

# An Extended Resource Information Layer for BPMN

Alexander Großkopf

Hasso-Plattner-Institute for IT Systems Engineering  
University of Potsdam, D-14480 Potsdam, Germany  
alexander.grosskopf@hpi.uni-potsdam.de

## Abstract

BPMN is an emerging standard for process modelling and has the potential to become a process specification language to capture and exchange process models between stakeholders and tools. Ongoing research and standardisation efforts target a formal behavioural semantics and metamodel. Yet it is hardly specified how humans are embedded in the processes and how the work distribution among human resources can be defined. This paper addresses these issues by identifying the required model information based on the Workflow Resource Patterns. We evaluate BPMN and the upcoming metamodel standard (BPDM) for their capabilities and propose extensions.

**Keywords:** BPMN, Business Process Modelling, Workflow Resource Patterns, BPDM

## 1 Introduction

IT support for business processes has dramatically changed the way companies are able to perform their work. Fully automateable processes such as payroll accounting are executed fast and reliable without the need for human interaction. Nevertheless, complex business processes require humans to take decisions and drive process steps.

The area of fully automated processes is well explored and WS-BPEL [1] is the established standard supported by 25 vendors with design tools and execution engines. The area of processes with human involvement is yet without a useful standard.

The Business Process Modelling Notation (BPMN), the current modelling standard of choice, could fill this gap as it is in favour to be refined from a drawing standard to a specification language. An according thread is the Business Process Definition MetaModel (BPDM) which aims to provide a notation independent representation of process models. Both standards, BPDM and BPMN, are driven by OMG <sup>1</sup> and BPDM is to become the metamodel foundation for BPMN. It can be expected that BPMN will extend its capabilities by aligning

---

<sup>1</sup>Object Management Group - [www.omg.org](http://www.omg.org)

with the new BPDM concepts. In this paper we focus on these two standards and examine their ability to capture resource information.

A resource in our context is a human within an organisation which is able to conduct a task in a process. A task is an instance of work at runtime which is created from an activity. In other words, we use the term *task* synonymous to activity instance.

**Different perspectives** are dedicated to separate concerns in process models. At a minimum process models capture *activities* (work to be done), *control flow* (dependencies between activities) and *data flow* (data dependencies & transformation). If humans participate in the process, the work distribution and related constraints are captured as *resource information*.

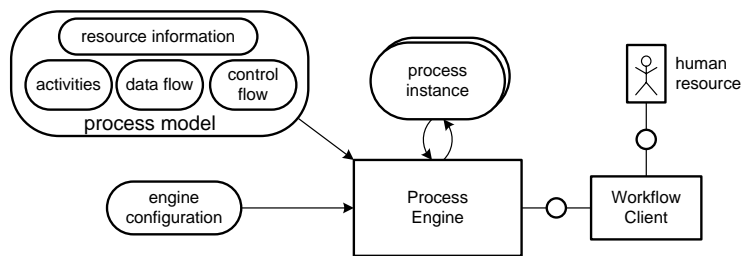


Figure 1: Workflow Environment

Using FMC block diagrams [2] Figure 1 depicts relevant parts of an environment which can execute processes driven by human resources. A *process model* with the different aspects, as described above, is the input to the process engine. Upon activation a *process instance* is created from the *process model*. The engine handles the process instance according to its *engine configuration*. Humans interact with the process instance using a *workflow client*.

All of the depicted components in Figure 1 contain information or features to handle humans participating in processes. For this paper we focus on the *resource information* in the process model.

**The remainder of the paper** is structured as follows: In Section 2 we discuss related work and then sharpen the scope of this paper by making assumptions and defining our goals in Section 3. We introduce a sample process in Section 4 to which we refer in the remainder of the paper.

A framework of required resource model information is identified in Section 5. Using that framework we assess BPMN and BPDM in Section 6 for their capabilities. We then propose extensions in Section 7. Our proposal is validated in Section 8. We conclude the paper in Section 9 by discussing the results and future work.

## 2 Related Work

In 2001 v.d.Aalst and Kumar investigated the challenges for team-enabled workflow management systems [3]. The result is a reference model on dependencies

of teams that work on a case. The focus of the work is to identify participants and their dependencies within the organisation.

In 2004 Nick Russel et.al. identified Workflow Resource Patterns [4, 5] by examining workflow implementations and standards for their capabilities in dealing with work performed by human resources. The patterns range from work distribution specification abilities to work list features [4] and summarize what can be done in that area.

In July 2005, SAP and IBM released a Joint White Paper named BPEL4People [6] which motivates extensions to the popular BPEL standard [1] in an attempt to support human involvements in BPEL processes. The paper describes scenarios and features of an environment which integrates humans into processes. The Advanced Integration Patterns ([6] p.6) describe a subset of the Workflow Resource Patterns [4] using other terminology. BPEL4People has not evolved to a standard but remained a motivation paper describing the problem domain.

Recent research on BPMN focussed on its suitability as a modelling language [7, 8]. In that context BPMN was also evaluated against the Workflow Resource Patterns [4]. But due to the missing formal grounding of BPMN, assumptions have been the basis for the language assessment. Extensions have not been discussed.

