Modeling Human Aspects of Business Processes – A View-Based, Model-Driven Approach

Ta'id Holmes, Huy Tran, Uwe Zdun, and Schahram Dustdar

Distributed Systems Group, Institute of Information Systems
Vienna University of Technology, Vienna, Austria
{tholmes, htran, zdun, dustdar}@infosys.tuwien.ac.at

Abstract. Human participation in business processes needs to be addressed in process modeling. BPEL4People with WS-HumanTask covers this concern in the context of BPEL. Bound to specific workflow technology this leads to a number of problems. Firstly, maintaining and migrating processes to new or similar technologies is expensive. Secondly, the low-level, technical standards make it hard to communicate the process models to human domain experts. Model-driven approaches can help to easier cope with technology changes, and present the process models at a higher level of abstraction than offered by the technology standards. In this paper, we extend the model-driven approach with a view-based framework for business process modeling, in which models can be viewed at different abstraction levels and different concerns of a model can be viewed separately. Our approach enables developers to work with meta-models that represent a technical view on the human participation, whereas human domain experts can have an abstract view on human participation in a business process. In order to validate our work, a mapping to BPEL4People technology will be demonstrated.

1 Introduction

In a process-driven, service-oriented architecture (SOA), process activities invoke services to perform the various tasks of the process. In such processes, often humans play a central role, and hence process activities must be provided that model human tasks and use services to "invoke" human actors who play a particular role in the process. In such a process-driven SOA with human involvement, various concepts and technologies (standard and proprietary) are involved. A typical standards-based solution in the Web services realm is to use REST [1] and SOAP [2,3] for distributed service invocations, WSDL [4] for service descriptions, BPEL [5] for orchestration of services through process models, BPEL4People [6] for human involvement in BPEL processes, and WS-HumanTask [7] to describe service-oriented descriptions of human tasks.

The diversity and constant evolution of these technologies and the underlying concepts hinders the changeability, understandability, and maintainability of process-driven SOAs – and hence makes evolution of process-driven SOAs a costly and error-prone undertaking. This is because systems are realized using specific technology without abstracting or conceptualizing the solutions. Historically, however, none of the technologies mentioned has emerged without having a precursor, and often companies have a legacy investment in one or multiple legacy technologies. For instance, the mentioned

BPEL standard evolved out of WSFL [8] and XLANG [9], which themselves emerged out of other existing workflow languages, and also there are several versions of the BPEL standard. The same is true for all other mentioned technologies and standards.

BPEL4People was proposed in a white paper in June 2005 as a technology for integrating human interaction of people with BPEL processes [10]. Two years later, in July 2007, version 1.0 of BPEL4People and the related WS-HumanTask [7] standards have been published. During this long period naturally solutions for integrating humans into service-oriented infrastructures have been proposed, for instance Human-provided Services [11]. Also BPEL4People concepts based on the white paper have been realized by industry and academia (see for instance [12]). They specified syntax and defined semantics for addressing the concepts as introduced in the white paper. These implementations now must be adapted to comply with the standards. Looking ahead it is clear that with new versions to come the current standards will become obsolete again in only a matter of time.

In order to reduce migration and maintenance costs, adaptation to such – rather typical – technology and (technology standards) life-cycles should be easy to perform. While concepts of a system may not change, new technology may introduce new syntax elements and may modify semantics. Therefore it is desirable to have conceptual representations within a system that have only the necessary dependencies on foundational technology.

The case of modeling human aspects in SOAs shows this aspect very clearly. There is a second, related problem that is also quite apparent in this case: the representation of the conceptual knowledge embedded in the technologies to end-users. While developers may be interested in the low-level, technical standards mentioned above, such technology-dependent views on a process-driven system are hard to communicate to domain experts. Instead a simplified view at a higher level of abstraction is needed.

Model-driven development (MDD) [13] addresses these needs by defining meta-models that express domain concepts. Platform-independent models that conform to these meta-models can subsequently be transformed to platform-specific code. In order to switch to similar but different technology all that needs to be done is to define a different transformation that transforms the conceptual models accordingly. Within this work we apply model-driven development to business process design. Particularly we focus on modeling business processes that involve humans.

However, these meta-models are typically still too detailed and technical to be presented to domain experts. To solve this issue, we propose to present a customized representation of our conceptual models to stakeholders. In particular, we propose to extend the model-driven approach using a view-based modeling framework [14] that realizes separation of concerns [15] for managing development complexity. For instance, while a business expert may be interested in the control-flow of a process, a human resource officer rather deals with assigning people to tasks. As we will introduce the framework, we will also present a concept of realizing such views as an extension of the model-driven approach.

But not only does the framework realize a separation of concern, it also enables developers to place meta-models at defined levels of abstraction. Complexity can be added gradually and while top level views give a simplified overview, refined models

may supply technical details on the given concern. This is needed for instance to model that technical developers need a technology-related view, while the two aforementioned stakeholders rather need less detailed views, abstracting from the technical details.

