Towards an Integrated Modelling Framework for Engineering Design Processes

Michalis Miatidis[1]
[1]Informatik V (Information Systems)
RWTH Aachen University, Germany
miatidis@cs.rwth-aachen.de

Matthias Jarke[1,2]
[2]Fraunhofer FIT, Schloss Birlinghoven
Sankt Augustin, Germany
jarke@cs.rwth-aachen.de

Abstract: Engineering design processes consist of an administration level where the coordination of the design workflow takes place and a process level where human actors perform their tasks following engineering methodologies. Most of the existing information system infrastructures support either of these two levels, resulting in isolated islands of support. In this paper, we present an integrated modelling framework that bridges the gap between these two levels. The key idea behind this framework is a multi-perspective modelling language that adequately captures both levels by interrelating them according to a number of semantic bridges. Our approach has been evaluated in an interdisciplinary chemical engineering project focusing on the conceptual design and basic engineering phases of the nylon production.

1 Introduction

Engineering design is arguably one of the most knowledge-intensive tasks undertaken by engineers. It primarily includes the original specification, simulation and evaluation of several design alternatives that mould the properties of the final product. Its inherent engineering nature manifests itself in the high degrees of creativity, complexity and unpredictability that make the followed engineering methodologies seldom clearly defined.

From a business perspective, the efficient and effective planning and coordination of the engineering design process is of great importance to the competitiveness of the enterprise. It has an impact on most of the 'sensitive' characteristics of the overall production like final product quality, cost of overall investments, duration of operations, as well as environmental impacts.

As a consequence, support to engineering design involves an intricate interplay of determinism demanded by contemporary business trends and flexibility because of its highly creative nature. In this paper, we present an integrated modelling framework that attempts to reconcile these two natures by providing a multi-perspective support.

The structure of the paper is as follows. In Sect. 2, we present the motivation for our approach. Section 3 introduces the key idea behind the proposed framework by elaborating the integrated modelling language it builds on. Our overall approach has been validated by a chemical engineering case study in the frame of a large interdisciplinary project (Sect. 4). Finally, in Sect. 5 we draw some conclusions and provide an outlook to future work.

2 Motivation

In this section, we provide the motivation for the development of our integrated modelling framework. We first identify the two levels of support to engineering design and subsequently, we detail the need for bridging them.

2.1 Administration and Process Levels of Support

Engineering design is a process of strategic importance that has been widely recognized inside the hierarchical structures of contemporary manufacturing enterprises [Ver96]. Figure 1 sketches the two prominent levels of any traditional engineering design process and its enterprise settings. We mainly refer to enterprises of the *process sector* (e.g. chemicals, food and oil). At the highest level, we identify the *business subsystem* where the overall enterprise goals, requirements and business policies are set. This high level subsystem will not concern us in this paper.

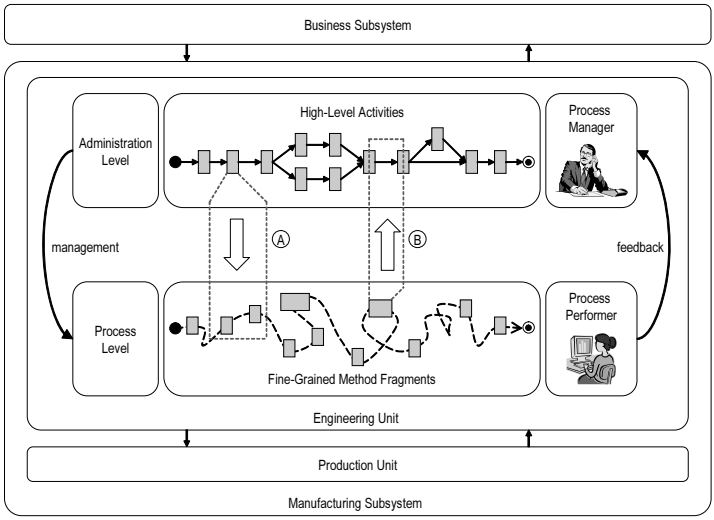

Figure 1: Enterprise settings of engineering design and a zoomed view inside its two levels of support

At the lower level resides the actual *manufacturing* subsystem decomposed into the *engineering* and *production* units. The engineering unit provides the environment where the design processes addressed in this paper are performed. According to the recursive management paradigm [AH02], the engineering unit can also be viewed as a smaller *virtual enterprise* that is further decomposed into two hierarchical levels: the *process level* where the actual design process is being performed and the *administration level* where the planning and coordination (*'management'* link) of the process takes place. Following the planning and control cycle, the process level (managed system) gives back feedback

reports (*'feedback'* link) to the administration level concerning its enactment status. Based on these reports, original managerial decisions, goals and policies are revised. The output of the design process is a set of requirements and specifications for the final production of the plant by the production unit.