Meanwhile a new standard is on its way as discussed earlier. The Business Process Definition MetaModel (BPDM) aims to provide the capability to represent business processes independent of notation or methodology (cf. [9] p. 16). As mentioned BPDM and BPMN are driven by the OMG. Presumably, further standardisation efforts will merge BPDM and BPMN although this day is far ahead. BPDM is currently in final submission state (since March 5, 2007).

### 3 Working Assumptions and Goals

While this paper focuses on model information, assumptions have been made about the environment described in Figure 1. In this section we state the goals for this paper based on the assumptions that we make. The assumptions are as follows:

- All BPMN [10] control flow constructs are available.
- We assume that a workflow data concept exists. Thus, it is possible to access and manipulate data on a process level.
- We do not assume that data outside the scope of the current process instance is directly accessible. In particular, we do not assume that an expression within a process can reference organisational or history data provided by the workflow environment. Such information can be brought into the process scope by a service call and referenced locally.
- We do not specify how one out of multiple possible resources gets offered or allocated a task. We restrict ourselves to describe how the possible resources are referenced.
- We assume that resources have attributes such as name, role and organisational information. However, we do not specify them here in this paper. The amount and quality of information is very specific for the application domain. Any proposal would be necessarily incomplete.

- Resources can deallocate, suspend and resume tasks. This is a purely human step which might be restricted by control-flow means.
- The act to delegate, skip or redo a task is purely a human decision taken at runtime on the basis of a single task.

Given the assumptions our goal is to provide a resource layer for BPMN. The solution should be aligned with the BPMN behavioural semantics and the BPDM metamodel. It is neither the goal to provide a full metamodel nor to re-specify BPMN. This work focuses on the resource information in process models.

## 4 A Sample Process

To demonstrate what set of information can be relevant in a process model we consider the sample given in Figure 2. A reporting process starts with a clerk creating a report. The report has to be read by another clerk who might edit the report before approving it. Approved reports can be released by a manager.

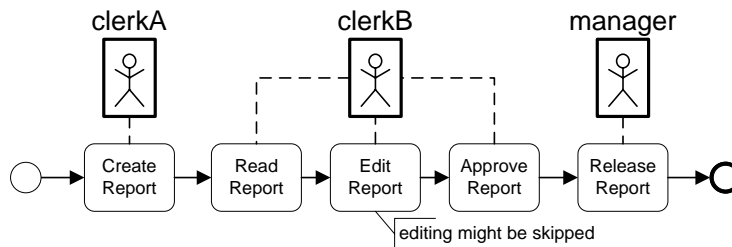


Figure 2: Reporting Process

The sample process involves three different resources (ClerkA, ClerkB, Manager). The model needs to contain information about the roles participating in the execution of the tasks (clerks, managers). It furthermore needs to capture that ClerkA and ClerkB, although they have the same role, need to be different persons. This concept is often referred to as the *Separation of Duties* or the *4-Eyes principle* [6]. Finally, the process step *Edit Report* might be skipped. Skipping is conceptionally different from a data-based routing decision (control flow perspective) because it is a manual decision made by ClerkB at commencement of the task. The sample pinpoints quite some issues that process modellers might want to express at design time. These information must be transported in the process model. We will refer to that sample in the remainder of this paper where suitable.

## 5 The Resource Information Framework

In Section 2 we presented the related research in the area of resource information for workflow systems. In this section we choose a framework of information to be captured in workflow models based on that research.

For this paper we rely on the Workflow Resource Patterns [4] (WRP). They describe the state-of-the-art from the resource perspective on workflow systems.

We consider the patterns as highly relevant as they have been identified by examining various industry standards and vendor implementations. Referring to the related work (see Section 2) we can state that it is the most comprehensive framework in that area. Other work such as BPEL4People [6] only captures a subset of the aspects described in the Workflow Resource Patterns. For that reason we use the Workflow Resource Patterns (WRP) to benchmark the resource capabilities for BPDM, BPMN and our solution.

However, not all patterns described in [4] are related to model information. As an example, Selection Autonomy (WRP-26) deals with the ability to select an arbitrary task from the work list. This is a work list feature but nothing to benchmark the expressiveness of the modelling language. Other patterns deal with a particular behaviour which is a convention rather than model information. We specify our framework in the next paragraphs by examining the forty three Workflow Resource Patterns in the seven categories and filtering out the relevant patterns for our context.

**Creation Patterns** correspond to restrictions on the manner in which specific work items can be advertised, allocated and executed by resources (cf. [8]).

In general, the creation patterns are relevant. In particular, Capability-Based Allocation (WRP-08), History-Based Allocation (WRP-09) and Organisational Allocation (WRP-10) are out of scope for this work as they require the ability of the workflow system to provide environment information (such as organisational data) to the process instance. We stated in Section 3 that we do not assume access to data outside the process instance. Furthermore, we described a work around to make external information local to the process context. The remaining creation patterns relevant for this context are [4]:

- WRP-01 Direct Allocation - The ability to specify at design time the identity of the resource that will execute a task.
- WRP-02 Role-based Allocation - The ability to specify at design time that a task can only be executed by resources which correspond to a given role.
- WRP-03 Deferred Allocation - The ability to defer specifying the identity of the resource that will execute a task until runtime.
- WRP-04 Authorization - The ability to specify the range of resources that are authorised to execute a task.
- WRP-05 Separation of Duties - The ability to specify that two tasks must be allocated to different resources in a given workflow case.
- WRP-06 Case Handling - The ability to allocate the work items within a given workflow case to the same resource.
- WRP-07 Retain Familiar - Where several resources are available to undertake a work item, the ability to allocate a work item within a given workflow case to the same resource that undertook a preceding work item.
- WRP-11 Automatic Execution - The ability for an instance of a task to execute without needing to utilise the services of a resource.