Within this work we will particularly refine an abstract human meta-model towards a technology-specific one for which a model-to-code transformation will be defined in order to obtain a BPEL4People process. That is, we will extend the view-based modeling framework by new views, dedicated to the concern of how people interact with business processes. While general Human-Process dependencies will be reflected by an abstract, conceptual view, details specific to BPEL4People and WS-HumanTask will be subjoined in an refined BPEL4People view. Finally, we will relate these meta-models to appropriate syntax elements.

This paper is structured as follows: After having defined the problem, Section 2 will introduce and extend a view-based modeling framework [14] with a meta-model for human-process participation and association. This view will then be extended with concepts from BPEL4People. A concrete binding to BPEL4People is presented in Section 3 by mapping these meta-models to the BPEL4People syntax. In Section 4 we will discuss related work and we will conclude with Section 5 by referring to further work.

2 A View-Based Approach

In order to describe human aspects of business processes, we have defined dedicated views for our view-based modeling framework (VbMF, see [14] for more details), which is introduced in this section. The VbMF framework consists of modeling elements such as a meta-meta-model, meta-models, and views. The meta-models are defined using the Eclipse Modeling Framework (EMF) [16]. These EMF models are utilized in openArchitectureWare (oAW) [17], a modular generator for MDD and model-driven architectures (MDA) [18], for integrating the view-based models and model-to-code-transformations.

A *view* is a representation of a process from the perspective of related concerns. In VbMF, new architectural views can be *designed*, existing meta-models can be *extended* by adding new features, views can be *integrated* in order to produce a richer view of a business process and using *transformations*, platform specific code can be generated.

Figure 2 demonstrates the core meta-model of the framework that is derived from the Ecore meta-meta-model [16]. It defines basic elements for the framework and introduces an element view as an representation of a business process concern. Other meta-models make use of this core meta-model, so that relationships between different meta-models can be defined and maintained. In particular model integration is realized by name-based matching amongst name attributes of elements within the core meta-model. Other matching algorithms, such as semantic matching can be plugged into VbMF, if needed.

As mentioned, the framework consists of multiple views, separating the various concerns that in their entirety describe an overall business process and the service integration. The main views defined by VbMF are: The *control-flow view* describes the control-flow of the process. The *collaboration view* specifies the orchestration of external activities. The *information view* contains details on data types and messages. The

transaction view deals with long-running transactions, as they can be found in process-based systems. Figure 1 gives an overview of the VbMF framework and its main views.

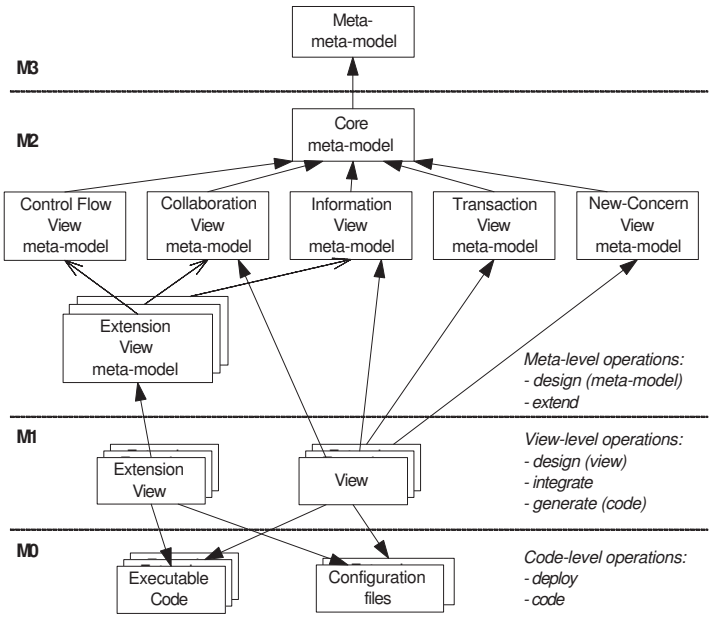


Fig. 1. Overview of the view-based modeling framework

VbMF is generally designed to be extensible in various ways. Firstly, VbMF can be extended with views for other concerns. We will illustrate this extensibility in the next section using a human view. Secondly, VbMF can be extended with views situated at different abstraction levels. For instance, for the above mentioned views, more technical views have been defined for BPEL/WSDL-based control-flow, collaboration, information, and transactions. Below we will illustrate this kind of extension using the BPEL4People view, which extends the human view.

2.1 Extending the Framework with Human Views

Figure 3(a) gives an overview of our extensions to the VbMF framework introduced in this paper: Two views dedicated to describe human aspects of business processes have been defined and a transformation for generating BPEL4People has been implemented (described in the next section).

We have extended the framework with a human view as shown in Figure 3(b) for describing human aspects of a process. Via name-based matching, it introduces the relation of processes and activities to human roles. Roles are abstracting concrete users that may *play* certain roles. The human view thus establishes a role-based abstraction.

