

7.7 Dynamic Process Management Based upon Existing Systems

M. Heller, M. Nagl, R. Wörzberger, and T. Heer

Abstract. In the past funding periods of the CRC 476 the process management system AHEAD has been developed as a research prototype (cf. 3.4). The project T6 aims at transferring the corresponding research results into two different industrial environments. Together with innotec GmbH, we realize a process management system on top of the chemical engineering tool Comos PT. This system will allow for the holistic management of the overall administration configuration (activities, products and resources). Furthermore, we extend our application experience by also considering dynamic business processes. In cooperation with AMB-Informatik, we build a workflow management system that allows dynamic changes of workflows at runtime. This new workflow management system is also built on top of an existing one, which strictly separates build-time from runtime.

7.7.1 Introduction

In IMPROVE the process management system *AHEAD* has been developed, as described in detail in Sect. 3.4. *AHEAD*'s main *characteristics* are (a) the medium-grained representation of the task structure, (b) the coverage and integration of processes, products, and resources at the managerial level, (c) the integration between managerial and technical level, (d) the support for dynamics of design processes, (e) the process management support across companies, and (f) the adaptability of the system to different domains.

Due to these features, *AHEAD* is *ahead of state of the art systems* for project, workflow, or document management. In particular, these systems cover only parts of the managerial configuration and do not integrate the managerial with the technical level. Common workflow management systems lack the support of dynamic changes within the task structure, which occur frequently in development processes.

Because of these shortcomings, the overall managerial configuration of a process is usually maintained manually. This induces inconsistencies in the administrative configuration, which may lead to, e.g., forgotten tasks or to the use of outdated documents. The risk of inconsistencies emphasizes the *need* for holistic *management support* which can be provided by *AHEAD*.

The *AHEAD* system is currently built using the tools *PROGRES* [414], *UPGRADE* [49], and *GRAS* [220], which allow for *rapid prototyping* based on graph transformations. Due to these technical dependencies, *AHEAD cannot easily be transferred* to arbitrary software platforms.

In the *following section*, we describe recent and future efforts concerning the transfer of concepts elaborated within *IMPROVE* into industrial environments. At first, we discuss two general approaches to transfer a system like *AHEAD* such that they integrate with existing systems for document,

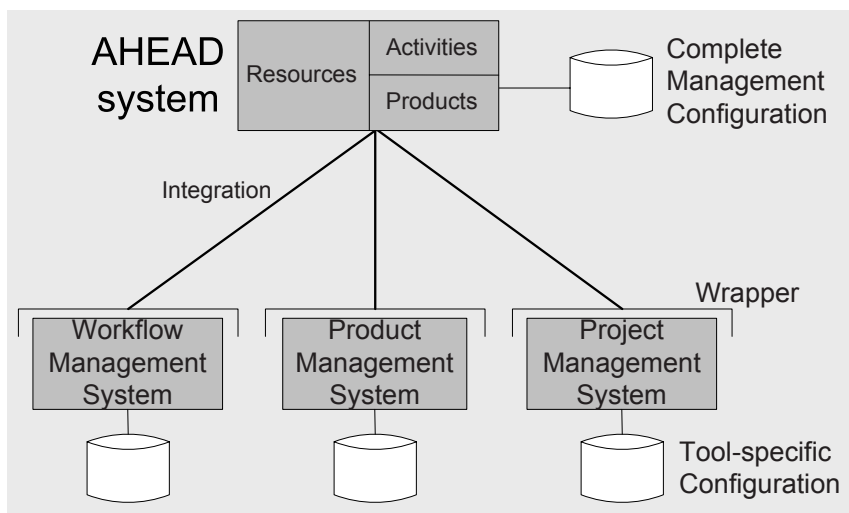


Fig. 7.21. Integration Approach 1 (IA1): AHEAD maintains full management configuration

workflow, or resource management. Thereafter, we depict preliminary work within the AHEAD project to facilitate technology transfer. Next, the implementation of an AHEAD-like system on top of an existing tool for chemical engineering is outlined. This is performed in cooperation with innotec, which develops and sells a tool named Comos PT. In an additional transfer activity together with AMB-Informatik, we show how AHEAD concepts can be applied to the domain of business processes and can be realized on top of an existing system for workflow management.

7.7.2 Strategies for A-posteriori Integration of Management Systems

The transfer into industry aims at the realization of an integrated, *industrial management support system* built up from existing tools. An AHEAD-like system is positioned on top of commercial tools as sketched in Figs. 7.21 and 7.22.