**Push Patterns** describe situations where a workflow system proactively offers or allocates work to resources (cf. [8]).

The push patterns deal with the way work is allocated to resources by the workflow system if choices are possible. In a situation where multiple resources

can get allocated the same task that might happen through Round Robin (WRP-16), Random (WRP-15) or Shortest Queue (WRP-17) algorithms. We stated in Section 3 that we consider this to be a feature of the workflow environment, not model information. Hence, none of the Push Patterns are relevant for our context.

**Pull Patterns** characterise scenarios where resources initiate the identification, allocation and execution of work [8].

Analogue to the push patterns, the pull patterns are not related to model information. They deal with the way resources can find and commit to available tasks at runtime. The pull patterns describe workflow engine configuration and workflow client features but no information to be captured in the process model. Hence, none of the Pull Patterns are relevant for our context.

**Detour Patterns** describe deviations from the normal sequence of state transitions for work items initiated by resources [8].

In general, the detour patterns are relevant for our context. Among other things, they capture the ability to delegate, escalate, skip or redo activities. At design time it is clear under which circumstances a work item can be e.g. delegated. Restrictions can be captured in the process description.

Out of scope here are the Stateful/Stateless Reallocation patterns (WRP-30/31). We consider it as a workflow environment feature to select a way of doing reallocation. Ideally, both alternatives are offered and the resource can choose upon reallocation to keep or skip the old state of the task. The ability to *Pre-Do* (WRP-35) an activity before its preceding activities have been completed is out of scope. To pre-do a task implies an early distribution mechanisms (WRP-19, out of scope). This is contrary to the BPMN semantics which intuitively creates and enables a task when the corresponding number of tokens arrive (not earlier). Since the precondition (WRP-19) is out of scope the *Pre-Do* pattern (WRP-35) is either. The remaining detour patterns relevant for this context are [4]:

- WRP-27 Delegation - The ability for a resource to allocate a work item previously allocated to it to another resource.
- WRP-28 Escalation - The ability of the workflow system to offer or allocate a work item to a resource or group of resources other than those it has previously been offered or allocated to in an attempt to expedite the completion of the work item.
- WRP-33 Skip - The ability for a resource to skip a work item allocated to it and mark the work item as complete.
- WRP-34 Redo - The ability for a resource to redo a work item that has previously been completed in a case.

**Auto-Start Patterns** relate to situations where the execution of work is triggered by specific events or state transitions in the business process [8].

BPMN intuitively supports Commencement on Creation (WRP-36) and Chained Execution (WRP-39) [7]. Both patterns relate to the genuine behaviour in which BPMN tasks are started and interlinked using control-flow. Commencement on Allocation (WRP-37) is an alternative point in time to start an activity

<b>Relevant</b>	<b>Out-Of-Scope</b>
01 Direct Allocation	08 Capability-Based Allocation
02 Role-Based Allocation	09 History-Based Allocation
03 Deferred Allocation	10 Organisational Allocation
04 Authorisation	12-20 Push Patterns
05 Separation of Duties	21-26 Pull Patterns
06 Case Handling	29 Deallocation
07 Retain Familiar	30 Stateful Reallocation
11 Automatic Execution	31 Stateless Reallocation
27 Delegation	32 Suspension-Resumption
28 Escalation	35 Pre-Do
33 Skip	36-39 Piled Execution
34 Redo	40-41 Visibility Patterns
43 Additional Resources	42 Simultaneous Execution

Table 1: Resource Patterns Classification Overview

in contrast to the point of creation. Since Commencement on Creation (WRP-36) is already supported we chose to align with that and do not seek support for the alternative. Finally, Piled Execution (WRP-38) is the ability to optimise task execution using queues per resource. Resources can get tasks of the same type allocated one after another to increase throughput. However, there is no support for this patterns in no known workflow environment [5]. None of the auto-start patterns require model information and thus they are considered to be out of scope for our framework.

**Visibility Patterns** describe the ability of resources to view the status of work within the workflow environment (cf. [8]).

The visibility patterns deal with workflow client features, precisely with work list features. The ability to show or hide information about allocated or unallocated work is not specific to a process instance or a process description. Hence, these patterns are not relevant for our context.

**Multiple Resource Patterns** describe scenarios where there is a many-to-many relationship between specific work items and the resources undertaking those work items [8].

The Simultaneous Execution patterns (WRP-42) is already supported in BPMN [8]. It deals with the ability to assign multiple tasks to the same resource at the same time. In BPMN there is no restriction and thus it is possible to assign concurrently running tasks to the same resource. Additional model information is not required. Requesting Additional Resources (WRP-43) is a manual step performed at runtime. Thus, this can not be modelled at design time but the model should keep the information about the resources involved in a particular activity instance. At design time restrictions about possible additional resources might be captured in the model. Hence, the multiple resource pattern relevant for our context is [4]:

- WRP-43 Additional Resources - The ability for a given resource to request additional resources to assist in the execution of a work item that they are currently undertaking.