At the administration level, the design process is considered from the business perspective. Here, the primary concern is the process-centric automated coordination of the interworking of the human actors. The whole process lifecycle is described in the coarse-grained and properly-formalized way expected for the managerial needs. To this end, a well-structured *workflow model* is employed that materializes the higher business rules and policies. The workflow model describes with precision the decomposition of the design process into well-defined activities assigned to experts, as well as the control and information flow among them (zoomed view of the administration level in Fig. 1).

At the process level reside the human actors who perform the design supported by domain-specific tools (e.g. CAD and simulation tools). These actors exploit their creativity by following various engineering *methodologies* depending vastly on their knowledge background and previous experience. Because of this non-determinism and high degree of freedom, the process trace is *poorly-structured* and evolves arbitrarily (i.e. no design process instance will be completely identical to an other). Only few of the engineering methodologies can be adequately formalized and captured by a fine-grained process metamodel. We call these well-understood methodologies *method fragments* (zoomed view of the process level in Fig. 1). Method fragments, contrary to workflow activities, cannot be organized in a networked topology. They are rather related to the conceptual *situation* that they can methodically guide the human actor.

2.2 Bridging the Gap

In the literature, we can identify many contributions aiming at building support infrastructures for engineering design processes. In the arena of the administration support, business process management methods are usually applied by generic *workflow management systems* [AH02]. For the process level support, in the area of chemical engineering that we focus in this paper, we can identify various domain-specific paradigms for the creation of design support infrastructures (e.g. [SKD+97], [JD94]).

Each of the above contributions provides adequate support for the needs of the support level it is intended. Nevertheless, the implications of the provided support to the farther side are ill-defined (if at all existent). As a consequence, these two classes of infrastructures constitute *isolated islands of support*. Their lack of a shared understanding of the design considerably hinders the unobstructed communication between the managerial and the engineering teams. Dissemination of project-specific information becomes anaemic, and the design rationale behind the managerial decisions is not adequately captured and provided to the human actors.

Several strategies have been recently pursued to bridge this gap. Van der Aalst et al. recognize the limited operational flexibility of traditional workflow management systems for

highly-dynamic knowledge-intensive processes (like engineering design), and proposes a paradigm shift towards product-driven workflows [AB01]. Other approaches have tried to make administration support systems more flexible by run-time exception handling and workflow adaptability. For example, the *AHEAD* management system developed at the RWTH Aachen University [HJS+04] represents workflows using graph grammars and employs graph rewriting techniques in order to introduce flexibility. However, there is no real support in terms of fine-grained method advice for the human actor.

Other researchers have started from the process level and tried to embed the detailed method fragments in higher-level strategies that are formally modelled, e.g. as non-deterministic automata in which human actors can navigate [RR01]. However, this approach, while reaching a higher granularity in the technical process work, does not really address the administrative issues covered by typical workflow models.

We believe that an effective and efficient approach for bridging the gap between the two levels should keep a multi-perspective viewpoint on the design process. The business and engineering natures of the process should remain intact in order to avoid capital- and labour-intensive rework efforts. The process rigidity demanded by the managerial trends should not suppress the engineering flexibility of the design process itself. Coordination and execution should rather seamlessly interweave.

To this end, we argue that any successful approach for bridging the gap should follow a multi-perspective bottom-up methodology. Such an approach should build on two formalisms (modelling languages) for the description of the two sides. These languages should, as much as possible, conform to well-established standards commonly adapted by enterprises. Since these two languages would refer to the same process from a different level of abstraction, they will definitely include some overlapping elements. This overlap is intentional: it is exploited for bringing the two sides closer to each other and promotes a shared understanding. *Concept mappings* should be defined between them in order to facilitate the dissemination of information from one level to the other. The resulting design *environment model* will encapsulate the high-level *'whats'* (activities) of the process and the precise *'hows'* and *'whys'* of the method fragments followed during the workflow enactment.