A-posteriori Integration

AHEAD offers advanced functionality for the management of dynamic design processes which is currently not available in commercial tools. As discussed in-depth in Sect. 3.4, process management covers several aspects including activity or workflow management, product management, resource management, and additionally time- and cost-based project management. For example, the storage of documents or the support for project planning is done with various

commercial tools. They constitute mature solutions for carrying out certain management tasks. Furthermore, users are familiar with these tools. Therefore, it is not feasible to replace these tools by a completely new and monolithic system.

Instead, *AHEAD* and the *commercial tools* are synergistically *combined* into a coherent overall system in a bottom-up fashion resulting in *two advantages*: (1) The commercial tools are embedded in a dynamic process environment. Thus, their functionality can be used in a coordinated manner to achieve new and advanced objectives. (2) *AHEAD* is extended with new functionality assembled from the commercial tools' functions. This avoids the need to re-implement already existing functionality offered by these tools. By the integration of commercial tools, the *AHEAD* system can be successfully "embedded" into the industrial technical and managerial overall environment.

The existing and coupled management systems are accessed by *wrappers* which provide an *abstract interface* to these systems. The wrappers comprise a functional interface to call functions and a data interface providing data views. The technical integration, including problems like data format conversion for example, can be encapsulated within the wrappers (cf. [136] and Sects. 5.7 and 7.8).

The functionality offered by commercial tools is often *stripped* in order to concentrate on the main functionality. For example, a product management system may also offer some functionality for managing activities in a design process. We decided not to use this functionality because a more versatile workflow management system is also to be integrated. Consequently, only one dedicated system is used for each management aspect (*separation of concerns*). In particular, activity, product and resource management are handled on the level of existing systems and on the integration level above.

Two Possible Solutions

The *storage* of the *management configuration* of the overall design process can be performed in *two possible ways*: The *AHEAD* system maintains the full management configuration with all details of the design process as shown in Fig. 7.21 (IA1). Alternatively, *AHEAD* stores only a part of the overall configuration together with additional integration data, while other parts are maintained in the existing systems and are accessible for *AHEAD* through views (IA2) as shown in Fig. 7.22. In the latter case, the *AHEAD* instance has to maintain all data relevant for the coordination of all systems on a more coarse-grained level, while other detailed data are left to be stored within the existing systems on a fine-grained level.

While the *AHEAD* system is used itself as integration *instance* in the first approach, the second approach implements *AHEAD-like functionality* on top of industrial management systems. For both alternatives, it is advantageous to locate *AHEAD's semantical models* of design processes at a *central place*, to guarantee that the semantical submodels fit together. These submodels

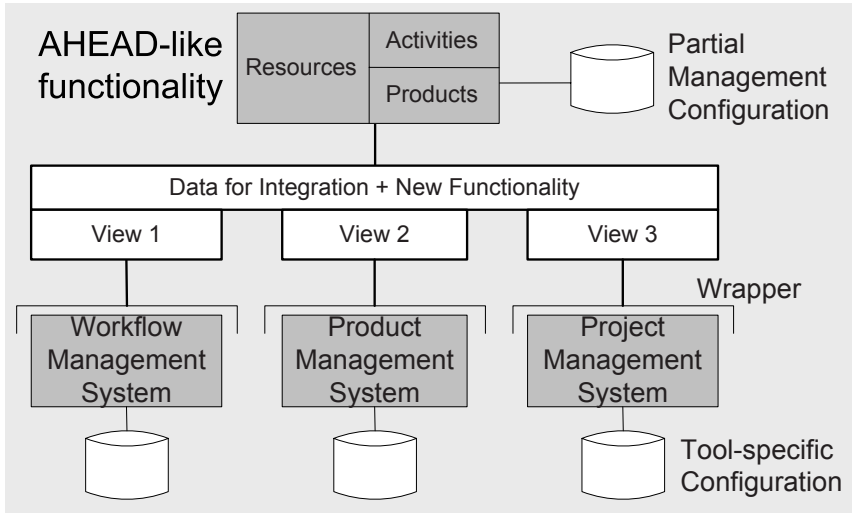


Fig. 7.22. Integration Approach 2 (IA2): AHEAD-like system merely coordinates existing systems

often differ from the models coming with existing systems. So, it cannot be guaranteed in general that all parts of the semantic models in AHEAD can be mapped onto that of existing systems. In some cases, data within the existing systems are not directly available in AHEAD, while in other cases all data are duplicated in AHEAD.

7.7.3 Preliminary Work

In the following subsection, we describe *preliminary work* for the transfer. We show how a *commercial workflow management system* can be used to *execute task nets* that are defined in the AHEAD system. After that, we describe the *integration* of the AHEAD system with *various existing systems* for project planning, email communication, and document management.