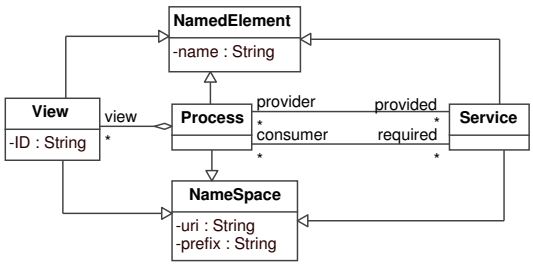
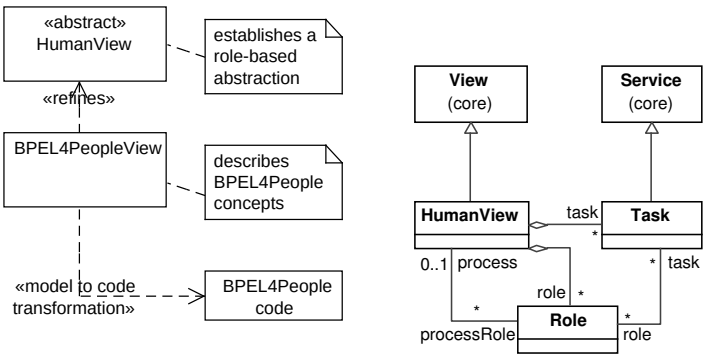


Fig. 2. The *core* meta-model

This role-based abstraction can be used for role-based access control (RBAC). RBAC, in general, is administered through roles and role hierarchies that mirror an enterprise's job positions and organizational structure. Users are assigned membership into roles consistent with a user's duties, competency, and responsibility [19].



(a) Overview of the human views

(b) The *human* meta-model

Fig. 3. Introducing the human views

We can specify an activity as defined within a control-flow view to be a human task that is bound to for instance an owner, the person who performs the task. Likewise process stakeholders can be specified for the process by associating them with the human view that together with other views describes the overall process.

In the meta-models, described so far, there are no restrictions on how processes and tasks may be bound to roles. Particularly this view does not define or propose roles for processes or tasks. BPEL4People as well as WS-HumanTask on the other hand define generic human roles (see also Section 3.2). The specifications describe certain use case

scenarios for different roles and therefore dictate access control for the human roles. By working with these roles, BPEL4People technology can help to reduce the complexity and cost of authorization management [19].

2.2 Refining the Meta-Model for BPEL4People

The more specific BPEL4People view extends the human view with a technology-specific perspective on the human aspect. BPEL4People glues BPEL activities and human tasks by introducing `peopleActivity` as a new BPEL `extensionActivity`. The human tasks that may be encapsulated or referenced by the people activities are described in the WS-HumanTask specification. In order to hide complexity, these technology-related aspects are not shown in the generic human view, but only in the specific BPEL4People view.

A meta-model for the BPEL4People view is shown in Figure 4. This view inherits from the human view, binds roles to people links (that themselves are bound to concrete people queries) and integrates other *concepts* from BPEL4People. A task for example may hold a description, may be specified to be skipable and can specify scheduled actions that occur when time constraints are violated. Also propagation of ad hoc attachments from and to the process can be defined for a task.

Although for instance descriptions of tasks may already have been defined within a technology neutral meta-model, the optional description – which might be supplied for an arbitrary number of languages – is a specific requirement of BPEL4People. Therefore and in order to avoid polluting the abstract meta-model, the task description is specified – together with other technology-specific concepts – in the BPEL4People view.

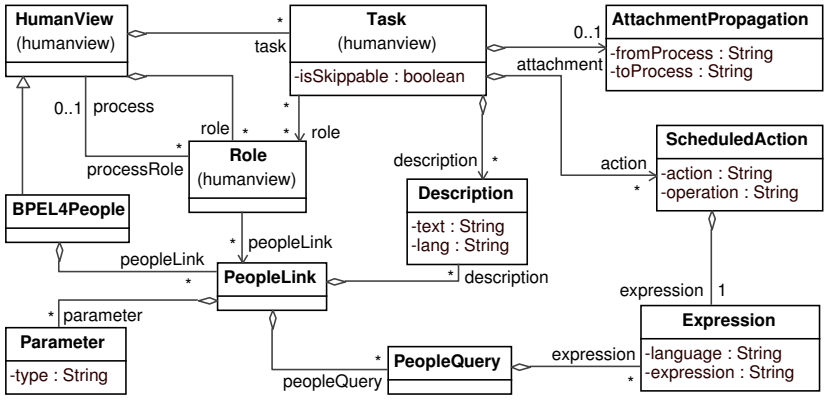


Fig. 4. Meta-model for the *BPEL4People* view

Roles need to be bound to a set of instances of data that identify persons. Therefore they make use of a *people link* that – when resolved by *people resolution* – results in

such a set. People links contain *people queries*, can have *descriptions* and may specify *parameters*. Expressed in a certain language, people queries will be executed during people resolution. By decoupling roles from people links, *reuse* of the latter can take place. Furthermore – and as mentioned above – a role-based abstraction is established from the people link with its more technical aggregations.