The middle part of Fig. 1 shows the two prominent patterns for such mappings. *Pattern A* reflects the interrelation between an activity and the method fragments representing the engineering methodologies followed during its enactment. The direction of this relation (administration towards process level) implies the interpretation of the managerial decisions into engineering methodologies. *Pattern B* indicates a method fragment that provides the methodology for the transition from one activity to the other. The direction of this relation (process towards administration level) implies how an engineering methodology influences the workflow enactment.

Based on these considerations, in earlier work we have experimented with more or less ad-hoc implementation techniques of integration tools that defined mapping rules between the constructs of typical process support tools such as flowsheet editors, and the aforementioned AHEAD system [SFB00]. In a separate attempt, we have also tried to define method fragments within the process perspective which view the administrative work-

place as just another engineering workplace which could be supported by process-level method guidance. However, this approach of simulating administrative work indirectly at the workplace process level again did only partial justice to the requirements of project administration and, in particular, to the need of being able to utilize different, standard-compatible workflow engines for administrative purposes.

In the sequel, we therefore describe a metamodeling approach that retains our basic approach to process-level method guidance but links it to workflow standard models rather than building idiosyncratic special-case solutions.

3 Key Idea: Integrated Modelling Language

In this section, we elaborate on our modelling language by describing the process and workflow metamodels along with the concept mappings we have developed between them.

3.1 Process Metamodel

Our existing process metamodel builds on a situation-based contextual metamodel initially proposed in the *NATURE* requirements engineering project. [JRSD99]. Fig. 2 introduces the key concepts of the metamodel using the UML notation.

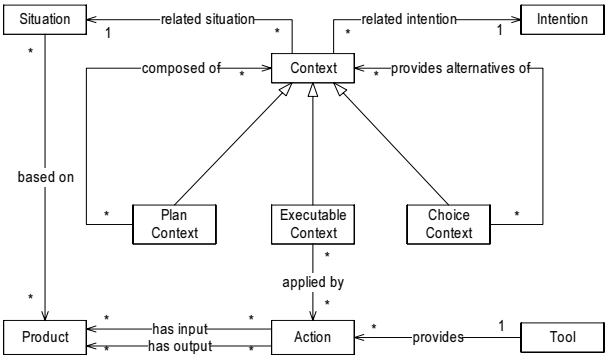

Figure 2: NATURE-based process metamodel

This metamodel is based on the assumption that, in highly creative processes the human actors tend to react contextually according to their accumulated experience and in analogy with previous similar situations they have been involved in. Therefore, it explicitly represents situations, intentions and contexts. A *situation* describes the subjectively perceived state of the process and is based on the individual states of the *products* undergoing the development. An *intention* reflects the goal that the human actor has in his mind.

The process knowledge of how to reach a specific intention in a given situation is captured by the *context*. Thus, a context is the NATURE-based abstraction for the method fragments that can be provided to the human actor in order to methodically guide him through his activity enactment.

A context can be refined into three categories. *Executable contexts* describe pieces of the process that can be automated and usually have a corresponding *action* provided by a *tool*. *Choice contexts* represent the most creative parts of the process where the human actor comes across the need for a decision among several alternatives in order to reach his goal. Strategies and systematic plans are defined by *plan contexts* that may recursively contain other contexts of all three types.

3.2 Workflow Metamodel

For our workflow modelling needs, we have adapted a workflow metamodel compatible with the *Workflow Management Coalition (WfMC)* reference metamodel standard [WfM01]. WfMC has developed specifications for standards that concentrate common characteristics of workflow management products. As a result, these specifications improve the opportunities for the effective integration of commonly used workflow concepts and thus promote the interoperability with other workflow management systems. The resulting metamodel is shown in Fig. 3 using the UML notation.