Workflow-Based Execution of Dynamic Task Nets

To further investigate the above mentioned integration alternative IA1, where AHEAD maintains the *full management configuration* for the integrated system, we have studied the *integration* of AHEAD with the *commercial workflow management system* COSA from Ley [208]. In this experiment, AHEAD was used for *planning* and *editing* of the overall task net structure, while the *execution* was *delegated* to the workflow management system. AHEAD uses *dynamic task nets* for process modeling, while COSA uses a *Petri net* variant for the same purpose. Thus, the *main problem* for the integration is the mapping of the dynamic task nets into Petri nets and respecting the semantics of

dynamic task nets during workflow execution within COSA. For the mapping, the structure and the dynamic behavior of dynamic task nets is mapped into a Petri net by mapping each modeling element of task nets into a small Petri net fragment consisting of places and transitions.

The integration of AHEAD and COSA is achieved in *two steps*. (1) The overall task net instance is stored in a task net description file. A *transformation module* implements this mapping and converts the task net into a COSA net which is stored in a COSA workflow description file. This file is imported into COSA where a corresponding workflow instance is created. (2) AHEAD and COSA are coupled at run-time using a *communication server* in between, similar to the coupling of two AHEAD systems for delegation-based cooperation as described in Subsect. 3.4.5. Both systems *exchange events* to keep each other informed about relevant process changes.

Changes of the process structure cannot be performed in COSA but *only in AHEAD*. The propagation of structural task net changes from AHEAD to COSA at run-time follows a *stop-change-restart-strategy*: the currently executing workflow instance is stopped in COSA after the current state has been stored; the obsolete workflow definition is changed and replaced with a new workflow definition containing the changed process structure; a new workflow instance is created according to the new workflow definition and is restarted after populating it with the stored process state.

The integration of AHEAD and COSA is *limited* with respect to the following aspects. First, the *full dynamic behavior* of task nets is not mapped to COSA; thus, the mapping is only implemented for a restricted part of the AHEAD model. Second, in order for the change propagation strategy to work, the old and new workflow definition of a changed workflow *cannot differ much*; only very limited structural changes can be accommodated to assure that the process state can be restored in the new workflow instance.

An Integrated Management Suite for Development Processes

We now describe preliminary work following *integration approach IA2*, which has been done during a half-year student project. In this project we have combined the AHEAD system with various commercial tools in order to form an integrated management suite. In this case, AHEAD does not maintain the complete management configuration. Parts of the *management configuration* are *distributed* across the commercial tools. For example, some details about documents are not contained in the product submodel of AHEAD but are rather stored in commercial product management systems.

The integrated management system provides the following synergistic functionality for its users reaching beyond the functionalities of the systems involved: (Fig. 7.23):

- The AHEAD system offers a *management* and a *work environment* to project managers and designers, respectively. Additionally, AHEAD pro-

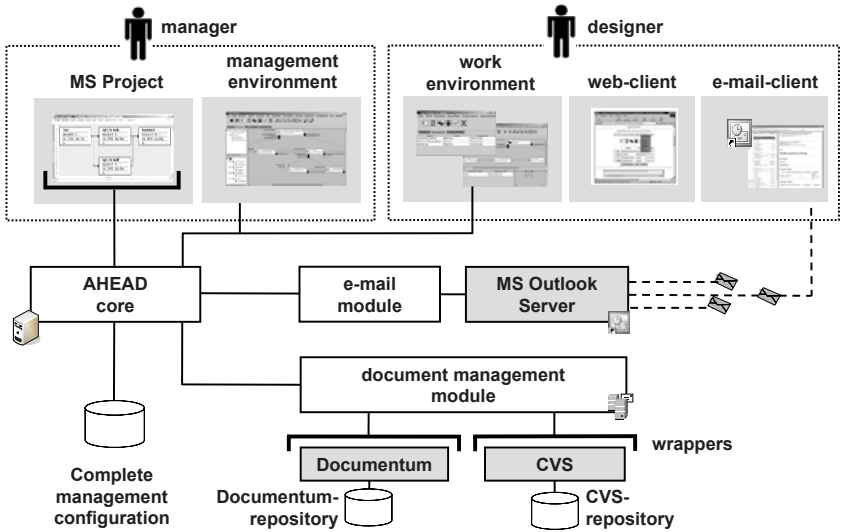


Fig. 7.23. Integrated management solution with AHEAD and commercial management systems

vides a *web-based client* for designers to communicate with AHEAD using a conventional web browser. All user interfaces access the AHEAD core which in turn stores the whole management configuration in its *own database*.

