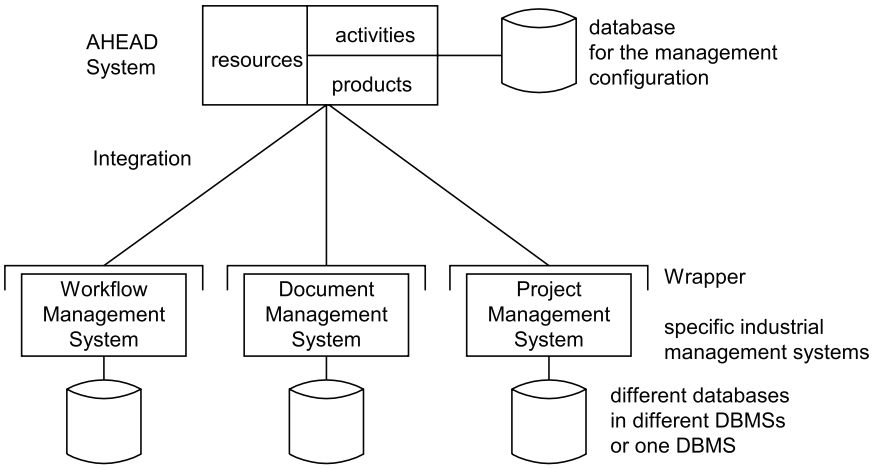

Fig. 18. The AHEAD system as integrating instance of commercial systems.

In both variants (a) and (b) we internally have to maintain and to retrieve the data of the *management configuration*. In variant (a), the management configuration is represented in the AHEAD system. In variant (b), the management configuration has to be built up by using views on existing systems. In addition, the mutual relations between these parts have to be defined (e.g., between activities and resources) in the second variant.

6.3. Discussion of both implementation alternatives

We now discuss the two ways of realizing an industrial system in more detail.

Way (a) can again be split into two variants. The first variant (a1) is to have the *full description* for products, activities, and resources within the management configuration of AHEAD. The second variant (a2) is that the AHEAD system only contains a *filtered* portion of the *data* of the existing systems *together with* some *integration* data. Then, the details are stored in the industrial subsystems.

In both cases (a1) and (a2) we find the *semantic models* of design processes in AHEAD at a *central place*, guaranteeing that the semantic submodels fit together. Within industrial systems a part of these models cannot be mapped (for example the integration of the three perspectives for products, activities, and resources). In solution (a1), we find the *detail information* for the submodels in the industrial systems as well as in AHEAD. As the access is simpler *via AHEAD*, we make data access and maintenance using this system. In solution (a2), we find only a part of the information in AHEAD. So, access of integrating information as well as coarse information is done via AHEAD, whereas the details are accessed *using* the *view interfaces* for industrial systems.

Let us discuss the *coupling* of *AHEAD* and the *industrial systems* by taking the task part of AHEAD and a workflow system as an example. As discussed in Section 3, workflow systems which allow to structure a task plan either do not regard dynamic aspects at all (static plans) or they allow subnet calling thereby

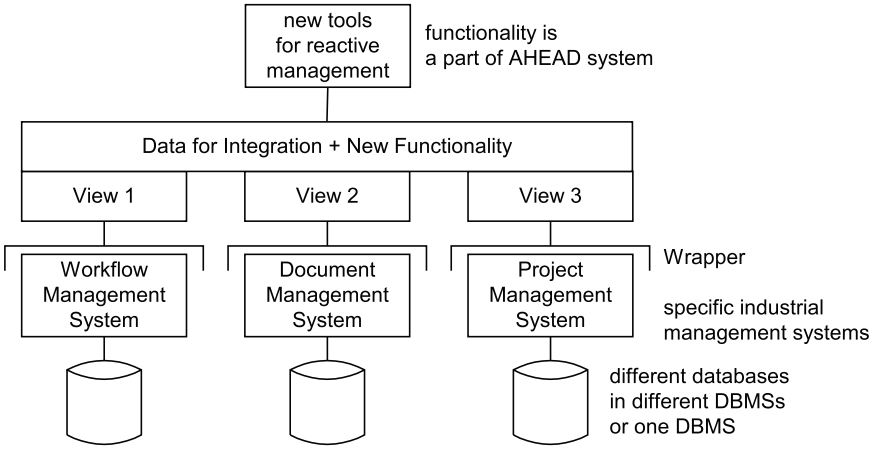

Fig. 19. Realization of AHEAD functionality using industrial infrastructure and systems.

distributing the static structure of a dynamically changing net over subnet definitions and the run time storage of the executing system. In solution (a1) the complete net structure is built up in AHEAD, the given workflow system is only used for executing the overall net or subnets. In solution (a2) only the global structure is kept within AHEAD whereas the detailed subnet structures are found in the workflow system. Thereby, execution is performed on a coarse level in AHEAD and on a more detailed level in the workflow system.