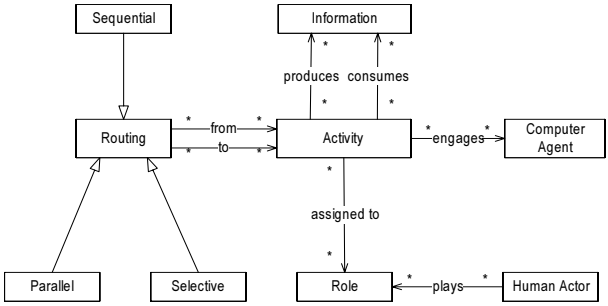

Figure 3: WfMC-based workflow metamodel

Following the WfMC reference metamodel, the logical pieces of work that require the support of human and machine resources for their execution are captured in the *activity*. An activity can be arbitrarily complex and be recursively composed of others until the *basic activity* level is reached (not shown in Fig. 3).

Each activity requires a number of *resources* in order to be carried out. We distinguish two kinds of them: *human actors* (i.e. human performers of the activities) and *computer agents* (i.e. tools). Human actors are indirectly associated to their assigned activities via their *roles*. Roles are distributed according to the knowledge background, skills and assigned responsibilities of the human actors playing them. During the activity enactment, human

actors and computer agents work on objects of the real world that are produced, consumed or transformed (e.g. products, project specification, simulation results and mathematical models). All these objects are captured as pieces of *information*.

The *routing* element models the different ways of control flow organization among activities. Three basic routing types (specializations) are distinguished: *sequential routing* when the activities are carried out one after the other, *parallel routing* when more than one activities can be active at the same time (i.e. *AND-split* and *AND-join*) and *selective routing* for the situation when a specific activity among several others has to be selected based on the evaluation of preconditions (i.e. *OR-split* and *OR-join*).

3.3 Concept Mappings: Building the Environment Metamodel

Our first prototype for the environment metamodel is based on a mathematical representation of the concept mappings. This representation empowers more sophisticated reasoning mechanisms on the internal consistency of the metamodel through the use of a number of integrity constraints. The formal expression of the metamodel follows.

An environment metamodel $\mathcal{E}$ is defined as a 3-tuple: $\mathcal{E} = \langle W, P, CM \rangle$, where W is the workflow metamodel, P is the corresponding process metamodel and CM are the concept mappings developed between the two metamodels. The workflow and process metamodels can be formalized as N-tuples containing instances of the elements they comprise along with the set of their respective relationships. The concept mappings can be grouped into three disjoint categories: the functional mappings M_f, the information mappings M_i and the resource mappings M_r (i.e. $CM = M_f \cup M_i \cup M_r$).

We define a *functional mapping* between an activity and the contexts representing the method fragments that can be followed during its enactment. A functional mapping class M_f is defined as: $M_f \subseteq A \times C \times S$, where A is the set of activity classes of the workflow metamodel and C is the set of context classes of the process metamodel. This mapping carries some *semantics* S for providing important shared knowledge affecting both of the levels, like time scheduling constraints, project specific values, or even signaling status information concerning the workflow enactment. For example, by mapping a context with an activity using 'starting' semantics, by the time that the process manager delegates it to a human actor, the context can automatically be enacted and methodically guide him. Conversely, the execution of a context with 'terminating' semantics can directly provide notification to the manager about the termination of the activity.

In the workflow metamodel, products are considered as pieces of information at the level of document (e.g. flowsheet diagrams). In the process metamodel, on the other hand, products are further detailed to their structure since their properties are changed by tool actions (e.g. a flowsheet diagram is refined to its devices, streams, chemical components etc). Thus, the following *information mapping class* M_i is derived: $M_i \subseteq I \times P$, where I is the set of information classes of the workflow metamodel and P is the set of product classes of the process metamodel.

Likewise, computer agents are refined into tools in terms of their capabilities (services provided to their user) for the transformation of the considered products. The following *resource mapping class* M_r is derived: $M_r \subseteq CA \times T$, where CA is the set of computer agent classes of the workflow metamodel and T is the set of tool classes of the process metamodel.

Besides the above three mapping classes, the environment metamodel facilitates consistency checks through a number of constraints that we have distinguished. These constraints can be grouped into two categories based on whether they check the 'completeness' or the 'correctness' of the environment model.

The prominent *completeness constraints* are as follows:

C_1 *assure that each context is mapped to at least one activity*: Given a context c, then there exists an activity a that is mapped to the c.