- On the *project management level*, the project manager can additionally use the *commercial tool* MS Project for the management of the design process from a project management perspective. For example, the overall task net can be imported within MS Project as a GANTT- or PERT-chart. The project manager can then use the *project management functionality* to calculate optimal start dates or end dates for the tasks using the critical path method, assigning resources to tasks etc. Finally, the results are exported back to the AHEAD system where the current state is updated.
- An *e-mail based work environment* is offered to project participants who prefer just to receive and to reply to work assignments via a common e-mail-client. An e-mail module of AHEAD realizes the e-mail communication, which is tailored to the commercial e-mail-client MS Outlook. Thus, *work assignments* are displayed as *Outlook activities*; necessary *documents* can also be received from and sent to AHEAD using e-mails. The state of the assigned tasks can be modified, too. An advanced *voting procedure* is also offered for the distribution of work within groups of participants.
- On the *product management level*, a central product management module is available in AHEAD which uses *wrappers* to *different product management systems* to maintain all documents employed throughout the whole

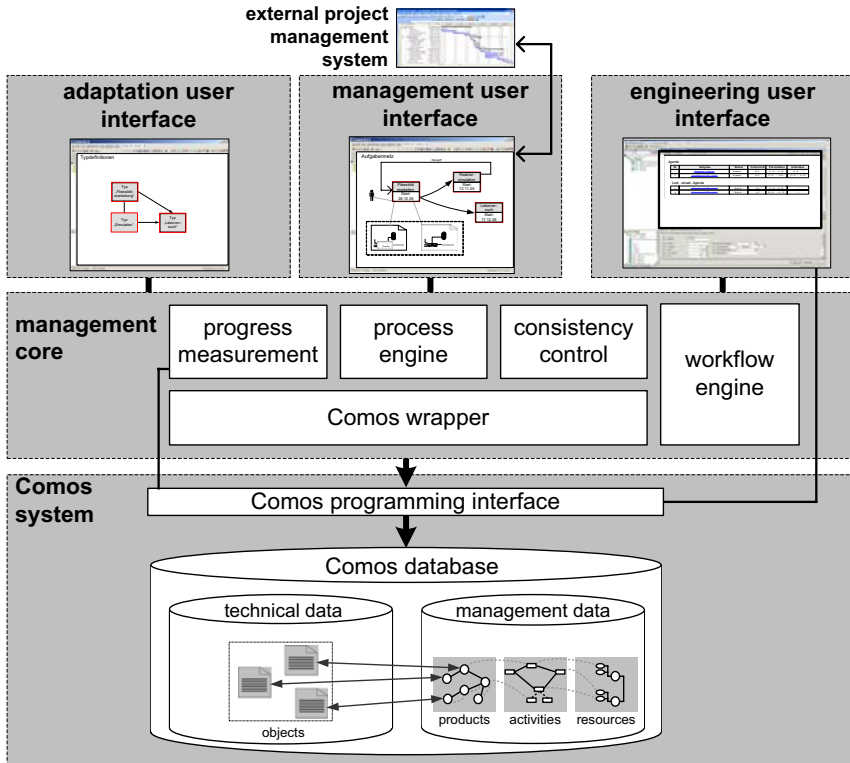


Fig. 7.24. Architecture of the planned management system for Comos PT

design process. This module is responsible for the mapping of logical documents used in the submodel for products in AHEAD to physical documents stored and maintained in the repositories of the coupled product management systems.

7.7.4 Management Tools for Chemical Design Processes

The following subsection gives a description of an integrated *process management system* which is based on an existing product management system according to approach IA2 (cf. Fig. 7.22). Its *realization* will be performed in *cooperation* with *innotec*, a partner in the transfer project T6.

Overview

Starting from the case study of the previous subsection, a process management system will be built to *extend* Comos PT, one of the tools for Computer Aided Engineering (CAE) and Process Lifecycle Management (PLM) supplied by

innotec to customers in the process industries (cf. subsection 7.6.3 for a short introduction).

For the time being, Comos PT strongly *focuses* on the *product part* of a design process. The tool is based on an *object-oriented database*, which allows to store all relevant parts of a chemical plant. All stored objects can be enriched with attributes and relationships between each other. Furthermore, Comos PT provides *dedicated views* on these objects (e.g. flow sheets) which show the two-dimensional layout of a (part of a) chemical plant. The support for the *coordination* of engineers is currently *restricted* to administrative attributes of objects, which represent the progress of a certain product according to a certain phase in the design process.

The planned process management system extends Comos PT by functionality for *holistic management* of chemical engineering design processes. It facilitates the coordination of engineers on a medium-grained level by introducing *integrated* submodels which cover not only the *product* structure but also the *task* and *resource* structure of the design process. Figure 7.24 depicts the *architecture* of this process management *system*.