**Patterns Classification Overview** In the last paragraphs we distinguished those patterns which need information in the model in order to be supported and those that do not. That does not mean that the other patterns are not supported. Indeed some are supported already such as Simultaneous Execution (WRP-42) through the commonly agreed behavioural semantics in BPMN. The classification is not a judgement about the usefulness of a pattern. We distinguish both classes of patterns to identify those which require model information. These build the framework for our further assessments and extensions. Table 1 gives an overview about the Workflow Resource Patterns classification done in this section.

## 6 Assessment of current Resource Information Support

After we have identified the framework of patterns in Section 5 we now examine the Business Process Modelling Notation (BPMN) and the Business Process Definition MetaModel (BPDM) for their abilities to capture the required resource information. The result is the identification of abilities and shortcomings in both standards.

### 6.1 Assessment of BPMN

BPMN [10] supports a swimlane concept to distinguish different participants in a process or a collaboration of processes. According to the standard *pools* represent participants in a process, e.g. a company, IT system or a role (cf. [10] p.111). Pools can be subdivided using lanes, where "The meaning of the Lanes is up to the modeler" ([10] p. 115). Pools and Lanes have a name attribute only. The value is depicted at the head of the pool or lane. Figure 3 depicts the sample from Figure 2 and uses lanes to depict the different roles. Other uses of pools and lanes are possible such as showing the organisational unit at which an activity is executed. Hence, the swimlane concept in BPMN is rather open and unspecific.

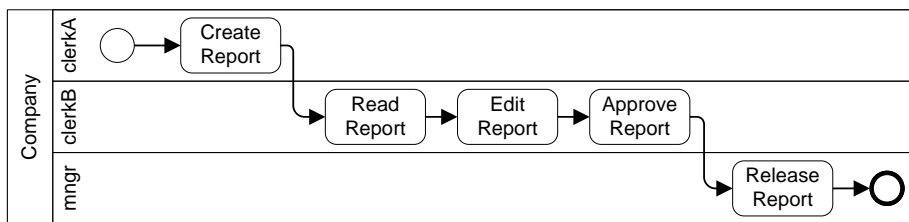


Figure 3: Sample process with different participants illustrated using the BPMN swimlane concept

In contrast to that, BPMN offers a specific attribute at the activity level. For user tasks the attribute *Performers* can optionally be set to identify the person(s), group(s) or organisational units which will perform the task (cf. [10] p. 90). This concept is more specific than the swimlane concept as it is manifested in a dedicated attribute at activity level. Thus, each activity captures

the information about their required resource. But the attribute is not very comprehensive and additional attributes may be added to adequately capture the information needed (cf. [10], p. 90, table 9.21).

**To assess the BPMN resource capturing abilities** for the set framework we rely on the *Performers* attribute for user tasks in BPMN. We can specify the role (WRP-02<sup>+</sup>) or the person (WRP-01<sup>+</sup>) to perform a task. The attribute type does not allow to specify variables to defer the allocation decision to runtime (WRP-03<sup>-</sup>). Consequently, it is also not possible to base the resource specification on already completed tasks in the same case (WRP-07<sup>-</sup>). As we said BPMN *Performers* are specified on a task level. Even if one single resource is specified to be responsible for all tasks within a process, it does not meet the criteria for case handling (WRP-06<sup>-</sup>) which demands the responsibility for all tasks to be specified on a process level.

BPMN has no means of restricting the allocation other than specifying a role or a person. Thus, we cannot realise the separation of duties (WRP-05<sup>-</sup>). Such a constraint could also be used to express authorization concerns (WRP-04<sup>-</sup>). Beyond that, a constraint would be helpful to restrict delegation (WRP-27<sup>-</sup>) and the option to require additional resources (WRP-43<sup>-</sup>) for a task. But additional resources and task delegation are not addressed in the BPMN specification and therefore unsupported. The same is true for escalation (WRP-28<sup>-</sup>) and there is no realisation possible with the given capabilities in BPMN.

The BPMN semantics assumes that a once initiated task is performed exactly once (or cancelled). Activities might lead to multiple tasks (activity instances) but it is not possible to redo (WRP-34<sup>-</sup>) or skip a task (WRP-33<sup>-</sup>). However, it is possible to leave the *Performers* attribute open and thereby imply that the task is executed automatically (WRP-11<sup>+</sup>) by the system.

In a few words, in BPMN modellers intuitively depict resource information through the name of the swimlanes in which a tasks are placed. This is not formal though as the swimlane concept is much broader than that. The ability to specify a performer at task level leads to very limited support of only three out of thirteen patterns in our framework.

## 6.2 Assessment of BPDM

The Business Process Definition MetaModel standard defines a metamodel to capture information relevant for business processes which also includes resource information. Figure 4 shows an excerpt from the BPDM Activity Model for orchestrations.