C_2 *assure that a product is mapped to at least one information item*: Given a product p, then there exists an information item i that is mapped to the p.

C_3 *assure that a tool is mapped to at least one computer agent*: Given a tool t. Then there exists a computer agent ca that is mapped to the t.

The prominent *correctness constraints* are as follows:

C_4 *assure that the information items mapped to the products that a related situation of a context is based on are consumed by all the activities that the context is mapped to*: Given a context c and the set of products p_c that the situation related to the c is based. Let a_c be a set of activities that are mapped to the c and i_a the set of information items consumed by the a_c. Then, every product in the p_c must be mapped to an information item in the i_a.

C_5 *assure that all the tools providing actions for a context are mapped to computer agents engaged during an activity that this context is mapped to*: Given a context c and the set of tools t_c that provide actions for the c. Let a_c be a set of activities that are mapped to the c and ca_a the set of computer agents engaged by the a_c. Then, every tool in the t_c must be mapped to a computer agent in the ca_a.

4 Validation

Based on the integrated modelling language outlined in the previous section, we have defined a generic integration platform architecture for the support of engineering design. In the following, we briefly describe this architecture and illustrate its effectiveness on an initial case study we are conducting with colleagues from chemical engineering.

4.1 Integration Platform Architecture

In the last years, we have designed and implemented a process support environment for engineering design processes. This environment is built on top of the *PRIME* (PRocess-Integrated Modelling Environments) implementation framework that empowers situated method guidance (using the NATURE-based process metamodel of Fig. 2) through process-integrated tools [PWD+99].

PRIME comprises three conceptually distinguishable domains. The *modelling* domain contains the so-called *Process Data Warehouse* (PDW) for the storage and the maintenance of the environment model definitions of the design process [JLK00]. The enactment domain contains an *enactment mechanism* (i.e. a *process engine*) that, based on the interpretation of the process definitions, drives the process enactment at the engineering workplaces where the actual design process is performed (*performance domain*). Project-specific method guidance can be directly offered to the human actors through their tools by process-integrating them using appropriate wrappers.

The original PRIME framework has been reengineered in order to become an integration platform for the interpretation of our newly-developed environment metamodel. The integration of our extended semantics to the existing modelling formalism (i.e. workflow metamodel and concept mappings) has imposed the following two implications to the original architecture.

At first, we had to find an appropriate implementation platform for the PDW role, that would accommodate our extended environment model. We have chosen the *ConceptBase* deductive object manager that is mainly intended for metadata management and conceptual modelling [JJNS98]. ConceptBase builds on the O-Telos modelling language that allows the representation of different levels of metamodelling and empowers the formulation of domain specific constraints and deductive rules. The mathematical formalization of the generic environment metamodel presented in Sect. 3 fits nicely into the ConceptBase framework and can be directly translated into an O-Telos model. For example, the following class represents the information mapping class with its corresponding constraints:

```
InformationMapping with
 attribute
  information: Information;
  product: Product
 constraint
  c2: $ forall p/Product
          (exists i/Information im/InformationMapping
          (im product p) and (im information i)) $
  c4: $ forall p/Product forall a/Activity
        forall c/Context forall s/Semantics
        FunctionalMapping(a,c,s) and InformationMapping(a,i) ==>
          (exists sit/Situation
          (c relatedTo sit) and (sit basedOn p)) $
end.
```

Until now, the enactment mechanism was solely used for the interpretation of NATURE-based process definitions. Since from now on our integrated environment metamodel pro-

vided a multi-perspective view on the design process, the enactment mechanism has been redesigned and reimplemented in order to host two synergetically working components, a *workflow engine* and a *process engine*. The role of the workflow engine is the interpretation of the workflow model and the correct delegation of activities, resources and information items to the human actors. The role of the process engine is, as previously, the interpretation of the method fragments. When demanded, the two components can interact one with the other exchanging messages in order to exploit the defined concept mappings.

4.2 Case Study

The extended PRIME-based integration platform has been validated in the frame of the *SFB 476 IMPROVE* research project [MN04] in cooperation with other German research institutes and industrial partners. Inside IMPROVE, we investigate the computer-based support of chemical engineering design. The addressed reference scenario focuses on the early phases of the polyamid6 (nylon) production, the so-called *conceptual design* and the *basic engineering*.