The existing Comos PT system is located on the *lowest layer*. The Comos database stores all technical data within objects. Furthermore, it serves as the *central storage* for *management data* representing products, task, and resources as well as their (overlapping) relationships. Managerial data are strongly related to the technical data but abstract from technical details according to approach IA2 (cf. Fig. 7.22). Access to the Comos PT database can only be realized via the available Comos programming interface.

On the *middle layer*, the management core, an additional Comos wrapper provides an appropriate *abstraction* for accessing integrated management data within the *Comos PT database*. Changes in the integrated *management data*, due to normal progress or occurrence of dynamics, are handled by the *process engine*. Before being committed, changes have to pass a consistency control, which comprises *checks for inconsistencies* regarding the structure of the task net, the execution states of the tasks, as well as for the violation of time constraints. The progress⁶³ measurement component is used to *measure the progress* of individual tasks and ultimately the progress of the overall design process. Besides the management data, the progress measurement component also has to access the technical data stored in the Comos database, as the internal structure of certain flow diagrams can be used to estimate the effort of related tasks in the design process. Together with innotec GmbH, a workflow engine has been developed based on Microsoft's Windows Workflow Foundation technology. The workflow engine will be used to *automatically execute* parts of the overall design process. Hence, the workflow engine will constitute a part of the management core and will have to be seamlessly integrated with the process engine. Workflows will be started by the process engine, their progress will be measured by the progress measurement com-

⁶³ Note, that this is not related to the PROGRES system mentioned before.

ponent, and their compliance to defined time constraints will be checked by means of the consistency control.

On the *topmost layer* people can access the integrated management data via distinct graphical *user interfaces*. Engineers are notified of assigned tasks by a dedicated *engineering user interface*. Chief Engineers use the *management user interface* to fulfill managerial operations. This interface offers a complete view of the overall management data and provides functions for changing these data in case of dynamics.

Comos PT is mainly used in chemical engineering but can also be used in *other engineering domains*. So the planned process management system is especially designed to be *adaptable* to different domains and to be *integrated* with other tools of innotec. Domain experts adapt the process management system to a certain domain by using the adaptation user interface. This interface allows to declare *new types* of products, resources, activities, and to predefine best-practices as activity patterns. All user interfaces are integrated into the Comos user interface, so that the users do not have to switch between separate applications during their work.

Furthermore, *external project management systems* will be *coupled* with the process management system. For example, *MS Project* offers additional views of the design process like Gantt-charts with marked critical paths.

Problems and Solutions

A-posteriori Integration

The process management system AHEAD has been implemented as a *research prototype* in an a-priori manner. The effort for AHEAD's implementation was reduced by leveraging the high-level graph-based language and compiler PROGRES, the runtime environment UPGRADE, and the graph-oriented database GRAS. PROGRES, UPGRADE and GRAS are themselves research prototypes. Hence, this implementation is *not suitable* for being used in an *industrial environment*.

Instead of porting the current AHEAD implementation directly into the Comos PT environment, AHEAD is *reimplemented* as an extension of Comos PT with restriction to those programming languages and tools that are used within Comos PT. Transferring AHEAD in this way poses two interesting *challenges*:

1. AHEAD's core is specified by a PROGRES specification. This specification comprises integrated submodels for managing products, resources, activities, and definitions of valid operations on the integrated management data as graph transformations. This graph-oriented specification has to be *mapped* to a *common programming language* such as C++, C# or Visual Basic. The resulting code must fulfill two *requirements*: (a) It has to provide an *efficient implementation* of the specified tests and transformations, which are necessary to check the management data for *consistency*

and to perform *complex modifications* to this data, respectively. (b) The code must be *readable and extensible* for developers at innotec. This requirement does not apply for the compiled PROGRES specification that constitutes the AHEAD system.

2. At runtime, AHEAD reads and stores the current managerial configuration via the database system GRAS, which provides a *graph-oriented access* to the stored *data*. The process management system has to read and store management data to the *object-oriented database* of Comos PT by means of the Comos programming interface. The Comos programming interface is restricted, because it just provides a technical view of the stored objects. Therefore, a newly implemented Comos wrapper has to comply with two *requirements*: (a) It has to *bridge* between an object- and a graph-oriented view onto the management data. (b) The Comos wrapper has to provide an *extended view* onto the stored data, which additionally covers management data (resources, activities, and overlapping relationships).

Time Constraints, Expenses, and Staffing

Time constraints, expenses, and staffing do not play a predominant role in the current state of the AHEAD system. Nevertheless, they have to be considered in process management. Widespread *project management systems* like Microsoft Project offer *functions* like the computation of a critical path in an activity network, workload balancing for the staff or calculation of the aggregated expenses for (parts of) a project. Besides these available functions, project management systems offer *additional views* on the current project state, like the activity oriented Gantt-chart or diverse resource-oriented views.