Let us now shortly *discuss alternative (b)* of above. Here, the AHEAD functionality of semantic submodels for products, activities, and resources, integration of submodels, interleaved invocation of structuring, analyzing and execution, adaptation mechanisms, etc. have to be re-implemented in the new industrial system on top of existing systems using only conventional software development techniques. This is not trivial. However, the essential features can be realized with some (remarkable) effort.

Comparing both solutions (a) and (b) we can state: Way (a) is much simpler, as the AHEAD system with its functionality is a part of the integration solution. However, the AHEAD system is just a demonstrator from academia and not an industrial system. So way (b) is more likely to be used. However, then the functionality of AHEAD has to be re-implemented, a nontrivial software development task.

As a consequence of the above discussion, the question now arises why we did not realize the AHEAD system by a bottom–up approach from the very beginning. The answer to this question is that we have firstly looked for the right concepts. In order to find them, the different possibilities had to be investigated. For playing around, a realization machinery as explained in Section 4.2 is much more convenient than coding different variants. Now, as a demonstrator is available, and if it has successfully passed the different stages of proof of concept (transfer 1 and 2), we are tackling the question of how an industrial system can look like. Our research is still continuing using the implementation machinery of Section 4.2 in order to quickly find further concepts/mechanisms which, again in a second run, are transferred to industry. Therefore, also in the future, we *distinguish* between *scientific investigation* on the one hand and *transfer projects* on the other.

7. Conclusion and outlook

We have presented the AHEAD system for managing design processes in chemical engineering. AHEAD goes beyond the functionality of commercial systems in various respects. Management is performed on a medium-grained level, allowing to effectively coordinate the work of designers. Products, activities, and resources are equally taken into account and are managed in an integrated way with the help of the management environment. Through its work environment, AHEAD is coupled with external tools supporting designers in their technical work. By seamless interleaving of planning and execution, AHEAD takes dynamic design processes into account. Finally, AHEAD may be adapted to different application domains with the help of its modeling environment.

Ongoing and future work addresses the following areas:

(1) *Technology transfer*: As discussed in Section 6, current work addresses the evolution of the AHEAD prototype into a system which may be used in an industrial context.

(2) *Improved functionality*: We are still extending the AHEAD prototype to improve its capabilities in various respects, e.g., concerning the management of distributed design processes (Becker, Jäger, Schleicher, & Westfechtel, 2001) or even more sophisticated support for process evolution (Schleicher, 2002).

(3) *Synergy*: We are integrating AHEAD more tightly with other components of the integrated design environment developed in the IMPROVE project. For example, we are designing an integration tool for coupling flow diagrams and task nets (so far, this coupling has to be performed manually), and we are investigating potentials for fine-grained process support for managers using the AHEAD system.

Acknowledgements

We are indebted to all researchers who have worked in the IMPROVE project and have contributed to the AHEAD system in one way or the other. In particular, we are indebted to A. Schleicher and D. Jäger, who have made essential contributions in their Ph.D. theses, and to B. Bayer, M. Eggersmann, and W. Marquardt.

References

Bañares-Alcántara, R. (1995). Design support systems for process engineering 1. Requirements and proposed solutions for a design process representation. *Computers and Chemical Engineering 19* (3), 267–277.

Bañares-Alcántara, R., & Lababidi, H. (1995). Design support systems for process engineering-II. KBDS: An experimental prototype. *Computers and Chemical Engineering 19* (3), 279–301.

Bayer, B., Eggersmann, M., Gani, R., & Schneider, R., (2002). Case studies in process design. In: Braunschweig & Gani, 2002.

Bayer, B., Marquardt, W., Weidenhaupt, K., & Jarke, M., 2001. A flowsheet centered architecture for conceptual design. In: Gani, R., & Jorgensen, S. (Eds.), *European symposium on computer aided process engineering 11* (pp. 345–350), Elsevier.

Becker, S., Haase, T., Westfechtel, B., & Wilhelms, J., 2002. Integration tools supporting cooperative development processes in chemical engineering. In: Ehrig, H., Krämer, B.J., & Ertas, A. (Eds.), *Proceedings of the sixth biennial world conference on integrated design and process technology (IDPT2002)*. Society for Design and Process Science, Pasadena, California.