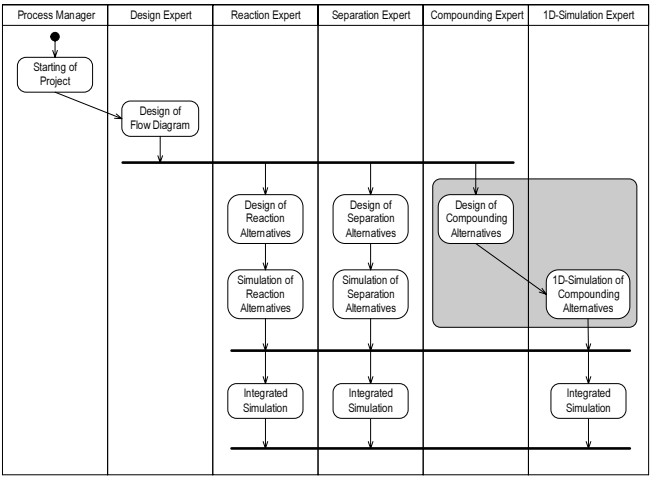

Figure 4: Simplified overview of the IMPROVE case study

The polyamid6 design process comprises three prominent steps: *reaction*, *separation* and *compounding*. In Fig. 4, a simplified overview of the IMPROVE workflow is represented using a UML activity diagram. In this diagram, we can identify the prominent roles of the experts participating in the design, as well as the control flow of their assigned activities. For reasons of clarity, information flow details are omitted.

Initially, the process manager starts the project and the design expert draws the initial flowsheet. This flowsheet is then sent to three different roles, the reaction, separation and compounding experts that investigate in parallel the different design alternatives for

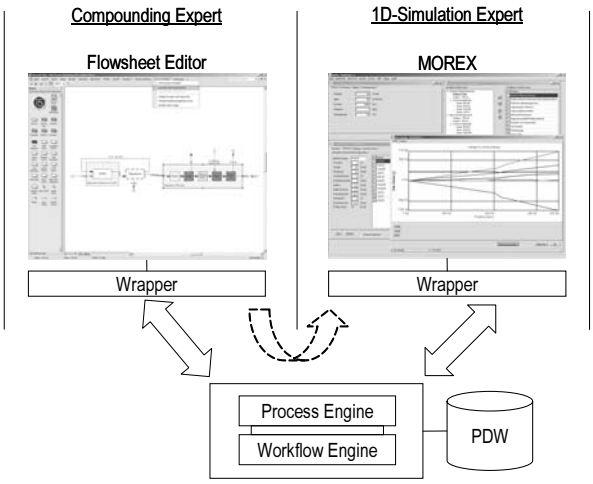

Figure 5: The IMPROVE integrated design environment

the reaction, separation and compounding steps respectively. The reaction and separation experts proceed with the simulation of the selected design alternative using special simulators, whereas the simulation of the compounding configuration is delegated to the 1D-simulation expert. At the end, the different simulation results are compared through a synchronous simulation and the original design models are corrected or improved.

Our integrated modelling framework has been tested on the basis of this case study. As indicated by various researchers and the current practice, the flowsheets play a prominent role in early chemical engineering design processes. Based on this assumption, we have created a flowsheet-centered architecture based on a fully process-integrated hierarchical flowsheet editor we have developed [BMWJ01]. This flowsheet editor plays the role of the main communication medium among all other tools used by the human actors. The basic interaction patterns while working on flowsheets have been encapsulated in method fragments.

In the following, we demonstrate our approach on the scenario fragment behind the shadowed part of the workflow in Fig. 4. We suppose that the design expert has finished designing the basic flowsheet and the compounding expert is waiting for his activity assignment. An overview of the relevant parts of the IMPROVE environment is shown in Fig. 5.

The workflow engine continually interprets the workflow model and delegates activities and their respective resources to human actors. At some point of time, the activity *'Design of Compounding Alternatives'* has to be delegated to the compounding expert. By the time that the compounding expert accepts his assignment, the workflow engine starts enacting the activity. This activity has input information the *flowsheet* that has been produced by the design expert and the *flowsheet editor* tool as engaged agent. Also, a functional mapping with *STARTING_CONTEXT* semantics exists between the activity and the context

EC_FBW_LoadFlowsheet. This executable context corresponds to an action of the flowsheet editor that opens the tool, loads the flowsheet information and displays it to the user. Thus, the control shifts to the process engine that starts executing the context. It instantly communicates with the flowsheet editor wrapper and loads the flowsheet information inside it through process automation.