In order to utilize this existing functionality, the process management system for Comos PT can be *coupled* with such existing systems. This can be done similar to the coupling of the AHEAD system with MS Project described in Subsect. 7.7.3.

Although common project management systems provide some useful functions for process management, they *cannot replace* the planned process *management system* of Comos PT since they lack the handling of dynamics and integrated management data. In particular, project management systems are unable to appropriately cope with *unexpected iterations* in the current projects or with *evolving activity structures*. Hence, these systems serve only for executing the described computations or rendering the views mentioned above but are not used as the main environment for process management.

Project management tools offer possibilities to define *time constraints* like deadlines for tasks. But most of these tools do not provide any means to *detect violations* of constraints and they do not *enforce actions* to bring the project back on schedule. Such issues will have to be addressed by our developments. The planned management system will permanently calculate the *current execution state* of the design process and will compare it with the

planned schedule of the process. In this way, violations of time constraints can be detected. Such violations can be *compensated* by *dynamic changes* of the project schedule at process runtime.

Progress Measurement

As mentioned before, *monitoring* the current execution state of a design process is an important feature of the planned management system. This includes the *measurement of the progress* of tasks and workflows. The integration of the management data with the technical data stored in the Comos database offers the opportunity to *estimate* the *required effort* and the *progress* of tasks based on the internal structure of the created documents.

The flow diagrams created during the design process determine to a certain degree the future tasks. In Comos PT, the *internal structure* of these *documents* is represented by objects. This data can be used for the calculation of the progress of certain tasks. Consequently, the progress is not solely calculated based on estimates of the engineers assigned to the tasks.

The calculated progress measures of elementary tasks have to be *aggregated* and *propagated* upwards in the task hierarchy. In this way, the progress of complex tasks, of project phases and ultimately of the whole project can be estimated. The progress measures help the project manager to decide whether the project is on schedule or whether deadlines will be missed.

Integration with the Workflow Engine

The workflow engine, developed in cooperation with innotec, is already used to *run and monitor workflows*, like the revision of documents or the handover of the final documentation of a chemical plant to the client. The provided *workflow management functionality* should be *used* by the process management system, so that parts of the design process can be *executed automatically* by the workflow engine.

The *integration* of the workflow engine into the management system core has to account for *several issues* such as the mapping of execution states of workflows and other subprocesses, the enforcement of time constraints for a workflow, and the measurement of its progress. Workflows can be started automatically, when the user performs certain actions in Comos. These workflow instances have to be integrated into the overall design process, i.e. their position in the task hierarchy has to be defined, and their relations to other tasks of the process have to be established.

Like all tasks of the design process, workflows have to be executed according to the *project time schedule*. When workflows are executed according to an overall schedule, it is possible to detect work-overload of resources, who participate in multiple workflow instances. The concepts for *progress measurement* of the design process have to be adapted and applied to the management of workflows. The resolution of the aforementioned issues leads to a *seamless integration* of the given workflow engine with the new process management system.

Impact

The implementation of the process management system for Comos PT is interesting from both, the *economic* and the *academic* point of view. The leading position of Comos PT is further improved by a process management system, which allows for the holistic management of the overall design process of a chemical plant.

From the *research perspective* the transfer of the AHEAD system into an industrial environment is challenging w.r.t. several aspects: The process management system has to be realized *on top* of an existing software-system that has not been designed for process management. In contrast to the AHEAD system, graph-based prototyping tools PROGRES, UPGRADE and GRAS cannot be used for its implementation. Instead, the given specification of AHEAD has to be mapped methodically to common programming languages and the graph-based storage of management data must be implemented using an object-oriented database.

Dynamics in design processes have to be addressed with particular emphasis on evolution and iterations of activities, but also on deadlines and expenses.

7.7.5 Dynamic Workflow Management for Business Processes

The following subsection describes the realization of a workflow management *system* which provides support for *dynamic business processes*. Following again approach IA2 (cf. Fig. 7.22), this system is based *on top* of an existing workflow management system, which only supports *static* business processes. This objective constitutes the second part of the transfer project T6 and is performed in collaboration with the IT service provider *AMB Generali Informatik Services GmbH*.

Overview

Although AHEAD has been designed within the subproject B4 to capture a multitude of processes, it has always been focusing the management of design processes, while business processes have not been considered in the past. In cooperation with AMB-Informatik, research results of the B4 subproject are *transferred* into the domain of *business processes*. AMB-Informatik is a full-service information technology provider for the insurance group AMB-Generali.

Figure 7.25 shows the coarse grained *architecture* of the planned system. The architecture is divided into three parts, which are described in the following.