3 Application to BPEL4People

To demonstrate the application of the presented models to BPEL4People and platform-specific code, we will elaborate on the mapping to BPEL4People and WS-HumanTask that define concrete syntaxes for human process interactions and relations. We will conclude this section by demonstrating a use case example. For explaining how the different modeled concepts relate to the standards we will already utilize code of this use case for illustrating purposes within the following paragraphs. Although we will cover important concepts as captured within the introduced human view meta-models, we do not intend to be complete in regard to the specifications [6,7] within this paper.

3.1 Tasks

A human *task* is a process element that is part of the control-flow. It is realized by a *people activity* in BPEL4People as shown in Listing 1.1 that defines or references a *task definition*. Section 4.7 of [6] permits the specification of *scheduled actions* like *defer activation* and *expiration* that can contain *for* or *until* expressions. Moreover the propagation of attachments from and to processes can be specified for a people activity and the attribute `isSkipable` indicates whether the task associated with the activity can be skipped at runtime or not. Section 4.1.1 of [6] lists and describes the semantic of different properties.

```
<bpel:extensionActivity>
   <b4p:peopleActivity name="Acknowledgement"
                  inputVariable="ack_input"
                  outputVariable="ack_output"
                  isSkipable="true">
      <b4p:localTask reference="tns:AcknowledgementTask"/>
      <b4p:scheduledActions>
         <b4p:deferActivation>
            <b4p:for>
               ...
            </b4p:for>
         </b4p:deferActivation>
      </b4p:scheduledActions>
      <b4p:attachmentPropagation fromProcess="all" toProcess="all"/>
   </b4p:peopleActivity>
</bpel:extensionActivity>
```

Listing 1.1. BPEL4People syntax for a human *process element*

In our view-based modeling framework human tasks are modeled as *simple activities* in the control-flow view. Within the human view such activities can be annotated via *name-based matching* to contain human aspects by specifying the name as denoted in the simple activity of the control-flow view.

We translate a task that was modeled in the human view to an appropriate task definition and reference it within a people activity. In contrast to encapsulated tasks, local reuse of task definitions within a process can thus take place which might result in an optimized behavior of the runtime engine that hosts the BPEL4People process.

3.2 Roles

Generic human roles as *process stakeholders*, *process initiators* and *business administrators* for processes have been defined in Section 3.1 of [6]. The human view metamodel contains an association between the human view of the process and roles that may define these generic human roles. Analogically this can be done for the task-role association. Section 3.1 of [7] defines appropriate roles for tasks such as *task initiator*, *task stakeholders*, *potential owners*, *actual owner*, *excluded owners*, *business administrators* and *notification recipients*.

Process Roles When mapping to BPEL4People we define the generic human roles for the process within a `peopleAssignments` container by referencing to corresponding `logicalPeopleGroups` as shown in Listing 1.2.

```
<b4p:peopleAssignments>
   <b4p:businessAdministrators>
      <htd:from logicalPeopleGroup="businessAdministratorsLPG" />
   </b4p:businessAdministrators>
   <b4p:processInitiator>
      <htd:from logicalPeopleGroup="processInitiatorLPG" />
   </b4p:processInitiator>
   <b4p:processStakeholder>
      <htd:from logicalPeopleGroup="processStackeholderLPG" />
   </b4p:processStakeholder>
</b4p:peopleAssignments>
```

Listing 1.2. BPEL4People syntax for associating human *roles* to a *process*

Task Roles People assignment for generic human task roles is performed within a task definition. As with process roles we reference corresponding `logicalPeopleGroups` as shown in Listing 1.3.

```
<htd:task name="AcknowledgementTask">
   <htd:interface operation="ack"
               portType="acknowledgementservice:acknowledgePT"/>
   <htd:peopleAssignments>
      <htd:taskInitiator>
         <htd:from logicalPeopleGroup="taskInitiatorLPG" />
      </htd:taskInitiator>
      <htd:taskStakeholders>
         <htd:from logicalPeopleGroup="taskStakeholdersLPG" />
      </htd:taskStakeholders>
      <htd:potentialOwners>
         <htd:from logicalPeopleGroup="acknowledgementPotentialOwnersLPG" />
      </htd:potentialOwners>
      <htd:notificationRecipients>
         <htd:from logicalPeopleGroup="notificationRecipientsLPG" />
      </htd:notificationRecipients>
      <htd:excludedOwners>
         <htd:from logicalPeopleGroup="peopleNotAllowed2AcknowledgeLPG" />
```

```
      </htd:excludedOwners>
   </htd:peopleAssignments>
</htd:task>
```

Listing 1.3. WS-HumanTask syntax for associating human *roles* to a *task*

The binding of roles to people links within the BPEL4People view is not restricted to a one-to-one mapping. Instead reuse of people links can take place as no composition is specified for the relation.