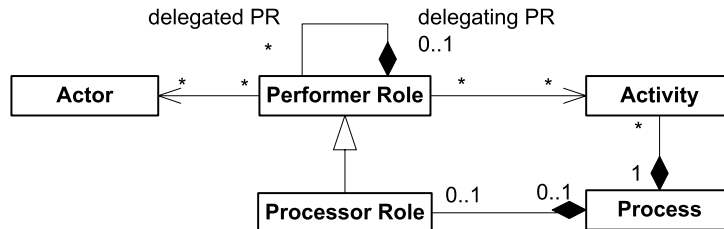


Figure 4: Simplified BPDM Activity Model

BPDM has *Performer Roles* specified to be responsible for an *Activity*. A specialised version of that is the *Processor Role* which is responsible for a *Process*. Performer Roles can delegate responsibility to other Performer Roles. A performer role is mapped to an actor at runtime who actually performs the task.

**The BPDM resource capturing abilities** allow to specify a role (WRP-02<sup>+</sup>) to perform a task. An actual person can not be named directly. We consider a work around where a performer role maps to one person only (WRP-01<sup>+</sup>). Like in BPMN it is not possible to reference variables or defer allocation decisions (WRP-03<sup>-</sup>) to runtime. The *Performer Role* is the only mechanism to specify distribution. Consequently, there is no means to further restrict the allocation to express authorization (WRP-04<sup>-</sup>) concerns or the separation of duties (WRP-05<sup>-</sup>). Such concerns have been explicitly not in focus for BPDM (cf. [9] p. 14).

But one *Performer Role* can be associated with multiple activities within a process. Thus, we can express that the same actor executes a set of activities (WRP-07<sup>+</sup>) within a case. We can even express responsibility of a resource for all activities within a process (WRP-06<sup>+</sup>) on a process level using the *Processor Role*. BPDM explicitly captures the ability to delegate (WRP-27<sup>+</sup>) work to other roles. However, it is not clear how such mechanisms can be controlled or restricted to ensure the responsible resource actually performs the task. In contrast to the delegation capabilities, there is no association to track which additional resources (WRP-43<sup>-</sup>) have been working on a particular task.

BPDM purely targets the static model information. As such it can not capture escalation of tasks (WRP-28<sup>-</sup>) as this is a behavioural concern combined with the ability to manipulate allocation data at runtime. Analogously, the behavioural concerns to redo (WRP-34<sup>-</sup>) or skip (WRP-33<sup>-</sup>) a task are not addressed either. Finally, like in BPMN the activities in BPDM do not need a performer to be specified. If no performer is set, the task is assumed to be executed automatically (WRP-11<sup>+</sup>) by the system.

In a few words, information captured in the BPDM metamodel is far more specific than the one in BPMN. Hence, BPDM fully supports six patterns from our framework. However, seven patterns remain unsupported leaving space for improvements.

### 6.3 Assessment Conclusion

In Section 5 we identified a framework of resource patterns relevant for our context. In last two Sections we assessed BPMN and BPDM for their support of the framework. We now summarize the results. Table 2 in Section 8 provides an overview about the pattern support.

BPMN and BPDM allow to specify roles to perform tasks. Both can through role mapping or direct definition specify concrete persons to conduct a piece of work and both support automated execution of tasks by simply leaving the resource specification open. BPDM captures quite some more information than BPMN. In particular, it allows to track task delegation and specify that multiple tasks within a case should be executed by the same person. As we have stated in the introduction, it is likely that BPDM becomes the metamodel foundation for the next BPMN version. It can be expected that BPMN inherits BPDM capabilities and thereby gains from the extended BPDM patterns support. We

aim to propose a solution aware of all relevant model information and aligned with the current standardisation efforts. That means, we avoid changes and minimize extensions.

## 7 Solution Proposal

We examined in Section 6 the support for our framework in BPMN and BPDM. We just stated that our goal is improved support while aligning with the standards. In this section we propose a metamodel extension based on BPDM (7.1), define the behavioural aspects of our solution (7.2) based on the intuitive semantics in BPMN and discuss the side effects (7.3) to other workflow perspectives.

### 7.1 Metamodel Proposal

Starting from the BPDM Activity Model ([9] p. 99) we extend the metamodel in Figure 4 with attributes and associations to capture the yet missing patterns and discuss their purpose.

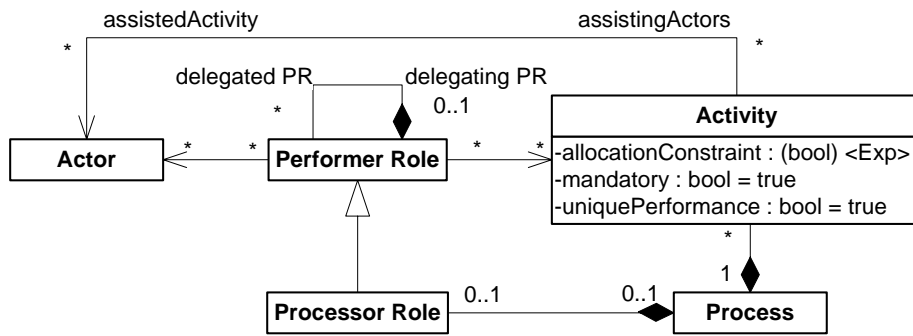


Figure 5: Enriched BPDM Activity Model

Figure 5 shows an extended metamodel known from Figure 4 with new activity attributes and a new association between *Activity* and *Actor*.