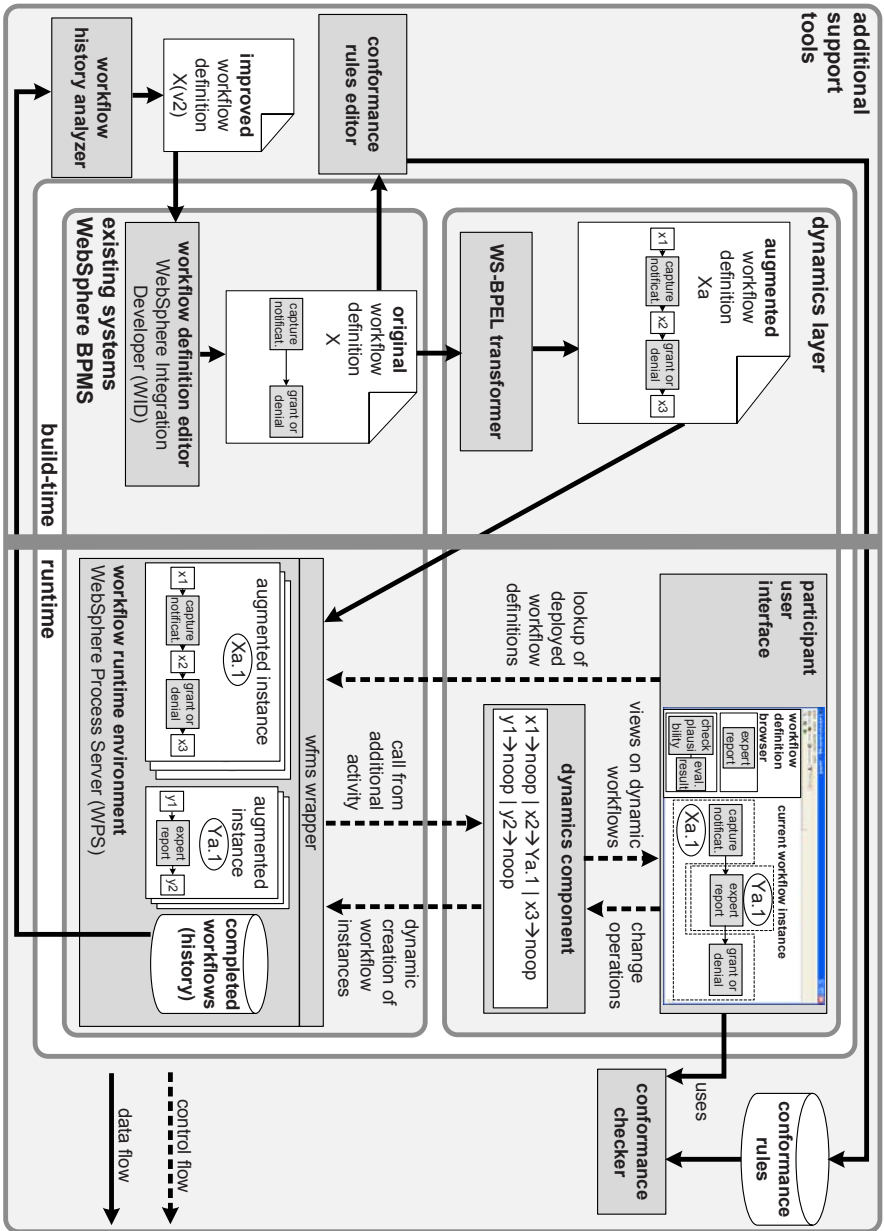


Fig. 7.25. Architecture of the AMB-Informatik workflow management system

Existing Systems

Currently, AMB-Informatik uses IBM's product family WebSphere BPMS in order to support their business processes. WebSphere BPMS [1034] *strictly separates* build-time from runtime of a workflow. Hence, *dynamic business processes* with undeterminable activity sequences *cannot be supported* appropriately. In general, these processes are likely to become dynamic if different people are involved, especially outside of the insurance company.

IBM WebSphere Integration Developer (WID) and IBM WebSphere Process Server (WPS) are the *existing systems* which are relevant for the transfer project. WID can be used at build-time to define workflows consisting of activities that are executed automatically or by human interaction. The workflow definitions are written in the de facto standard "Web Services Business Process Execution Language" (WS-BPEL) [887]. At runtime these workflows can be executed within WPS, but can *neither be altered* within their predefined sequences of activities *nor extended* by additional activities.

Dynamics Layer

The existing systems are extended by a dynamics layer to *facilitate dynamic changes* to workflows in the first place [487]. Roughly speaking, the components of this layer simulate dynamic workflows while the underlying WebSphere BPMS still supports only static workflows.