3.3 People Links

People links are transformed to `logicalPeopleGroup` elements as shown in Listing 1.4. Descriptions are translated to `documentation` elements that – together with optional parameters, that data can be used for people query evaluation – are placed as sub-elements within the corresponding elements.

```
<htd:logicalPeopleGroups>
   <htd:logicalPeopleGroup name="peopleNotAllowed2AcknowledgeLPG">
      <htd:documentation xml:lang="en">
         These people are not allowed to
         acknowledge the order.
      </htd:documentation>
      <htd:parameter name="name" type="xs:string"/>
      ...
   </htd:logicalPeopleGroup>
   <htd:logicalPeopleGroup name="processStackeholderLPG" >
      ...
   </htd:logicalPeopleGroup>
   ...
</htd:logicalPeopleGroups>
```

Listing 1.4. WS-HumanTask syntax for associating *people links* to *people queries*

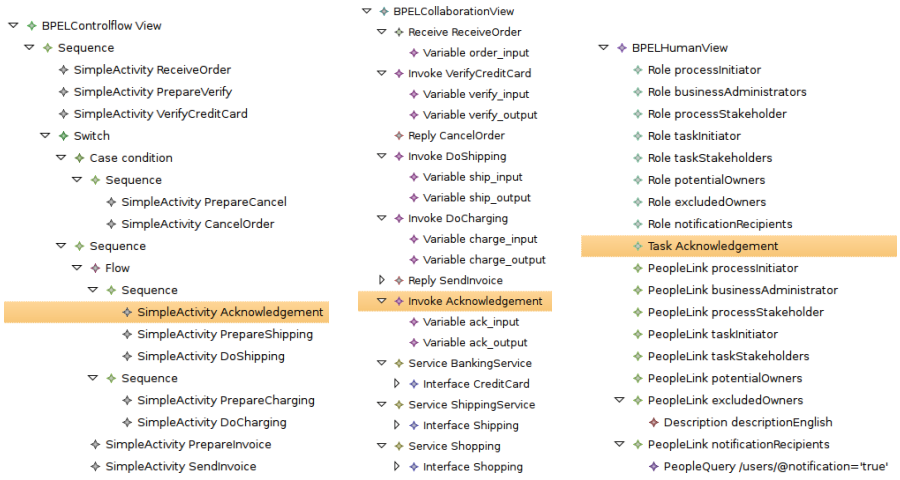
We have stated a possible mapping from the presented conceptual meta-models to BPEL4People code. We have seen that elements like people links, together with their aggregated descriptions and parameters, have concrete relations to designated BPEL4People and WS-HumanTask syntax elements.

3.4 A Use Case Scenario

In this section, we have explained the main concepts of human-process relation in regard to the BPEL4People and WS-HumanTask specifications. For demonstrating purposes we already have shown code examples that derive from a use case that we now want to describe in more detail where we will illustrate code generation using model-to-code templates. These templates are written in Xpand, which is a language for model-to-code transformation within the oAW's expression framework.

We illustrated a use case scenario for a shopping ordering process. At a certain stage of the process an acknowledgement from an authorized person is required. Therefore the process makes use of a human task as a special kind of a process activity.

We have modeled this scenario using the VbMF and Figure 5 shows how the human task *Acknowledgment* has been designed in various views. While Figure 5(a) shows the control-flow of the process with a corresponding *simple activity*, Figure 5(c) defines the task in the human view. Invocation of activities as well as human tasks with input and output variables is specified in the collaboration view as shown in Figure 5(b). Besides the definition of the human task, the appropriate generic human roles, that are associated with the process and the tasks, are also defined in the BPEL4People view.



(a) control-flow view (b) collaboration view (c) human view

Fig. 5. A human task within the different views

People links are transformed into a set of `logicalPeopleGroups` as Figure 6 demonstrates. In order to obtain valid BPEL4People code the element names for the generic human roles as defined in [6] and [7] have to be used as the name of the people links when modeling. Figure 7 shows the generation of task definitions with task roles that reference corresponding people links.

The invocation of process activities may be performed by a Web service invocation using the BPEL `invoke` activity or in case of a local task the BPEL4People `peopleActivity`. Figure 8 shows the transformation template that generates an appropriate `extensionActivity` for the `peopleActivity` in case a task is found in the human view for the activity as specified in the control-flow by name-based matching. The `getTaskByName` function accomplishes this matching algorithm. This referenced function has been implemented in Xtend, another language of the oAW's expression framework, that provides the possibility to define libraries of independent operations and no-invasive meta-model extensions [17]. If no task has locally been specified for the process, an external activity will be invoked.

```
«DEFINE LogicalPeopleGroups(bpelinformation::BPELInformationView iv,
           bpelcollaboration::BPELCollaborationView cv,
           bpel4people::BPEL4People_HumanView hv)
           FOR controlflow::ControlFlowView-»
    «IF (getPeopleLinks(hv).size > 0)-»
        <htd:logicalPeopleGroups>
        «FOREACH getPeopleLinks(hv) AS link-»
            <htd:logicalPeopleGroup name="«link.name»LPG">
            «FOREACH link.description AS description-»
                <htd:documentation«IF (description.lang != null)-»
                    xml:lang="«description.lang»"«ENDIF»>
                    «description.text»
                </htd:documentation>
            «ENDFOREACH-»
            «FOREACH link.parameter AS parameter-»
                <htd:parameter name="«parameter.name»"
                    type="«parameter.type»" />
            «ENDFOREACH-»
            </htd:logicalPeopleGroup>
        «ENDFOREACH-»
        </htd:logicalPeopleGroups>
    «ENDIF-»
«ENDDEFINE»
```

Fig. 6. Model-to-code template for *people links*

The control-flow view does not distinguish between the invocation of an external activity or the delegation to a local human task. Adopting the notion of transparency [20] we can therefore say that invocation is made transparent for modeled activities of the control-flow from a design point of view. As a consequence, activities can simply be exchanged between human tasks or Web service invocations without the need to alter the control-flow in turn.