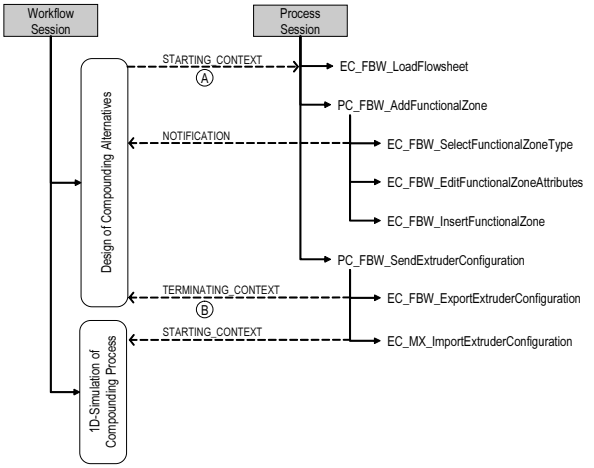

Figure 6: Workflow and process trace chunks of the described scenario

After the initialization of the activity, the compounding expert can start designing the compounding alternatives inside his flowsheet editor. During the design, whenever a situation is detected where an eligible method fragment can be offered to him, the process engine becomes active and provides method guidance. For example, for the situation that the expert wants to add a new functional zone to the extruder configuration of his flowsheet, the method fragment *PC_FBW_AddFunctionalZone* can be provided. The expert is guided step by step through this composite action according to the interpretation of the constituents of this plan context. The process step where the expert has to select the functional zone type is reflected by the *EC_FBW_SelectFunctionalZoneType* context. This type information is of special interest to the process manager since it determines to a large extent the mechanical and thermal properties of the end product and might have several side effects on other process steps. Therefore, a functional mapping with *NOTIFICATION* semantics exists between this context and activity. The process manager can get directly notified and, through an information mapping relationship, is able to get the detailed product description of the respective information item.

When the compounding expert judges that his design outcome is ready to be transmitted to the 1D-simulation expert, he notifies the process engine of his intention sending a request through the flowsheet editor wrapper for the enactment of the plan context *PC_FBW_SendExtruderConfiguration*. From a process perspective, this plan context models a tool-spanning method fragment that transfers the compounding configuration to the 1D-simulation expert workplace. This information is then imported by the *MOREX* com-

pounding extruder simulation tool [SH02] and the simulation starts. From a workflow perspective, this plan context models an activity-spanning method fragment that implements an inter-activity control transition and transfer of information. The first constituent of this plan context is the *EC_FBW_ExportExtruderConfiguration* executable context that exports the extruder configuration from the flowsheet tool and is mapped as a *TERMINATING_CONTEXT* to the first (and current) activity. Its second constituent imports the extruder configuration inside MOREX and is mapped as a *STARTING_CONTEXT* to the second activity *'1D-Simulation of Compounding Process'*.

Figure 6 shows the trace chunks of the session of the described scenario from both a workflow and process perspective. It emphasizes the temporal order of activities and contexts, as well as their interrelations through concept mappings. For reasons of clarity, only the described concept mappings are shown. The reader can also identify instances of the two interrelation patterns A and B shown on Fig. 1.

5 Conclusions and Future Work

The presented integrated modelling framework makes considerable steps towards bridging the gap between the administration and process levels of support to engineering design processes. We argued that our approach should keep a multi-perspective viewpoint on the design process without reinventing the wheel or making any kind of compromise for either of the two sides. To this end, we started with two well-understood and established metamodels for the description of business workflows and engineering methodologies respectively. As a next step, we elaborated a set of 'semantic bridges' among the two metamodels. These bridges have been materialized through concept mappings interrelating the corresponding functional, information and resource elements of the two sides. The integration of the workflow and process definitions with their concept mappings has formed the environment model. Inside the IMPROVE project, we have created a prototypical PRIME-based environment for its interpretation and have applied our approach on a chemical engineering case study with industrial relevance.