The new association illustrates the relationship of actors with activities in which they participate. The multiplicity is unbounded just as the option to request additional resources for a task. Analogue to delegation the decision to request assistance in completing a task is taken at runtime by the actor who performs the current task. The model can only track this fact. That allows us at design time to express limitations or refer to e.g. all actors which assisted in performing a task.

The attributes *allocationConstraint*, *mandatory* and *uniquePerformance* were added to the *Activity* to capture constraints at design time for the allocation and execution of tasks at runtime.

The attribute *mandatory* is a Boolean value which determines whether the outcome of the activity is crucial to the process success. If set to *false* then the activity can be skipped. This is directly related to the Skip pattern (WRP-33). Analogously, the attribute *uniquePerformance* is dedicated to the Redo pattern (WRP-34). It indicates whether a previously completed task (activity instance) can be performed again.

The attribute *allocationConstraint* holds a boolean expression which must be *true* whenever the activity is assigned to an actor. This holds for the initial assignment, all delegations and all assisting actors. A possible constraint could be that the actors first name is 'Hans'. That means, whoever is performing or assisting in the performance of this activity must be named 'Hans'. We already described in Section 3 that we assume actors to have attributes exposing their characteristics (such as names) although we do not specify these attributes here.

## 7.2 Behavioural Aspects

While Section 7.1 describes the static information, this Section illustrates the impact on the activity lifecycle implied by the additional resource information.

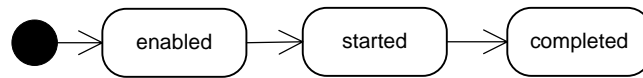


Figure 6: Simplified Activity Lifecycle without Resource Dependencies

Figure 6 depicts a simplified activity lifecycle model for tasks without human interactions. In this simplified model an activity is *enabled*, then *activated* and finally *completed*.

For tasks performed by humans we expand the lifecycle model as shown in Figure 7. Accordingly a task is *not allocated* after being *enabled*. When a task changes its state from *not allocated* to *allocated* the *allocationConstraint* has to be met (1). Allocation is the act of assigning a resource to a task. An allocated task might be re-allocated (2) to another resource by delegation. When delegating a task the *allocationConstraint* has to be met just as for the state transition from *not allocated* to *allocated*.

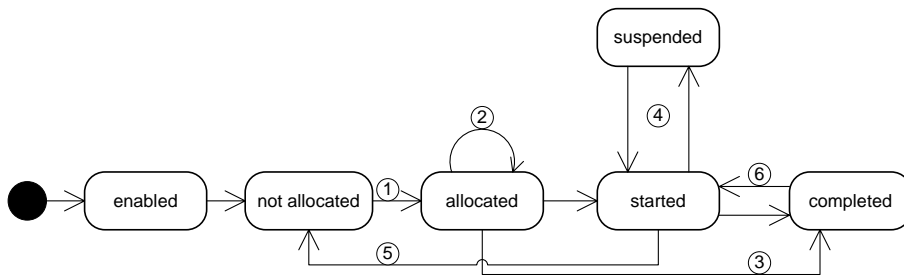


Figure 7: Extended Activity Lifecycle without Resource Dependencies

Allocated tasks can change their state to *started* or *completed*. Changing the state from *allocated* to *completed* (3) can happen through skipping if the attribute *mandatory* for the activity was set to *false*. A *started* task might also change its state to *suspended* and back to *started* (4) as we discussed in Section 3. Suspending and resuming an activity might be restricted through other means, nevertheless we depict it here in Figure 7 as we assumed that it is possible.

A *started* task can be deallocated by the executing resource. In that case the state of the activity changes from *started* back to *not allocated* (5). The

task is now free to be allocated again as explained above. Ultimately, an *started* task changes its state to *completed*. In contrast to our original model without resources involved *completed* is not necessarily the final state. If the activity attribute *uniquePerformance* is set to *false* an already *completed* task can re-enter the state *started* (6) and be performed again.

In the activity lifecycle models we abstract from cancellation. A task might be cancelled in state *allocated*, *started* or *suspended* which is all not shown here. But cancellation is needed to realise escalation as we discuss in Section 8.

### 7.3 Side Effects to other Perspectives

As we described in the Section 7.2 the behaviour of tasks changes if resources are involved in their execution. This has side effects to other workflow perspectives as we discuss now.

In particular, from a control-flow point of view a BPMN task is enabled when it has consumed the according number of tokens. Such a task could change its state to *started* without further conditions. Through the introduction of the resource perspective further requirements (e.g. resource allocation) have to be met before the execution of an activity can start.

Significant impact is generated by the ability to skip or redo tasks. This results in changed control-flow semantics. If a task is redone, subsequent tasks might have to be redone as well. Earlier instances might need to be compensated. Finally, a point of no return might be explicitly defined by the control-flow designer to determine when a task must be stable (not re-doable any more). Analogue considerations can be done for the impact of skipped tasks to the control-flow semantics.

From a data-flow perspective, we already described in Section 3 that we assume resource data to be treated as workflow data. The resource perspective can be influenced by advanced data accessibility concepts. It might gain from environment data concepts (e.g. direct access to history data) or be limited by internal data visibility restrictions (e.g. scopes).