4 Related Work

Related Work on Model-Driven Design Our work is based on the model-driven development (MDD) paradigm [13,21] and extends the MDD paradigm with the notion of architectural views – expressed in terms of meta-models. Two kinds of extensibility are supported: "horizontal" extension with views for additional concerns and "vertical" extension with views at different abstraction levels. Using our view-based approach, we are able to separate the human view – in focus of this paper – from other concerns, and separate the views for different stakeholders on the human view: high-level views for domain experts and low-level views for technical experts.

List and Korherr [22] compare and evaluate seven conceptual business process modeling languages and propose a generic meta-model that is categorized according to the framework introduced in [23] to four perspectives: organisational, functional, behavioural, and informational. Additionally a perspective for the business context addresses context information like process goals. While it is interesting to capture different conceptual business process modeling languages into a common normalized meta-model we do not need to support these in order to obtain code for process runtime

```
«DEFINE Tasks(bpelinformation::BPELInformationView iv,
            bpelcollaboration::BPELCollaborationView cv,
            bpel4people::BPEL4People_HumanView hv)
            FOR controlflow::ControlFlowView-»
    «IF (getTasks(hv).size > 0)-»
        <htd:tasks>
        «FOREACH getTasks(hv) AS task-»
            «LET getInvokeByName(task.name, cv) AS invoke-»
            <htd:task name="«task.name»Task">
                <htd:interface
                    operation="«invoke.operation.name»"
                    portType="«invoke.associatedInterface.service.name.toLowerCase(
                        )»:«invoke.associatedInterface.name»PT"
                />
                <htd:peopleAssignments>
                «FOREACH getTaskRoles(task) AS role-»
                    <htd:«role.name»>
                        <htd:from logicalPeopleGroup="«role.peopleLink.name»LPG"/>
                    </htd:«role.name»>
                «ENDFOREACH-»
                </htd:peopleAssignments>
                <htd:presentationElements>
                «FOREACH task.documentation AS description-»
                    <htd:description«IF (description.lang != null)-»
                    xml:lang="«description.lang»"«ENDIF» contentType="text/plain">
                        «description.text»
                    </htd:description>
                «ENDFOREACH-»
                </htd:presentationElements>
            </htd:task>
            «ENDLET»
        «ENDFOREACH-»
        </htd:tasks>
    «ENDIF-»
«ENDDEFINE»
```

Fig. 7. Model-to-code template for *tasks*

execution. As a matter of fact, a mapping from high level business process modeling languages such as the Business Process Modeling Notation (BPMN) to a certain technology such as BPEL often is missing [24,25,26]. Instead of a comprehensive meta-model we, moreover, want to work with small conceptual models as proposed in [27] that rather represent the least common denominator but can be extended and bound and/or translated to low-level models that support process execution.

A Model Driven Visualization framework is introduced by Bull [28] that provides a mechanism to rapidly prototype new visualizations from meta-models. So called *snap points* define views for domain experts. Considering that the VbMF defines the meta-models for business processes, a direct application of the presented work - that also is based on EMF - would allow customized business process representation to various stakeholders.

Related Work on View-Based Modeling There are only a few view-based approaches to business process modeling. To the best of our knowledge, none of them integrates a human view. The approach by Mendling et al. [29] is inspired by the idea of schema

```
«DEFINE SimpleActivity(bpelinformation::BPELInformationView iv,
    bpelcollaboration::BPELCollaborationView cv,
    bpel4people::BPEL4People_HumanView hv)
    FOR bpelcollaboration::Invoke-»
    «LET getTaskByName(name, hv) AS task-»
        «IF (task != null)-»
            <bpel:extensionActivity>
                <b4p:peopleActivity name="«name»"
            «IF (in != null)-»
                    inputVariable="«in.name»"
            «ENDIF-»
            «IF (out != null)-»
                    outputVariable="«out.name»"
            «ENDIF-»
            «IF (task.isSkipable != null)-»
                    isSkipable="«task.isSkipable»"
            «ENDIF-»
                >
                    <b4p:localTask reference="tns:«name»Task"/>
                </b4p:peopleActivity>
            </bpel:extensionActivity>
        «ELSE»
            <bpel:invoke name="«name»"
            «IF (in != null)-»
                inputVariable="«in.name»"
            «ENDIF-»
            «IF (out != null)-»
                outputVariable="«out.name»"
            «ENDIF-»
            partnerLink="«partnerLink.name»"
            «LET associatedInterface.service.name.toLowerCase() AS prefix-»
                portType="«prefix»:«associatedInterface.name»"
            «ENDLET-»
            operation="«operation.name»"/>
        «ENDIF-»
    «ENDLET-»
«ENDDEFINE»
```

Fig. 8. Model-to-code template for activity *invocation*

integration in database design. Process models based on Event-driven Process Chains (EPCs) [30] are investigated, and the pre-defined semantic relationships between model elements such as *equivalent*, *sequence*, and merge operations are performed to integrate two distinct views. In contrast our approach introduces a common core meta-model and well-defined extension points, and utilizes the model-driven paradigm for view integration. That is, our approach is both more flexible and provides more well defined extension points for view integration and extension.