The proposed framework is still under development. On the formal side, a stronger formalization and refinement of the process mapping semantics is an issue of ongoing work, since it is one of the prominent communication means between the managers and the process performers, with a strong impact. Moreover, we feel that an evaluation of our approach in a real industrial context will greatly help us to improve our concepts according to real-world requirements and needs. In the near future, we are planning, in cooperation with industrial partners, to apply our approach to the scenario of a project dealing with continuous production processes for rubber profiles.

Acknowledgments This work was carried out in the Collaborative Research Center SFB 476 IMPROVE which is funded by the Deutsche Forschungsgemeinschaft (DFG). We are indebted to Christoph Quix for his valuable contribution to our work. Last but not least, thanks are due to Michael Comanns and Muhammad Tauseef Ikram for their implementation efforts.

References

[AB01] Wil van der Aalst and Paul Berens. Beyond Workflow Management: Product-Driven Case Handling. In *Proceedings of International ACM SIGGROUP Conference on Supporting Group Work*, 2001.

[AH02] Wil van der Aalst and Kees van Hee. *Workflow Management*. The MIT Press, 2002.

[BMWJ01] Birgit Bayer, Wolfgang Marquardt, Klaus Weidenhaupt, and Matthias Jarke. A Flowsheet Centered Architecture for Conceptual Design. In *Proceedings of the European Symposium on Computer Aided Process Engineering*, 2001.

[HJS+04] Markus Heller, Dirk Jäger, Marcus Schlüter, Ralph Schneider, and Bernhard Westfechtel. A management system for dynamic and interorganizational design processes in chemical engineering. *Computers and Chemical Engineering*, 29:93–111, 2004.

[JD94] Margarida F. Jacome and Stephen W. Director. A Formal Basis for Design Process Planning and Management. In *Proceedings of the International Conference on Computer-aided Design*, 1994.

[JJNS98] Manfred A. Jeusfeld, Matthias Jarke, Hans W. Nissen, and Martin Staudt. *Handbook of Information Systems*, chapter ConceptBase – Managing Conceptual Models about Information Systems, pages 265–285. Springer, 1998.

[JLK00] Matthias Jarke, Thomas List, and Jörg Köller. The Challenge of Process Data Warehousing. In *Proceedings of the 26th International Conference on Very Large Data Bases*, pages 473–483, 2000.

[JRSD99] Matthias Jarke, Colette Rolland, Alistair Sutcliffe, and Ralf Dömges, editors. *The NATURE of Requirements Engineering*. Shaker Verlag, 1999.

[MN04] Wolfgang Marquardt and Manfred Nagl. Workflow and Information Centered Support of Design Processes – The IMPROVE Perspective. *Computers and Chemical Engineering*, 29:65–82, 2004.

[PWD+99] Klaus Pohl, Klaus Weidenhaupt, Ralf Dömges, Peter Haumer, Matthias Jarke, and Ralf Klamma. PRIME – Toward Process-Integrated Modeling Environments. *ACM Transactions on Software Engineering and Methodology*, 8:343–410, 1999.

[RR01] Jolita Ralyté and Colette Rolland. An Assembly Process Model for Method Engineering. In *Proceedings of the 13th International Conference on Advanced Information Systems Engineering*, 2001.

[SFB00] SFB 476 IMPROVE, editor. *Work Report*. RWTH Aachen University, 2000.

[SH02] Marcus Schlüter and Edmund Haberstroh. Design of Twin Screw Extruders with the MOREX Simulation Software. In *Proceedings of the Chemical Engineering Processes Conference*, 2002.

[SKD+97] Eswaran Subrahmanian, Suresh L. Konda, Allen Dutoit, Yoram Reich, Douglas Cunningham, Robert Patrick, Mark Thomas, and Arthur W. Westerberg. The n-dim Approach to Creative Design Support Systems. In *Proceedings of the ASME Design Engineering Technical Conference*, 1997.

[Ver96] Francois B. Vernadat. *Enterprise Modeling and Integration - Principles and Applications*. Chapman and Hall, 1996.

[WfM01] WfMC. The Workflow Management Coalition Specification (http://www.wfmc.org), 2001.