## 8 Validation of the extended BPMN

We now validate the proposed solution from Section 7 against the framework from Section 5. The validation is an assessment of our extended BPMN proposal analogue to the assessment we did in Section 6.

Our metamodel solution is based on the BPDM activity model (see Figure 4). Hence, we can capture the same information. We can thereby support Role-based Allocation (WRP-2<sup>+</sup>) by referring to a Performer Role. We can capture Direct Allocation (WRP-1<sup>+</sup>) by specifying a role that maps to one actor only (workaround for BPDM assessment) but furthermore we can also express that using the new activity attribute *allocationConstraint*. It must be met whenever an activity is allocated to a resource and might reference actor information (such as the name). Furthermore, we can use the new attribute *allocationConstraint* to enhance support for a variety of patterns.

Since we allow variables to be referenced in the *allocationConstraint* we can defer the allocation decision to runtime (WRP-3<sup>+</sup>). The attribute can also be

used to encode authorization concerns in the distribution mechanism (WRP-4<sup>+</sup>/-). However this is considered a work around leading to partial support [4] only.

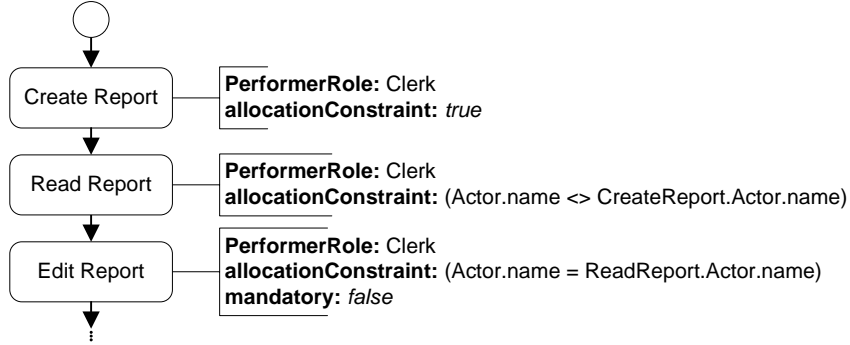


Figure 8: Extended concepts applied to an excerpt of the sample process from Figure 2

The expression kept in the *allocationConstraint* can also be used to define exclusions. In other words, it is possible to state that an actor performing a task should not be the same that executed a preceding task in that case. This ability is called the 4-eye-principle or separation of duties (WRP-5<sup>+</sup>).

As in BPDM we can capture the responsibility of a role to perform multiple tasks within a process (WRP-7<sup>+</sup>) or even all tasks (WRP-6<sup>+</sup>). On a process level, we can now enforce that all tasks are allocated to the same resource using the *allocationConstraint* and referencing the *Processor Role*. As we just described the new attribute *allocationConstraint* is quite powerful to express limitations and conditions for task assignment.

We introduced new activity attributes, *mandatory* and *uniquePerformance*, which are dedicated to the support of Skip (WRP-33<sup>+</sup>) and Redo (WRP-34<sup>+</sup>). If the attribute *mandatory* is set to *true* the task must be executed, otherwise it might be skipped. Analogously, if the attribute *uniquePerformance* is set to *true* the task can be executed at most once, otherwise the same instance might be redone.

To capture and track the additional resources (WRP-43<sup>+</sup>) assigned to a task, we introduced a new association between actors and tasks into the metamodel. We can thereby track who participated in completing a task and also apply restrictions to that. The delegation ability, captured through the association between performer roles in BPDM, can be utilized in the same way (WRP-27<sup>+</sup>).

Automatic Execution (WRP-11<sup>+</sup>) is supported by making all resource information for activities optional. If no attribute is set, then the activity is executed without the service of resources, just like in BPMN and BDPM.

In Figure 8 we depict an excerpt of the sample process from Figure 2 to illustrate the application of our concepts. For better readability we hide attributes with default values (see metamodel in Figure 5) and abstract from instance identifiers. Note, that we reference trivial actor attributes although they have not been formalized in this work (see Section 3). Moreover, we employ a pseudo syntax for the *allocationConstraint* expression which is not normative. We de-

mand that the result must be a boolean value. The choice for an expression language is a technical decision and not considered in this paper.

**Escalation** (WRP-28) is the only patterns which was not mentioned yet. We state that escalation can be realised by combining control flow, data flow and resource abilities. In particular, the ability to cancel and restart an activity, data manipulation and the ability to reference process data to identify resources.