The Amfibia [31,32] approach focuses on formalizing different aspects of business process modeling, and/or develops an open framework to integrate various modeling formalisms through the *interface* concept. Akin to our approach, Amfibia has the main idea of providing a modeling framework that does not depend on a particular existing formalism or methodology. The major contribution in Amfibia is to exploit dynamic interaction of those aspects. Like our approach, Amfibia's framework also has a core model with a small number of important elements, which are referred to, or refined in other models. The distinct point to our framework is that in Amfibia the interaction

of different 'aspects' is only performed by event synchronization at run-time when the workflow management system executes the process. Using extension and integration mechanisms in our framework, the integrity and consistency between models can be verified earlier at the model level.

Related Work on Role-Based Abstraction Working with human labor, the use and administration of roles in regard to business processes is another topic that relates to our work as by modeling the human aspects of processes, relations between processes and tasks to authorized roles are defined.

Johnson and Henderson [33] propose data authorization and access control mechanism for Workflow Management Systems (WfMS) [34]. Similar to our human view the presented comprehensive access control model also establishes - besides defining other relations - a role based abstraction. Predicate-based access control is applied for implementation.

A standard for a functional RBAC model has been proposed by Ferraiolo et al. [19]. While the human view within the VbMF establishes role based abstraction, the actual access control for different roles is defined implicitly and applied by the technology. For instance, it is possible to specify the role of a business administrator in the human view and assign actual users to this role via people queries in the BPEL4People view. The semantics concerning the access controls of this role however are defined in the BPEL4People specification and will only be interpreted by the executing engine. Therefore, there is no need to model access controls within the VbMF as assignment of people to specific roles suffices in order to obtain a business process with predefined role based access controls for human participants.

5 Summary

We have presented meta-models for expressing human aspects of business processes within a view-based modeling framework that support the specification of processes and tasks containing human aspects. Modeling the human aspects in a process-driven SOA is challenging because technology for human aspects (such as BPEL4People and WS-HumanTask) is constantly evolving, the technology is dependent on many other technologies which also evolve (such as BPEL and Web service technology used in our work), and, finally, different stakeholders, such as domain experts in the field of human resource management, as well as software architects and developers must work with the models – and require different views. Our approach resolves this challenging case by providing a view-based modeling extension to the model-driven paradigm and by providing models that cover the concern of human-process relation. The presented views are split into an abstract, platform independent meta-model as well as an refined one. While the latter can be used for model-to-code transformation, the conceptual model is suitable for being presented to human domain experts. By exchanging the transformations and/or adapting the low-level models, the adaptation of a refined model to a new version of a standard or to another technology can easily be performed. Via model-to-code transformation we have demonstrated a possible mapping to BPEL4People as a specific technology addressing the mentioned concern. While in this work, we have

focused on the case of human aspects in process models, the same view-based approach can be generalized and be applied to other cases with similar requirements as well.

In addition to applying our approach to other cases, we plan for the following further work: Specifying the relationship between models by formalizing *integration points* would permit automatic and generic model integration. Therefore we plan to extend the framework with a domain specific language tailoring the merging of views. In order to support more features of BPEL4People like *escalation*, additional refinements to the VbMF may be defined to the presented basic views. Besides the design of business processes, also the monitoring needs to be addressed by conceptual models. Therefore we plan to provide execution views for capturing business processes runtime states.