Becker, S., Jäger, D., Schleicher, A., & Westfechtel, B., 2001. A delegation-based model for distributed software process management. In: Ambriola, V. (Ed.), *Proceedings eighth european workshop on software process technology (EWSPT 2001). LNCS 2077* (pp. 130–144). Witten, Germany: Springer.

Biegler, L. T., Grossmann, I. E., & Westerberg, A. W. (1997). *Systematic Methods of Chemical Process Design*. Upper Saddle River: Prentice Hall.

Blass, E. (1997). *Entwicklung verfahrenstechnischer Prozesse: Methoden, Zielsuche, Lösungssuche, Lösungsauswahl*. Berlin, Germany: Springer.

Bogusch, R., Loehmann, B., & Marquardt, W. (2001). Computer-aided process modeling with ModKit. *Computers and Chemical Engineering 25* (7–8), 963–995.

Böhlen, B., Jäger, D., Schleicher, A., & Westfechtel, B., 2002. UPGRADE: Building interactive tools for visual languages. In: Callaos, N., Hernandez-Encinas, L., & Yetim, F. (Eds.), Vol. I: Information Systems Development I. *Proceedings of the sixth world multiconference on systemics, cybernetics, and informatics (SCI 2002)* (pp. 17–22). Orlando, Florida.

Booch, G., Rumbaugh, J., & Jacobson, I. (1999). *The Unified Modeling Language User Guide*. Reading, Massachusetts: Addison Wesley.

Braunschweig, B., & Gani, R. (Eds.). *Software Architectures and Tools for Computer Aided Process Engineering*. Elsevier Publishers, Vol. 11.

Bullinger, H.-J., Warschat, J., *Concurrent Simultaneous Engineering Systems* (1996). Berlin, Germany: Springer.Springer, Berlin, Germany

Derniame, J.-C., Baba, A.K., Wastell, D., *Software Process: Principles, Methodology, and Technology*. *LNCS 1500* (1998). Berlin, Germany: Springer.

Deutsches Patentamt, 1969. *Verfahren and Vorrichtung zum kontinuierlichen Polykondensieren von Lactamen*. Patent number P 14 95 198.5 (B78577).

Douglas, J. M. (1988). *Conceptual Design of Chemical Processes*. New York: McGraw-Hill.

Eggersmann, M., Hackenberg, J., Marquardt, W., & Cameron, L., 2002. *Applications of modeling—a case study from process design*. In: Braunschweig & Gani, 2002.

Foltz, C., Killich, S., Wolf, M., Schmidt, L., & Luczak, H. (2001). Task and information modeling for cooperative work. In: M. Smith, & G. Salvendy (Eds.), Systems, Social and Internationalisation Design Aspects of HumanComputer Interaction. Volume 2 of the *Proceedings of HCI International 2001*. Lawrence Erlbaum Associates, Mahwah, New Jersey (pp. 172–176).

Georgakopoulos, D., Prinz, W., & Wolf, A.L. (Eds.), 1999. *Proceedings of the International Joint Conference on Work Activities Coordination and Collaboration (WACC-99)*. Vol. 24–2 of ACM SIGSOFT Software Engineering Notes. San Francisco, CA: ACM Press.

Gerrens, H. (1981). On selection of polymerization reactors. *German Chemical Engineering 4*, 1–13.

Harris, S.B. (1996). Business strategy and the role of engineering product data management: A literature review and summary of the emerging research questions. *Proceedings of the Institution of Mechanical Engineers*, Part B (Journal of Engineering Manufacture) *210*(B3) 207–220.

Jablonski, S., & Bußler, C. (1996). *Workflow Management-Modeling Concepts and Architecture*. Bonn, Germany: International Thomson Publishing.

Jäger, D., Schleicher, A., & Westfechtel, B. (1999). AHEAD: A graph-based system for modeling and managing development processes. In: Nagel, et al. (pp. 325–339).

Jäger, D., Schleicher, A., & Westfechtel, B. (1999b). Using UML for software process modeling. In O Nierstrasz & M. Lemoine (Eds.), *Software engineering-ESEC/FSE '99. LNCS 1687* (pp. 91–108). Toulouse, France: Springer.

Kerzner, H. (1998). *Project Management: A Systems Approach to Planning, Scheduling, and Controlling*. New York: Wiley.

Kiesel, N., Schürr, A., & Westfechtel, B. (1995). GRAS, a graph-oriented software engineering database system. *Information Systems 20* (1), 21–51.

Kohan, M., *Nylon Plastics Handbook* (1995). Munich, Germany: Carl Hanser Verlag.

Krapp, C.-A., Krüppel, S., Schleicher, A., & Westfechtel, B. (1998). Graph-based models for managing development processes, resources, and products. In: G. Engels & G. Rozenberg, *TACT '98-6th International Workshop on Theory and Application of Graph Transformation. LNCS 1764*, pp. 455–474. Paderborn, Germany: Springer.