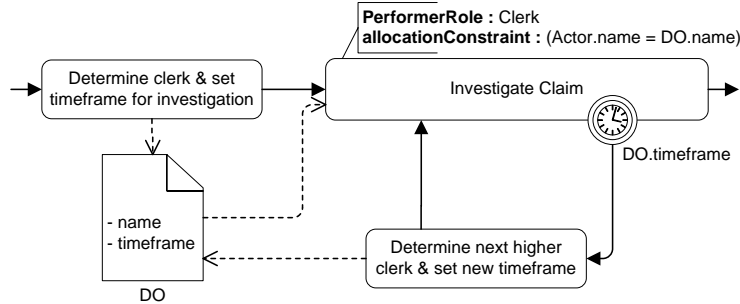


Figure 9: Escalation (WRP-28) Sample in BPMN

Figure 9 depicts a part of an approval process with escalation. A clerk has a defined timeframe to investigate a claim. The time frame and the clerk are elected at runtime by a previous activity. The information is kept in the process scope, here in a DataObject (DO). The activity *Investigate Claim* can only be allocated to a person with the role *clerk* and a name specified in the *allocationConstraint*. The constraint refers to the *name* value kept in *DO*. If the activity is not completed once the time elapses the *Investigate Claim* activity is cancelled. Subsequently, a new timeframe is set and a new clerk is determined. This is done by changing the values in *DO*. To escalate a new instance of *Investigate Claim* is started and assigned to a clerk dependent on the values in the *DO*. As illustrated in this sample, we can achieve Escalation (WRP-28) by combining the control flow, data flow and extended resource capabilities.

**The validation in a nutshell** proves that the extensions proposed in Section 7 can be used to extend the support for the patterns in our framework. More precisely, we can fully support 12 out of 13 patterns. Table 2 gives an overview about the pattern support in our extended BPMN and compares it with the results from our initial assessment in Section 6. Authorization is partially supported, because our solution incorporates authorization as a part of the distribution mechanism. A full support would require an authorization framework independent from work distribution.

## 9 Conclusion

This paper started with background about humans in process models. We motivated the need for more information in the model and identified a framework of resource relevant information based on the Workflow Resource Patterns. We then examined the Business Process Modelling Notation (BPMN) and the

WRP	BPMN	BPDM	extended BPMN
01 Direct Allocation	+	+	+
02 Role-Based Allocation	+	+	+
03 Deferred Allocation	-	-	+
04 Authorisation	-	-	+/-
05 Separation of Duties	-	-	+
06 Case Handling	-	+	+
07 Retain Familiar	-	+	+
11 Automatic Execution	+	+	+
27 Delegation	-	+	+
28 Escalation	-	-	+
33 Skip	-	-	+
34 Redo	-	-	+
43 Additional Resources	-	-	+

Table 2: Comparison of Framework Support in BPMN, BPDM and our extended BPMN Solution

Business Process Definition MetaModel (BPDM) for their abilities to capture relevant resource information. Based on these standards we proposed an extended metamodel aligned with BPMN and BPDM to capture all model related resource information. We furthermore defined the behavioural aspects of activities driven by users. The contributions of this paper are the identification of resource patterns relevant for workflow models, the assessment of BPMN and BPDM abilities and shortcomings. Finally, our result contributes to the OMG standardisation efforts by proposing aligned extensions attributes and defining their semantics. The solution is applied to BPMN and BPDM but can be analogously used to assess and extend other languages for the same purpose.

**Further research** needs to investigate industry use cases to prove the applicability of the proposed extensions or reveal shortcomings. Furthermore our BPMN resource model must be merged with a control-flow and data-flow concept in order to provide a complete language. Based on such an integrated language a design tool and an engine could be built to prove the applicability of our extensions. For the engine environment further research is needed to specify features such as work lists and allocation algorithms which are captured in the Workflow Resource Patterns [4] but were out of scope here.

## References

- [1] Fallside, David C. and Priscilla Walmsley: *Web Services Business Process Execution Language Version 2.0*. Technical report, Oct 2005. <http://www.oasis-open.org/apps/org/workgroup/wsbpel/>.
- [2] Tabeling, Peter: *Softwaresysteme und ihre Modellierung : Grundlagen, Methoden und Techniken*. Springer Verlag Berlin, 2006.
- [3] Aalst, WMP van der and A. Kumar: *A reference model for team-enabled workflow management systems*. *Data & Knowledge Engineering*, 38(3):335–363, 2001.

- [4] Russell, N.: *Workflow Resource Patterns*. Beta, Research School for Operations Management and Logistics, 2005.
- [5] Russell, N., W.M.P. van der Aalst, A.H.M. ter Hofstede, and D. Edmond: *Workflow Resource Patterns: Identification, Representation and Tool Support*. Proceedings of the 17th Conference on Advanced Information Systems Engineering (CAiSE 05), 3520:216–232.
- [6] Kloppmann, M., D. Koenig, F. Leymann, G. Pfau, A. Rickayzen, C. von Riegen, P. Schmidt, and I. Trickovic: *WS-BPEL Extension for People-BPEL4People*. Joint white paper, IBM and SAP, July, 2005.
- [7] Wohed, P., WMP van der Aalst, M. Dumas, AHM ter Hofstede, and N. Russell: *Pattern-based Analysis of BPMN-An extensive evaluation of the Control-flow, the Data and the Resource Perspectives*. BPM Center Report BPM-05-26, BPMcenter. org, 2005.
- [8] Wohed, P., WMP van der Aalst, M. Dumas, AHM ter Hofstede, and N. Russell: *On the Suitability of BPMN for Business Process Modelling*. Technical report, Queensland University of Technology (QUT), <http://www.bpm.fit.qut.edu.au/projects/babel/docs/BPMN-eval-BPM06.pdf>, 2006.
- [9] (OMG), Object Management Group: *Business Process Definition Meta-Model (BPDM) - Final Submission*, 2007.
- [10] BPMI.org: *Business Process Modeling Notation Specification 1.0. Final Adopted Specification*, 2006.