References

1. Fielding, R.T.: Architectural styles and the design of network-based software architectures. PhD thesis (2000) Chair-Richard N. Taylor.
2. W3C Recommendation: SOAP Version 1.2 Part 1: Messaging Framework. (April 2007)
3. W3C Recommendation: SOAP Version 1.2 Part 2: Adjuncts. (April 2007)
4. W3C Note: Web Services Description Language (WSDL) Version 1.1
5. Curbera, F., Khalaf, R., Mukhi, N., Tai, S., Weerawarana, S.: The next step in web services. Commun. ACM **46**(10) (2003) 29–34
6. Agrawal, A., Amend, M., Das, M., Ford, M., Keller, C., Kloppmann, M., König, D., Leymann, F., Müller, R., Pfau, G., Plösser, K., Rangaswamy, R., Rickayzen, A., Rowley, M., Schmidt, P., Trickovic, I., Yiu, A., Zeller, M.: WS-BPEL Extension for People (BPEL4People), Version 1.0. (June 2007)
7. Agrawal, A., Amend, M., Das, M., Ford, M., Keller, C., Kloppmann, M., König, D., Leymann, F., Müller, R., Pfau, G., Plösser, K., Rangaswamy, R., Rickayzen, A., Rowley, M., Schmidt, P., Trickovic, I., Yiu, A., Zeller, M.: Web Services Human Task (WS-HumanTask), Version 1.0. (June 2007)
8. IBM Software Group: Web Services Flow Language (WSFL) version 1.0. (2001)
9. Thatte, S.: Web Services for Business Process Design. Microsoft Corporation. (2001)
10. IBM-SAP whitepaper: WS-BPEL Extension for People. (August 2005)
11. Schall, D., Truong, H.L., Dustdar, S.: On Unifying Human and Software Services for Ad-hoc and Process-centric Collaboration. IEEE Internet Computing **12**(3) (2008)
12. Holmes, T., Vasko, M., Dustdar, S.: VieBOP: Extending BPEL Engines with BPEL4People. Parallel, Distributed and Network-Based Processing, 2008. PDP 2008. 16th Euromicro Conference on (13-15 Feb. 2008) 547–555
13. Völter, M., Stahl, T.: Model-Driven Software Development: Technology, Engineering, Management. Wiley (2006)
14. Tran, H., Zdun, U., Dustdar, S.: View-based and Model-driven Approach for Reducing the Development Complexity in Process-Driven SOA. In: Intl. Working Conf. on Business Process and Services Computing (BPSC'07). Volume 116 of Lecture Notes in Informatics. (sep 2007) 105–124
15. Ghezzi, C., Jazayeri, M., Mandrioli, D.: Fundamentals of Software Engineering. Prentice Hall (1991)
16. Foundation, T.E.: Eclipse Modeling Framework. `http://www.eclipse.org/modeling/emf/` (2006)
17. openArchitectureWare.org: openArchitectureWare. `http://openarchitectureware.org` (2006)

18. Open Management Group: MDA Guide Version 1.0.1. (June 2003)
19. Ferraiolo, D.F., Barkley, J.F., Kuhn, D.R.: A role-based access control model and reference implementation within a corporate intranet. ACM Trans. Inf. Syst. Secur. **2**(1) (1999) 34–64
20. Carpenter, B.: Internet transparency (2000)
21. OMG: MDA Guide Version 1.0.1. Technical report, Object Management Group (2003)
22. List, B., Korherr, B.: An evaluation of conceptual business process modelling languages. In: SAC '06: Proceedings of the 2006 ACM symposium on Applied computing, New York, NY, USA, ACM Press (2006) 1532–1539
23. Curtis, B., Kellner, M.I., Over, J.: Process modeling. Commun. ACM **35**(9) (1992) 75–90
24. White, S.A.: Using BPMN to Model a BPEL Process (February 2005)
25. Ouyang, C., Dumas, M., ter Hofstede, A.H.M., van der Aalst, W.M.P.: From BPMN Process Models to BPEL Web Services. In: ICWS '06: Proceedings of the IEEE International Conference on Web Services (ICWS'06), Washington, DC, USA, IEEE Computer Society (2006) 285–292
26. Recker, J., Mendling, J.: On the Translation between BPMN and BPEL: Conceptual Mismatch between Process Modeling Languages. `http://citeseer.ist.psu.edu/recker06translation.html`
27. Johnson, J., Henderson, A.: Conceptual models: begin by designing what to design. Interactions **9**(1) (2002) 25–32
28. Bull, R.I.: Integrating dynamic views using model driven development. In: CASCON '06: Proceedings of the 2006 conference of the Center for Advanced Studies on Collaborative research, New York, NY, USA, ACM (2006) 17
29. Mendling, J., Simon, C.: Business Process Design by View Integration. In: Business Process Management Workshops. Volume 4103 of LNCS., Springer (2006) 55–64
30. Keller, G., Nüttgens, M., Scheer, A.W.: Semantische Prozeßmodellierung auf der Grundlage "Ereignisgesteuerter Prozeßketten (EPK)". Arbeitsbericht Heft 89, Institut für Wirtschaftsinformatik Universität Saarbrücken (1992)
31. Axenath, B., Kindler, E., Rubin, V.: An open and formalism independent meta-model for business processes. In: Proceedings of the Workshop on Business Process Reference Models. (2005) 45–59
32. Kindler, E., Axenath, B., Rubin, V.: AMFIBIA: A Meta-Model for the Integration of Business Process Modelling Aspects. In: The Role of Business Processes in Service Oriented Architectures. Number 06291 in Dagstuhl Seminar Proceedings (2006)
33. Wu, S., Sheth, A.P., Miller, J.A., Luo, Z.: Authorization and access control of application data in workflow systems. J. Intell. Inf. Syst. **18**(1) (2002) 71–94
34. Georgakopoulos, D., Hornick, M., Sheth, A.: An overview of workflow management: from process modeling to workflow automation infrastructure. Distrib. Parallel Databases **3**(2) (1995) 119–153