Lawrence, P., *Workflow Handbook* (1997). Chichester, UK: Wiley.

Levy, S., Subrahmanian, E., Konda, S., Coyne, R., Westerberg, A., Reich, Y., 1993. An overview of the n-dim environment. Tech. Rep. EDRC-05-65-93, Carnegie Mellon University, Pittsburgh, Pennsylvania.

Lipperts S., Thißen D., 1999, CORBA wrappers for a-posteriori management: An approach to integrating management with existing heterogeneous systems. In: *Proceedings third International Conference on Formal Methods for Open Object-Based Distributed Systems.* Florence, Italy, pp. 273–280.

Marquardt, W., & Nagl, M., 1999. Tool integration via interface standardization? In: *Computer Application in Process and Plant Engineering-Papers of the 36th Tutzing Symposion*. Vol. 135 of DECHEMA Monographie. Wiley VCH, Weinheim, Germany, pp. 95–126.

McCarthy, J., & Bluestein, W., 1991. *The Computing Strategy Report: Workflow's Progress*. Forrester Research, Inc.

McGuire, M. L., & Jones, K. (1989). Maximizing the protential of process engineering databases. *Chemical Engineering Progress 85* (11), 78–83.

Mostow, J. (1985). Toward better models of the design process. *The AI Magazine 6* (1), 44–57.

Nagl, M., Schneider, R., & Westfechtel, B., Synergetische Verschränkung bei der A-posteriori-Integration von Werkzeugen. In: Nagl, M., & Westfechtel, B. (Eds.), *Modelle, Werkzeuge and Infrastrukturen zur Unterstützung von Entwicklungsprozessen*. Wiley VCH, Weinheim, Germany, in press.

Nagl, M., & Westfechtel, B., *Integration von Entwicklungssystemen in Ingenieuranwendungen* (1998). Heidelberg, Germany: Springer.

PIEBASE Working Group 2, 1998. Activity Model. Available from http://cic.nist.gov/piebase (Accessed 8 October, 2001).

Pohl, K., Klamma, R., Weidenhaupt, K., Dömges, R., Haumer, P., & Jarke, M. (1999). Process-integrated (modelling) environments (PRIME): Foundations and implementation framework. *ACM Transactions on Software Engineering and Methodology 8* (4), 343–410.

Ponton, J. (1995). Process systems engineering: Halfway through the first century. *Chemical Engineering Science 50* (24), 4045–4059.

Rudd, D., Powers, G., & Siirola, J. (1973). *Process Synthesis*. Englewood Cliffs: Prentice-Hall.

Schleicher, A. (1999). In: Nagl, et al., *Formalizing UML-based process models using graph transformations*. pp. 341–358.

Schleicher, A., 2002. Roundtrip process evolution support in a wide spectrum process management system. Ph.D. thesis, Aachen University of Technology, Aachen, Germany.

Schüppen, A., Trossen, D., & Wallbaum, M. (2002). Shared workspace for collaborative enginering. *Annals of Cases on Information Technology IV*, 119–130.

Schürr, A., Winter, A., & Zündorf, A. (1999). The PROGRES approach: Language and environment. In: H. Ehrig, G. Engels, H.-J. Kreowski & G. Rozen-berg. Handbook on Graph Grammars and Computing by Graph Transformation: *Applications, Languages, and Tools*, vol. 2: Applications, Languages and Tools. (pp. 487–550). Singapore: World Scientific.

Smithers, T., & Troxell, W. (1990). Design is intelligent behaviour, but what's the formalism? Artificial Intelligence for Engineering Design. *Analysis and Manufacturing 4* (2), 89–98.

Thayer, R.H. (1988). Software engineering project management: A top–down view. In: R.H. Thayer. *Tutorial: Software Engineering Project Management*, (pp. 15–54). Washington, DC: IEEE Computer Society Press.

Tichy, W.F. *Configuration Management of Trends in Software*, vol. 2 (1994). New York: Wiley.

Westerberg, A. W., Subrahmanian, E., Reich, Y., & Konda, S. (1997). Designing the process design process. *Computers & Chemical Engineering 21* (S), S1–S9.

Westfechtel, B. (1999). *Models and Tools for Managing Development Processes. LNCS 1646*. Heidelberg, Germany: Springer.

Whitgift, D. (1991). *Methods and Tools for Software Configuration Management. Wiley Series in Software Engineering Practice*. New York: Wiley.

Yin, R. (1984). *Case Study Research*. Beverly Hills: Sage Publicat-ions.