The distinction between build-time and runtime can also be found in the dynamics layer. At build-time, there is a *WS-BPEL transformer* which adds additional WS-BPEL activities, to an original workflow definition X resulting in an augmented workflow definition Xa. In Fig. 7.25 activities x1, x2, and x3 represent such additional activities.

At runtime, execution of an additional activity triggers a call to the *dynamics component*. This component stores workflow *instances' specific information* about how to handle a call from an additional activity. An additional activity can invoke a newly instantiated workflow instance depending on the respective information stored in the dynamics component. For example, in Fig. 7.25 additional activity x2 invokes workflow instance Ya.1

Since the participant user interface of WPS displays only small parts of the overall workflow in table form, it is inappropriate for supporting dynamic workflow changes. Thus, a new *participant user interface* has to be implemented from scratch which displays the *overall structure* of a workflow instances in a graphical manner. Additionally, workflow instances are displayed in a *condensed manner*, i.e., the additional activities remain hidden and dynamically called workflow instances are displayed inline within the calling workflow instances. Furthermore, *predefined workflow definition* fragments can be selected and inserted by a participant via a *workflow definition browser*. Altogether, the workflow participant experiences a dynamic workflow although the technical basis WPS remains static.

Additional Support Tools

Before a *dynamic change* of a running workflow instance takes effect, the modification is *checked* against certain conformance rules by the *conformance checker tool*. If a check fails, an error message is displayed in the participant user interface, where the change request originated from. Otherwise, the change of the workflow instance takes effect. The conformance rules that are used by the conformance checker originate from a build-time tool named *conformance rules editor*. A workflow modeler can use this editor to define rules concerning dynamic workflow changes like the non-deletability of certain (mandatory) workflow activities.

In order to benefit from the *implicit knowledge* expressed by dynamic workflow changes, completed workflow instances can be analyzed by a *workflow history analyzer* with regard to dynamic changes. A finished workflow instance can be imported into the *workflow history analyzer* and compared with its original workflow definition. Then, it is up to a workflow modeler to *decide* whether a deviation of the instance from its workflow definition is *generic*, and should be part of a *improved workflow definition*, or *special* to the respective instance, and should therefore be kept out of the definition.

Problems and Solutions

Support for Dynamic Changes at Runtime

In contrast to design processes, business processes are more *repetitive* and contain *more static parts* in their overall activity structure. Therefore, major parts of business processes can be *predefined*. Hence, typical dynamic situations are not only evolution of activity structures in a running workflow instance but also *deviations* from prescribed activity structures.

Business processes are likely to be *constrained* by laws or company-specific rules. These rules may demand the execution of certain activities, which therefore must not be removed from a workflow instance.

The planned workflow management system for AMB-Informatik meets both issues. First, it provides support for the *definition at build-time* of workflows and conformance rules via the conformance rules editor. Second, laws or domain specific regulations are *enforced at runtime* by checking dynamic changes of a workflow instance by a conformance checker before these changes actually take effect in the workflow instance.

Process Improvement

As described above, the workflow management system for AMB-Informatik offers functionality for predefining workflows at build-time and for changing workflow instances at runtime. There are *dynamic changes* of *two types*: (1) A dynamic change in a workflow instance might occur as a deviation from the workflow definition due to a *special situation* in a business case that is

associated to the workflow instance. (2) A dynamic change in a workflow instance might take place because of an *insufficient workflow definition*.

In the latter case, it is probable that the dynamic change has to be done over and over again for each workflow instance of the corresponding workflow definition. Since completed workflows can be imported into the *workflow history analyzer* and compared to the original workflow definition, a workflow modeler is able to recognize recurring deviations from the workflow definition and can *adopt these deviations* by adding them to the definition. In this way, business processes can be significantly improved.

A-posteriori Integration

Like the process management system for Comos PT, the workflow management system for AMB-Informatik resides on top of an existing system. Hence, it has to be realized in an a-posteriori manner. In contrast to Comos PT, which focuses on product-related aspects of design processes, WebSphere BPMS mainly covers *activity-related parts* of a business process. WebSphere BPMS is therefore used as a *building block* for the new workflow management system for AMB-Informatik. It should be possible to substitute this block.

Impact

The realization of the workflow management system is *beneficial* both in *economic* and *research* respect. The workflow management system empowers AMB-Informatik to *support dynamic business processes*, which WebSphere BPMS does not provide itself. The workflow management system does *not replace* WebSphere BPMS but extends it such that subsequent integration problems can be avoided.

From the *research* perspective, new interesting question emerge by transferring existing concepts for design processes to the domain of business processes. The consideration of dynamics in processes will *shift* from *continuously evolving design processes* to more repetitive but nonetheless *dynamic business processes*.